Universidade Federal da Bahia

Universidade Estadual de Feira de Santana

DISSERTAÇÃO DE MESTRADO

Sistema de Recomendação baseado em conteúdo textual: avaliação e comparação

Rafael Glauber Nascimento e Silva

Mestrado Multiinstitucional em Ciência da Computação

MMCC

Salvador – BA

2014

Rafael Glauber Nascimento e Silva

Sistema de Recomendação baseado em conteúdo textual: avaliação e comparação

Dissertação apresentada ao Programa Multi-institucional em Ciência da Computação da Universidade Federal da Bahia (UFBA) e a Universidade Estadual de Feira de Santana (UEFS) como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Angelo Conrado Loula

Feira de Santana, BA
10/2014

Ficha Catalográfica – Biblioteca Central Julieta Carteado

Silva, Rafael Glauber Nascimento e
S583s Sistema de Recomendação baseado em conteúdo textual: avaliação e comparação / Rafael Glauber Nascimento e Silva. – Feira de Santana, 2014.

122 f.: il.

Orientador: Prof° Dr. Angelo Conrado Loula

Mestrado (dissertação) – Universidade Estadual de Feira de Santana, Universidade Federal da Bahia, Programa Multi–institucional em Ciência da Computação, 2014.

1. Sistema de Recomendação. 2. Conteúdo Textual. 3. Filtragem Colaborativa. 4. Avaliação I. Loula, Angelo Conrado, orient. II. Universidade Estadual de Feira de Santana. III. Universidade Federal da Bahia. IV. Título

CDU: 681.3

RAFAEL GLAUBER NASCIMENTO E SILVA

SISTEMA DE RECOMENDAÇÃO BASEADO EM CONTEÚDO TEXTUAL: AVALIAÇÃO E COMPARAÇÃO

Esta Dissertação foi julgada adequada à obtenção do título de Mestre em Ciência da Computação e aprovada em sua forma final pelo Programa Multi-institucional de Pós-Graduação em Ciência da Computação da UFBA-UEFS.

Salvador, 17 de outubro de 2014.

Prof. Dr. Ângelo Conrado Loula (orientador), D.Sc.
Universidade Estadual de feira de Santana

Prof. Dr. João Rocha Júnior, D.Sc.
Universidade Estadual de feira de Santana

Prof. Dr. Leandro Nunes de Castro Silva, D.Sc.
Universidade Presbiteriana Mackenzie

Resumo

Sistemas de Recomendação sugerem itens de interesse explorando as preferências dos usuários ajudando-os contra o problema da sobrecarga de informações. Primeiramente estes sistemas eram construídos, exclusivamente, através de técnicas com origem nas áreas de Recuperação de Informação e Aprendizado de Máquina. Porém, desde o início da década de 90 a abordagem conhecida como Filtragem Colaborativa, que não explora qualquer tipo de conteúdo disponível dos itens para realizar sua tarefa, passou a ser a mais utilizada como solução para estes sistemas. Pesquisas como as de Shardanand & Maes (1995); Das et al. (2007); Konstan & Riedl (2012) justificam essa preferência por conta de deficiências preexistentes nos algoritmos de filtragem por conteúdo dos itens. Entretanto, nestas pesquisas não são evidenciadas essas deficiências ou mesmo as diferenças e semelhanças das recomendações geradas pelos algoritmos dessas duas abordagens levando esta discussão ao senso comum. Neste trabalho é proposta uma metodologia para comparação de algoritmos de recomendação que vai além da precisão das previsões. Para demonstrar essa metodologia a aplicamos na comparação das abordagens de Filtragem Baseada em Conteúdo Textual e a Filtragem Colaborativa. Nossos resultados demonstram que algoritmos dessas duas abordagens não só diferem em diversas dimensões em um teste de sistema, como também apresentam características que sugerem grande complementariedade.

Palavras-chave: Sistema de Recomendação, Conteúdo Textual, Filtragem Colaborativa, Comparação, Avaliação.

Abstract

Recommender Systems suggest items of interest by exploring users' preferences, helping them in the information overload problem. At first, these systems were built exclusively using techniques from Information Retrieval and Machine Learning. However, since early 90s, the Collaborative Filtering approach, which does not use the content available in items to accomplish its task, has become the most widely used solution for these systems. Research such as that from Shardanand & Maes (1995); Das et al. (2007); Konstan & Riedl (2012) justify this preference on account of deficiencies in the existing content-based filtering algorithms. However, in these surveys there is no further evidence of such deficiencies or even the differences and similarities of the recommendations generated by algorithms from these two approaches, directing this discussion only to common sense. This work proposes a methodology for comparing recommendation algorithms that go beyond the accuracy of forecasts. To demonstrate this methodology, we applied it to compare Textual Content Based Filtering and Collaborative Filtering approaches. Our results show that these algorithms not only differ in various dimensions in a back-test, but also have characteristics suggesting a high complementarity.

Keywords: Recommender System, Collaborative Filtering, Textual Content, Comparison, Evaluation.

Agradecimentos

Ao meu orientador Angelo Loula que não só colaborou com este trabalho, mas também me guiou na conquista deste sonho. Serei eternamente grato por todo o tempo dedicado e conhecimento compartilhado. Certamente sou um acadêmico melhor desde que começou essa convivência e espero que este seja mais um de muitos outros trabalhos.

Ao Jakmuller Cedraz por ceder seu equipamento para realização dos experimentos e por toda a solicitude diante das diversas requisições.

À Próton Sistemas Ltda nas figuras de Adolfino Neto, Paulo Rios e Wilson Prado que me proporcionaram a ausência durante todo o tempo necessário para a realização desse trabalho.

À CAPES e à FAPESB pela concessão da bolsa de estudos.

Aos meus filhos Pedro e Catarina que amo de uma forma que não sei explicar e ao meu amor Juliana Vital.

Sumário

Lista de Figuras

Lista de Tabelas

Glossário

AIP Filtragem por Agregação dos Itens do Perfil

AM Aprendizado de Máquina

AveP Average Precision

BC Baseado em Conteúdo

BOW Bag-Of-Words

CCOV Catalog Coverage

DF Document Frequency

FC Filtragem Colaborativa

IDF Inverse Document Frequency

IMDb Internet Movie Database

kNN k-Nearest Neighbor

MAP Mean Average Precision

RI Recuperação de Informação

SIP Filtragem pela Similaridade dos Itens do Perfil

SR Sistemas de Recomendação

TF Term Frequency

UCOV User Coverage

Capítulo 1

Introdução

A popularização da Internet e os diversos mecanismos de compartilhamento de conteúdo neste ambiente nos apresenta a um paradoxo: não mais a escassez, mas o grande volume de informações têm prejudicado os usuários na tarefa de recuperar o conteúdo que os interessa. Este problema, denominado de *sobrecarga de informação*, passou a ser estudado mais profundamente na década de 90 quando os primeiros Sistemas de Recomendação (SR) surgiram como mecanismos para ajudar aos usuários com essa questão. O primeiro tipo de conteúdo a gerar este problema para os usuários foi o e-mail. Em um estudo feito por Whittaker & Sidner (1996) foi observada a importância desse tecnologia para a comunicação e colaboração entre usuários, porém a grande quantidade de mensagens recebidas prejudicava as outras atividades do ambiente corporativo. Os usuários gastavam muito tempo filtrando as mensagens em um esforço manual.

Após os primeiros trabalhos sobre os Sistemas de Recomendação houve um grande interesse da academia e da indústria sobre o tema. Agora não só e-mail, mas outros tipos de itens passaram a ser recomendados. Sites como *Amazon*[1] utilizam desses sistemas como meio de apresentar ao usuário itens relevantes de acordo com seus interesses. Utilizando um sistema denominado *Cinematch* introduzido em 2000 e aprimorado ao longo dos anos, a *Netflix*[2] obteve 60% de suas vendas realizadas por meio de recomendações (Thompson, 2008). Para o site *Google News*[3] houve um aumento inicial de 38% nos cliques de seus links de notícias após implantação dessa tecnologia (Das et al., 2007). Esses resultados são motivadores e nos últimos anos novos trabalhos, conferências, livros[4] e até mesmo desafios com prêmio de 1 milhão de dólares[5] foram dedicados exclusivamente para esta área.

Neste contexto da sobrecarga de informações, ainda hoje, itens textuais como: e-mail, artigos científicos, notícias e livros estão entre os mais impactantes. Os primeiros trabalhos de recomendação deste tipo de item utilizavam representações das informações de conteúdo sobre os itens e os algoritmos de recuperação de informações atuavam sobre essas representações, normalmente, vetores de termos e palavras ou sentenças. E, apesar de itens textuais possuírem esse importante conteúdo que os caracterizam, a maioria das soluções atuais de recomendação não o utilizam. Ao invés disto, as

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

[2] http://www.netflix.com/

[3] http://news.google.com/

[4] Ver links de livros e outros assuntos relacionados aos Sistemas de Recomendação no site da *ACM Recommender Systems community* em http://recsys.acm.org/

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

abordagens sociais, que ignoram o conteúdo dos itens, passaram a ser utilizadas amplamente dominando em aplicações como e-commerce e redes sociais. Essa abordagem baseia-se no princípio de a partir de um usuário identificar outros usuários com preferências semelhantes (chamados de vizinhos) e, inferindo a popularidade dos itens de sua vizinhança, a recomendação é executada.

A abordagem social, também conhecida como Filtragem Colaborativa, propõe que o trabalho de recuperação de conteúdo relevante deve ser feito de modo colaborativo entre os usuários do sistema, diferentemente da abordagem que utiliza os dados disponíveis do conteúdo. Essa característica dos recomendadores sociais levanta a tese que essa abordagem apresenta melhor desempenho em diversos aspectos ao realizar a tarefa de recomendação em comparação com a filtragem por conteúdo. Trabalhos importantes como o de Goldberg et al. (1992), Shardanand & Maes (1995) e Das et al. (2007) trazem críticas às técnicas de recomendação baseadas em conteúdo, principalmente ao que se refere a surpreender o usuário. Os autores acreditam que somente técnicas sociais podem filtrar itens que, embora possuam características diferentes, despertam o interesse do usuário. Porém, nestes mesmos trabalhos não há avaliações abrangentes o suficiente entre as técnicas que possam comprovar ineficiência aos recomendadores baseados em conteúdo.

Embora a discussão sobre que tipo de abordagem seja superior ou inferior que a outra não seja o foco dessa dissertação, um trabalho comparativo com mais profundidade tem grande relevância para novos estudos na área. De fato, qualquer uma das abordagens possuem características que a tornam mais eficiente em determinado cenário e em outro a tornar menos eficiente. Analisar diferentes dimensões de avaliações sobre as abordagens de recomendação pode nos levar a novas discussões sobre os Sistemas de Recomendação e avançar mais sobre o tema. Vejamos primeiro as questões de pesquisas que vão nos conduzir ao longo deste trabalho na busca de esclarecer melhor as diferenças entre as duas principais abordagens de recomendação.

1.1 Questões de Pesquisa

Os recomendadores que utilizam conteúdo textual podem utilizar representações diferentes do conteúdo, métodos diferentes para identificar novas preferências e critérios diferentes para filtrar as recomendações. Verificar como técnicas de filtragem baseadas em conteúdo textual podem ser aplicadas em Sistemas de Recomendação é uma das atividades realizadas neste trabalho. Muitos sistemas utilizados na web como: *GroupLens* (Resnick et al., 1994), *Fab System* (Balabanović & Shoham, 1997), *Siteseer* (Rucker & Polanco, 1997) e *Google News* (Das et al., 2007) utilizam técnicas sociais e ignoram o conteúdo dos itens. Por conta disso, no desenvolvimento dessa pesquisa algumas questões relacionadas são analisadas e ao final desse trabalho as mesmas devem ser respondidas.

Q1. O conteúdo textual dos itens pode ser uma fonte de dados relevante para Sistemas de Recomendação? O conteúdo textual de um item possui informações que o caracteriza e este tipo de dado não-estruturado é amplamente utilizado em sistemas de Recuperação de Informação e Mineração de Dados. Entretanto, diversas pesquisas questionam a qualidade de técnicas de Recuperação de Informação (RI) e Aprendizado de Máquina (AM) quando aplicadas à Sistemas de Recomendação (Das et al., 2007; Shardanand & Maes, 1995). Em nosso trabalho buscamos utilizar a variedade de conteúdo disponível nos itens do sistema verificando se este tipo de conteúdo é relevante para o tipo de tarefa e contexto no qual o recomendador por filtragem de conteúdo textual está inserido.

Q2. Como utilizar conteúdo textual para recomendação? Pesquisas como as de Phelan et al. (2011); Mak et al. (2003); Van Meteren & Van Someren (2000); Mooney & Roy (2000) apresentam configurações para o recomendador baseado em conteúdo textual e pouca informação sobre as decisões de configuração estão descritas nos artigos publicados. Este fato torna os resultados difíceis de serem reproduzidos e comparados com outros trabalhos. Além disso, é muito difícil perceber nestas pesquisas uma relação entre as técnicas escolhidas e a avaliação da tarefa de recomendação realizada. Desta forma inferir se as decisões tomadas na construção do recomendador são as melhores para o contexto estudado passa a ser um ponto subjetivo. Em nossa pesquisa buscamos explorar diferentes configurações para recomendadores por conteúdo avaliando-os sobre diferentes dimensões.

Q3. Quais as diferenças entre o recomendador Baseado em Conteúdo e Filtragem Colaborativa? Diversos autores, tais quais Das et al. (2007); Shardanand & Maes (1995); Goldberg et al. (1992) atribuem limitações às técnicas utilizadas em Sistemas de Recomendação Baseado em Conteúdo Textual. E, entre as mais citadas temos: não surpreender o usuário e não filtrar baseado em questões subjetivas como qualidade e estilo, mas estes trabalhos não apresentam resultados comparativos que embasem tal suposição. Exploraremos novas dimensões sobre os recomendadores das duas abordagens para que ao final possamos contribuir nesta questão que permeia as pesquisas comparativas entre as duas abordagens.

1.2 Objetivos

A principal área de concentração deste trabalho é Sistemas de Recomendação com especial destaque para a abordagem Baseada em Conteúdo, na qual os dados serão tratados como texto (dados não estruturados). Vejamos a seguir o nosso principal objetivo dentro dessa área e seus respectivos objetivos específicos construídos para alcançar as metas desejadas.

1.2.1 Objetivo Geral

O objetivo geral desta dissertação é propor uma metodologia e utiliza-la para avaliar Sistemas de Recomendação Baseados em Conteúdo Textual e Filtragem Colaborativa em uma tarefa do tipo "recomendar bons itens". A comparação das abordagens além da precisão da previsão que os recomendadores são capazes de realizar nos motiva a identificar características particulares para cada abordagem. Em um contexto onde a esparsidade de dados entre os usuários e a avaliação dos itens é muito alta esperamos desempenhos diferente entre os recomendadores, assim como em aplicações diferentes, como recomendar um filme cinematográfico ou um artigo científico, também pode imprimir resultados diferentes para diversas dimensões dos recomendadores que serão estudados.

1.2.2 Objetivos Específicos

Os objetivos específicos são uma sequência de passos lógicos que devemos executar para alcançar o objetivo geral deste trabalho. Após a revisão da literatura e buscas na web por conteúdo relacionado aos Sistemas de Recomendação chegamos aos seguintes objetivos específicos:

- O primeiro objetivo específico de nosso trabalho é propor uma metodologia de montagem de conjuntos de dados para execução e avaliação da tarefa de recomendação. Não existem conjuntos de dados que possuam conteúdo textual relevante para serem explorados por nossa pesquisa. A grande popularização das técnicas sociais pode justificar a quase ausência desses conjuntos de dados e montá-los será um desafio particular. Outro aspecto importante deste objetivo é a possibilidade de analisar os recomendadores em diferentes contextos. Será necessário montar conjuntos de dados de diferentes sistemas para testar os recomendadores e buscar um pouco mais de generalização nos resultados.

- Outro objetivo é desenvolver algoritmos de recomendação para comparação entre as abordagens. Veremos mais adiante que para a abordagem social, nosso *baseline*, o algoritmo de Filtragem Colaborativa baseado em memória e usuário descrito no trabalho de Su & Khoshgoftaar (2009) foi codificado. Para a abordagem baseada em conteúdo vamos desenvolver dois algoritmos, o primeiro, que representa o perfil do usuário por meio dos termos mais representativos contidos no conteúdo dos itens avaliados pelo usuário. O segundo algoritmo calcula similaridade entre cada item do perfil do usuário com os demais disponíveis no sistema. Mais detalhes sobre os algoritmos de Filtragem Colaborativa e Baseado em Conteúdo veremos no Capítulo 3.

- Definir medidas de avaliação é um passo necessário. Em um teste de sistema é possível aplicar das mais diversas medidas oriundas da área de Recuperação de Informação. Entretanto, muitas pesquisas sobre Sistemas de Recomendação se concentram somente na precisão das previsões. Apresentaremos em nossa metodologia de trabalho novas dimensões de avaliações para ampliar a discussão entre o desempenho dos recomendadores sociais e os baseados em filtragem de conteúdo.

- Realizar testes é outro ponto importante de nosso trabalho. A criação de uma plataforma de testes para os recomendadores é necessária para que se depure os algoritmos de recomendação e obtenhamos os resultados comparativos das medidas aplicadas. Algumas iniciativas desta natureza como: *Mahout* [6] e *LensKit* [7] são artefatos interessantes, mas focados ou em tarefas diferentes de recomendação, ou não exploram novas dimensões de comparação entre as técnicas de recomendação. A criação de um *arcabouço* para os testes de sistema[8] foi outra atividade particular desse trabalho e que mereceu grande esforço.

1.2.3 Organização da Dissertação

No próximo capítulo é apresentada uma breve história dos Sistemas de Recomendação e a revisão bibliográfica sobre este tema necessária para compreensão de nossa metodologia experimental apresentada no Capítulo 3. Além disso, vamos expor alguns pontos considerados como deficientes por parte de grande pesquisadores que são atribuídos aos recomendadores Baseados em Conteúdo e em contrapartida apresentar as características positivas deste tipo de abordagem. Os resultados obtidos com a execução de nosso experimento serão apresentados no Capítulo 4, assim como as discussões que envolvem os resultados gerados pelas medidas utilizadas em nossa experimentação. Ainda neste

[6] https://mahout.apache.org/users/recommender/quickstart.html
[7] http://lenskit.grouplens.org/
[8] Detalhes no próximo capítulo.

capítulo construiremos as respostas de nossas questões de pesquisa. No Capítulo 5 é finalizado o trabalho com os indicativos de novas pesquisas e realizando uma discussão geral sobre todo este trabalho de dissertação de mestrado.

Capítulo 2

Sistemas de Recomendação

Os Sistemas de Recomendação aplicam os mais diversos algoritmos e abordagens na tentativa de colaborar com os usuários sobre o problema da *sobrecarga de informações*. Mesmo sendo uma área da Ciência da Computação com destaque nos últimos anos, a história de pesquisa é recente e os primeiros trabalhos foram publicados somente no início da década de 90. Com o interesse comercial em sua aplicação, novas utilidades foram incorporadas a este tipo de sistema e muitos avanços foram alcançados. Começamos este capítulo falando um pouco sobre o início da pesquisa nesta área na Seção 2.1. Em seguida, na Seção 2.2, apresentamos uma definição formal sobre os Sistemas de Recomendação e o cerne deste tipo de sistema: as preferências dos usuários (Seção 2.3). As principais abordagens são discutidas na Seção 2.4 e na Seção 2.5 apontamos para as tarefas mais comuns destinadas a este tipo de sistema. Na Seção 2.6 detalhamos como é feita a avaliação do tipo de tarefa escolhida neste trabalho. Ao final do capítulo apresentamos algumas pesquisas relacionadas ao contexto de nosso trabalho e que serviram como referências para construção de nossa pesquisa na Seção 2.7.

2.1 As Primeiras Pesquisas

Em 1987 um importante trabalho denominado *Intelligent Information-Sharing Systems* por Malone et al. (1987) apontava o problema da sobrecarga de informação gerado pela grande quantidade de e-mails trocados entre colaboradores de uma organização. Este estudo, precursor dos SR, produziu uma primeira caracterização dos sistemas de filtragem que foram divididos em três grupos: filtragem cognitiva, filtragem social e filtragem econômica. A primeira tem por princípio caracterizar o conteúdo das mensagens e as necessidades do usuário. Em seguida são utilizadas, de forma inteligente, essas representações para filtrar a correspondência. A segunda abordagem, social, foi apresentada como complementar à cognitiva e concentra-se, exclusivamente, nas características do indivíduo, suas relações interpessoais e como se organiza em comunidade. A abordagem econômica tem por princípio o $custo \times benef\acute{i}cio$ de um item, onde aspectos como tamanho da mensagem pode representar o custo de um e-mail, e a quantidade de citações o benefício de um artigo científico.

Alguns anos depois Goldberg et al. (1992) propõem uma nova forma de ajudar o usuário com o problema da sobrecarga de informação proveniente do serviço de e-mail. A proposta inicial seria fornecer listas de discussões por onde o usuário receberia o conteúdo. Essas listas seriam criadas

automaticamente inferindo as preferências dos usuários com base no conjunto de e-mails enviados e recebidos. Porém os autores afirmam que esse conjunto de e-mails raramente representa a lista completa de todos os interesses dos usuários. Os autores propõem então a Filtragem Colaborativa (FC) como "as pessoas colaborando mutuamente para realizar a filtragem por meio da gravação de suas reações nos documentos que leem" (Goldberg et al., 1992, p. 1). As reações dos usuários incluem a indicação de que um item foi particularmente interessante ou desinteressante, o que se tornou uma das bases da Filtragem Colaborativa.

Após esta primeira proposta, outros sistemas foram apresentados nos anos seguintes utilizando abordagem semelhante ao trabalho de Goldberg et al. (1992). Dentre eles, estão: *GroupLens* (Resnick et al., 1994), *Fab System* (Balabanović & Shoham, 1997) e *Siteseer* (Rucker & Polanco, 1997), porém somente com o artigo *Recommender systems* de Resnick & Varian (1997) é que o termo "Sistemas de Recomendação" passou a ser reconhecido como um padrão pela comunidade científica. Os pesquisadores sugerem que o termo é mais adequado que "Filtragem Colaborativa" como vinha sendo usado por outros pesquisadores. E por duas razões: normalmente os usuários são desconhecidos uns dos outros, logo não é explícita a colaboração. Segundo, esses sistemas podem ir além da filtragem de informação: eles podem sugerir itens de interesse dos usuários e surpreendê-los. Além disso, é possível adicionar uma terceira proposição a esta definição: Filtragem Colaborativa é apenas uma das abordagem possíveis para os algoritmos de recomendação. Como veremos ainda neste capítulo é possível construir recomendadores de diversos métodos.

Não só mensagens de e-mail como outros tipos de itens textuais e não textuais se tornaram alvo de pesquisas em SR, principalmente quando a indústria do *Comércio Eletrônico* percebeu que essas técnicas podiam produzir *Customização em Massa* (Silveira et al., 2001). Assim itens como CDs, filmes, livros, artigos científicos e notícias passaram a ser explorados neste tipo de aplicação. Hoje os SR são usados para sugerir qualquer tipo de item, desde um simples produto a ser comprado, até quem será seu próximo amigo em uma rede social. Nestes Sistemas de Recomendação a abordagem mais utilizada é a Filtragem Colaborativa. E, embora itens de conteúdo textual estejam presentes desde a origem desses sistemas, o conteúdo de tais itens ainda é pouco utilizado para gerar recomendações, uma vez que a FC não faz uso desta informação. Mas os Sistemas de Recomendação Baseados em Conteúdo Textual podem explorar tal recurso.

2.2 Definição

Um Sistema de Recomendação é um sistema de filtragem de informação que oferece ao usuário sugestões de itens de seu possível interesse (Ricci et al., 2011). Segundo Adomavicius & Tuzhilin (2005) as áreas bases dos Sistemas de Recomendação são as Ciências Cognitivas, Teoria da Aproximação, Recuperação de Informação, Teoria da Previsão, Ciência da Gestão e Marketing ao Consumidor. Entretanto, surgiu como área de pesquisa independente quando as pesquisas passaram a se concentrar nas avaliações dos usuários sobre os itens para realizar a tarefa de recomendação desejada. Ainda segundo Adomavicius & Tuzhilin (2005) o problema de recomendação pode ser reduzido ao problema da estimativa de classificação para os itens que ainda não foram visualizados pelo usuário.

Os recomendadores, independentemente das técnicas utilizadas, baseiam-se no princípio da personalização da recuperação de informação de modo a agregar valor na experiência do usuário na web. Este princípio da personalização tem forte dependência em como modela e identifica as preferências

do usuário, pois é deste modelo que se infere as recomendações.

2.3 Preferência do Usuário

Identificar com precisão os interesses do usuário é fundamental para um recomendador (Balabanović & Shoham, 1997). O histórico de uso do sistema pelo usuário é a principal fonte para identificação de suas preferências que podem ser obtidas explicita ou implicitamente. A abordagem explícita obtem diretamente dos usuários avaliações sobre os itens ou informações úteis sobre suas preferências e pode apresentar informações de melhor qualidade sobre o usuário (Claypool et al., 2001). No site da *Amazon*, por exemplo, é solicitado do usuário pontuações (*rating*) de 1 até 5 sobre os itens adquiridos ou a qualquer outro item da loja. Essas pontuações podem também possuir intervalos diferentes como: péssimo, ruim, bom, muito bom e excelente. Ou mesmo utilizar o método binário, onde o usuário pode somente gostar ou não gostar de algo (Ricci et al., 2011). É possível também obter comentários textuais ou mesmo solicitar informações extras dos usuários na tentativa de mapear quais seus principais interesses.

Na abordagem implícita há a tentativa de inferir, indiretamente, quais são as preferências do usuário. É possível analisar diversos comportamentos no uso do sistema, tais como: tempo de permanência na página web e quantidade de visualizações de um documento. Por exemplo, existem web sites onde não há avaliações explícitas para os itens, mas podemos inferir as preferências dos usuários por seu histórico de uso do sistema. Em plataformas como CiteULike[1] e Bibsonomy[2] os usuários adicionam em suas contas apontadores para conteúdos na web ou artigos científicos e desta forma pode-se inferir que estes itens são de seu interesse. Essa abordagem, diferentemente das abordagens explícitas, não necessita do apoio do usuário e pode oferecer um grande volume de dados. Porém, apresenta um grau de dificuldade maior na interpretação dos dados e pode conter muito ruído (Joachims et al., 2007). Essa abordagem tem despertado o interesse de diversos pesquisadores (Joachims et al., 2007; Agichtein et al., 2006b; Nichols, 1997; Shahabi & Chen, 2003; Oard & Kim, 1998) e pode substituir técnicas explicitas, ou mesmo complementá-las em soluções híbridas (Nichols, 1997).

As dificuldades em construir um modelo de usuário com dados confiáveis fez surgir nos últimos anos novas técnicas para estimular o usuário em fornecer suas preferências. *Gamification* (Deterding et al., 2011) é um bom exemplo disto. Esta técnica apresenta resultados eficientes na busca das preferências dos usuários oferecendo recompensas para as avaliações sobre os itens (Hacker & von Ahn, 2009). O uso de elementos de jogos eletrônicos enriquece a experiência do usuário gerando assim estímulo para as colaborações corretas. Outra forma de compreender melhor os interesses dos usuários é analisando suas reações em comentários textuais. Entender se sua avaliação foi especialmente positiva ou negativa para determinado item, pode colaborar na construção do modelo do usuário. Deste modo pesquisas sobre *sumarização de sentimentos* podem oferecer elementos para a construção dos perfis de modo indireto (Lerman et al., 2009).

Todas as técnicas, implícitas ou explícitas, visam fundamentalmente obter informações sobre o usuário de modo a modelá-lo. Entretanto, essa etapa, fundamental para um Sistema de Recomendação, é dependente de qual abordagem será utilizada para realizar a tarefa de recomendação planejada.

[1] http://www.citeulike.org/
[2] http://www.bibsonomy.org/

2.4 Principais Abordagens

Os Sistemas de Recomendação podem ser construídos por meio das mais diversas abordagens. A decisão de qual abordagem deve ser utilizada em um algoritmo de recomendação é uma decisão de projeto do sistema que pode ser influenciada pelos mais diversos fatores. Tipos de dados sobre os itens que estejam disponíveis, informações sobre os usuários e suas preferências, quantidade de usuários e quantidade de itens, além da esparsidade da relação $usuário \times item$ são alguns dos fatores que devem ser analisados para decidir qual tipo de abordagem será utilizada na solução de recomendação adotada no sistema.

2.4.1 Abordagem Baseada em Conteúdo

Os recomendadores baseados em conteúdo utilizam um princípio para realizar sua tarefa: analisar o conteúdo dos itens que foram avaliados pelo usuário e os demais itens do sistema para realizar as recomendações (Pazzani, 1999). Os métodos de recomendação Baseados em Conteúdo têm origem nas áreas de RI e AM. Vejamos cada etapa executada por um recomendador que baseia sua tarefa no cálculo de similaridade entre os atributos dos itens do sistema.

Representação do Conteúdo

Nas recomendações por meio de conteúdo é necessário, primeiro, construir uma representação para o conteúdo. Os dados podem ser estruturados, como data da notícia, categoria do produto ou gênero do filme. Entretanto, é comum a presença do formato semiestruturado dos dados textuais (sinopse, descrição, título...) o que implica em estabelecer uma nova forma de representar os itens. Este problema é amplamente estudado em RI (Manning et al., 2008), mas é fundamental na construção de um recomendador Baseado em Conteúdo (BC). A representação mais utilizada para textos (ou mesmo quando utilizado diversos dados estruturados) é feita por meio de termos/palavras. Essa abordagem conhecida como *Bag-of-words* (BOW) representa os documentos como vetores de palavras verificando a frequência em cada documento. Outras abordagens que podem ser utilizadas para representação de textos incluem: *Phrases*, *Bag-of-words with n-gram*, *Bag-of-words and word position*, *Concept categories* e outros (Kosala & Blockeel, 2000).

O processo de construção de qualquer das representações pode ser dividido em etapas, a primeira denominada *extração*, que decompõe o texto em elementos correspondentes aos atributos definidos pela representação escolhida. Essa etapa de pré-processamento tem forte dependência do idioma que foi escrito o texto e normalmente filtros como *stopword*[3] e *stemming*[4] são aplicados (Wang & Wang, 2005). Na segunda etapa – *seleção* – os atributos extraídos são mapeados, normalmente em um vetor de características. Associado a cada atributo se estabelece um peso e alguns dos mais populares são: *Document Frequency* (DF), *Term Frequency* (TF), *Inverse Document Frequency* (IDF) e *Information Gain* (IG) (Sebastiani, 2002). Neste trabalho será utilizado como peso de cada termo $TF \times IDF$ e normalmente essa equação é definida de diferentes maneiras na tentativa de normalização dos valores

[3] O filtro de stopwords, para a língua portuguesa por exemplo, envolve remover artigos, preposições ou qualquer outro elemento que não possua representatividade semântica no domínio do problema

[4] Stemming envolve a redução de palavras às suas raízes, como por exemplo associar as palavras computador, computação e computado como a mesma raiz: computa

de TF e IDF (Manning et al., 2008; Rajaraman & Ullman, 2012). Abaixo, é definido TF e IDF como utilizado ao longo deste trabalho (Alias-i, 2011):

$$TF_{t,d} = \sqrt{count(t,d)} \tag{2.1}$$

onde t é o termo que se busca calcular a frequência no documento d e $count$ a função que realiza a contagem do termo t no documento d. O cálculo é feito contando as ocorrências do termo no documento inspecionado.

$$IDF_{t,docs} = \sqrt{log\left(\frac{n}{df_{t,docs}}\right)} \tag{2.2}$$

Em *Inverse Document Frequency* t é o termo que se busca calcular a frequência inversa no conjunto de documentos $docs$. Para realizar esse cálculo é necessário definir o conjunto de termos presentes em todos os documentos. A variável n define a quantidade de documentos da coleção e $df_{t,docs}$ é a quantidade de documentos em que o termo t está presente.

E finalmente a última etapa de construção de representação de documentos que é a *redução* de dimensionalidade, pois documentos textuais podem gerar representações com milhares de dimensões[5]. Desenvolver uma forma de reduzir a dimensionalidade, mantendo a representatividade dos documentos é um dos desafios para um recomendador Baseado em Conteúdo se manter escalável.

Outras formas de representações de documentos são estudadas na tentativa de criar maneiras mais eficientes de representar um texto. Neste cenário propostas como representação baseada em grafo de Jin & Srihari (2007) pode tornar-se uma alternativa. É possível também utilizar ontologias para aprimorar as representações e a *WordNet*[6] é bastante utilizada em tarefas como Desambiguação Lexical de Sentido (DLS) (Agirre & Edmonds, 2006) o que pode enriquecer o modelo criado. Outro método que vem sendo estudado é a associação de conhecimento extraído do texto em fusão com conhecimento obtido em enciclopédias de senso comum. Esse tipo de técnica pode estender a representação do texto, pois é possível adicionar outros conceitos associados da enciclopédia aos extraídos do texto. Repositórios como *DMOZ - Open Directory Project*[7] e *Wikipedia*[8] são alguns dos disponíveis livremente para uso. Outras representações que exploram relações semânticas entre termos, como a *Latent Semantic Indexing* (LSI) (Berry et al., 1995) também foram propostas, porém, de modo muito tímido esses avanços são explorados por recomendadores Baseados em Conteúdo.

Há uma grande quantidade de representações para o conteúdo textual que foram amplamente pesquisadas em trabalhos científicos sobre Recuperação de Informação. Nos Sistemas de Recomendação a representação mais comum para o conteúdo textual dos itens é dada por BOW (Mooney & Roy, 2000; Mak et al., 2003; Vanetti et al., 2011).

Descoberta dos Interesses dos Usuários

Utilizando sua história de uso do sistema, normalmente, o usuário é modelado por meio de dois conjuntos de itens: os interessantes ao usuário ($d+$) e os desinteressantes ($d-$). É por meio dessas

[5]Ver mais detalhes sobre representações de documentos e redução de dimensionalidade em Sebastiani (2002)
[6]http://wordnet.princeton.edu/
[7]http://www.dmoz.org/
[8]http://www.wikipedia.org/

duas listas que se inicia o processo de "conhecer" os interesses do usuário. Neste momento, o sistema deve identificar a qual conjunto pertence ($d+$ ou $d-$) os itens ainda não avaliados pelo usuário. Os dois modos mais tradicionais de realizar a identificação de qual conjunto pertence um item desconhecido pelo usuário são: por meio de algoritmos de Aprendizado de Máquina (classificadores) ou utilizando cálculo de similaridade entre os itens.

Algoritmos de Classificação são usados quando é necessário associar um documento em classes preestabelecidas. Um usuário, a medida que fornece ao sistema mais indícios de suas preferências, possibilita um maior aprimoramento das recomendações baseadas em conteúdo textual. Neste processo de avaliar se um documento será interessante ou não para o usuário, os algoritmos de classificação são responsáveis em predizer a qual classe de interesse pertence o item. Os classificadores que possuem a característica de além de classificar, também definir o grau de relevância de cada documento em relação às preferências do usuário podem ser preferidos. A seguir alguns dos principais classificadores de textos que podem ser aplicados em SR (Aas & Eikvil, 1999; Baharudin et al., 2010).

- *Naïve Bayes* é um classificador probabilístico bastante utilizado para classificação de textos. Seu princípio "ingênuo" é dado por considerar os atributos dos exemplos independentes entre si. Este princípio, apesar de não ser verdadeiro no mundo real, torna-se bastante eficiente no processo de classificação, principalmente para grandes dimensões de atributos (o que é bastante pertinente em textos).
- *Rocchio's algorithm* é um método clássico da RI muito utilizado para filtragem e encaminhamento de documentos. O seu principal apelo de uso em SR é o baixo custo computacional em relação a outros algoritmos.

Os algoritmos citados são apenas algumas das soluções clássicas disponíveis. Outros métodos de classificação como: *K-Nearest Neighbor*, *Centroid based classifier*, *Decision trees*, *Support Vector Machine* e *Neural Network classifier* podem ser avaliados para executar a etapa de descoberta das preferências dos usuários. Trabalhos específicos de *categorização* e *classificação* de textos podem ser inspecionados para identificar soluções mais eficientes para cada cenário[9].

Cálculo de Similaridade entre dois documentos (a e b) utiliza uma representação de conteúdo, normalmente um vetor de termos/palavras, e realiza uma comparação entre os dois conteúdos. Ao final do cálculo de similaridade é possível estabelecer um ranking entre quais itens são mais ou menos similares às preferências dos usuários ($d+$ e $d-$) e utilizar esta informação para realizar a tarefa de recomendação desejada. Há diversas medidas de similaridade como podemos verificar no trabalho de Real & Vargas (1996), entretanto, apresentamos somente duas das mais utilizadas, a primeira denominada *Similaridade Jaccard* é dada por:

$$sim_{Jaccard}(a, b) = \frac{|A \cap B|}{|A \cup B|} \tag{2.3}$$

onde A é o conjunto de termos presentes no documento a e B é o conjunto de termos presentes no documento b. Esta medida aplicada a uma representação BOW produz um cálculo de similaridade

[9]Ver algumas pesquisas sobre *Mineração de Textos* como (Sebastiani, 2002; Berry, 2003; Harish et al., 2010)

onde os termos tem pesos binários, ou seja, há o termo ou não no documento e todos possuem a mesma importância. Agora vejamos a segunda medida denominada *Similaridade Cosseno* definida por:

$$sim_{Cosseno}(a, b) = \frac{A \cdot B}{\|A\| \, \|B\|} \tag{2.4}$$

Essa medida é mais utilizada, em uma representação BOW, com pesos nos termos calculados por sua importância no documento. O método de cálculo dos pesos é normalmente dado pela multiplicação $TF \times IDF$ e deste modo será utilizado ao longo deste trabalho.

Recomendador Baseado em Conteúdo

Após o processo de descoberta dos interesses dos usuários é possível executar a recomendação dos itens. Esta etapa funciona de modo similar a um filtro de informações mantendo sempre atualizada uma lista ordenada de itens potencialmente interessantes ao usuário (Lops et al., 2011). Em um sistema trivial a recomendação pode ser feita verificando a similaridade entre um novo documento visualizado pelo usuário e a lista de documentos fornecida pelo processo de descoberta de seu interesse (similares a $d+$). Essa primeira etapa gera uma lista de itens ordenada pelos valores de similaridade determinados anteriormente. Em seguida o *recomendador* constrói a *lista de documentos recomendados*, tipicamente selecionando uma quantidade K (*tops*) dos itens mais representativos, ou seja, mais similares ao último documento visualizado.

A recomendação baseada em conteúdo textual é sensível às mudanças de interesse dos usuários que podem ocorrer ao longo do tempo. Um usuário que inicialmente se interessa por artigos científicos sobre Recuperação de Informação e Aprendizado de Máquina, pode, futuramente, interessar-se por Sistemas de Recomendação. Assim a cada nova lista de recomendação gerada ou novo item filtrado diretamente pelo usuário, novas avaliações do usuário podem ser inferidas. As novas avaliações atualizam o *perfil do usuário* que é novamente utilizado pelo sistema para descobrir "ainda mais" quais são seus interesses. Este processo interativo pode fornecer, a cada nova rodada, itens mais interessantes e significativos para cada usuário.

Em nossa questão de pesquisa **Q3** discutiremos as consequências de utilizar conteúdo textual. Alguns trabalhos apontam algumas consequências de utilizar o conteúdo textual, normalmente, sem apresentar resultados experimentais que as justifiquem. Deficiências pré-existentes das técnicas de RI e AM, quando aplicadas nos recomendadores, são propagadas gerando baixa precisão da predição das preferências dos usuários como apontam os trabalhos de Shardanand & Maes (1995); Das et al. (2007); Konstan & Riedl (2012). Vejamos alguns dos aspectos positivos e negativos normalmente atribuídos aos Sistemas de Recomendação Baseados em Conteúdo Textual. Este levantamento apresenta que nas limitações descritas existem oportunidades e preocupações que serão analisadas em nosso experimento no Capítulo 3.

Vantagens e Limitações

As técnicas de recomendação baseadas em conteúdo textual são caracterizadas por fornecer aos usuários itens por meio da similaridade que existe entre o conteúdo que caracteriza os itens. Esse princípio gera as principais vantagens dessa técnica que podemos verificar na lista que segue:

- Necessário conhecer somente o próprio usuário, ou seja, não é necessário conhecê-lo em comunidade (Lops et al., 2011). Quando se utiliza conteúdo textual não é necessário identificar quais são os usuários com preferências semelhantes para a realização da tarefa de recomendação desejada.

- Transparência, pois as recomendações não utilizam dados de outros usuários (Lops et al., 2011). Neste ponto as abordagens baseadas em conteúdo contrapõem a FC que, em sua essência, utiliza dados de uso do sistema de um usuário para recomendar a outro usuário.

- Um item novo recém incluído na base de documentos pode ser recomendado, pois não há dependência do registro de reações dos usuários (Lops et al., 2011). Quando um novo item é cadastrado no sistema nenhuma avaliação feita por usuários está disponível, porém por meio da verificação de suas propriedades e similaridade com outros itens, o Sistema de Recomendação é capaz de recomenda-lo.

- Normalmente há mais usuários do que itens em uma aplicação, logo identificar itens similares pode ter custo computacional menor que identificar usuários similares (Konstan & Riedl, 2012). Esta característica está, principalmente, presente em *e-commerce* onde a quantidade de clientes cresce exponencialmente. E consequentemente criar um modelo de usuário capaz de gerar representações válidas para a grande variedade de indivíduos eleva o custo computacional da tarefa de recomendação.

Em contrapartida, as mesmas propriedades que atribuem vantagens a estes recomendadores geram limitações que precisam ser investigadas, vejamos algumas:

- Não surpreender o usuário, pois as técnicas existentes buscam outros itens similares (Shardanand & Maes, 1995). Esta abordagem não é capaz de fornecer itens que não estejam de acordo com o conteúdo dos itens de interesse do usuário. Ainda segundo Goldberg et al. (1992) uma lista de documentos avaliados dificilmente é capaz de representar todas as preferências do usuário.

- Dificuldades em filtrar com base em qualidade, estilo ou mesmo ponto de vista do texto (Shardanand & Maes, 1995). Os algoritmos de AM utilizados para predizer quais itens estão de acordo com as preferências do usuário ainda não são capazes de "aprender" aspectos mais sutis que extrapolam a similaridade textual. Guerra nas Estrelas e Jornada nas Estrelas são similares[10]?

- Nem sempre termos/tópicos podem ser utilizados para vincular um texto ao domínio de outro texto (Das et al., 2007). Há questões de escrita ou de estilo autoral que podem fazer dois textos do mesmo domínio serem classificados diferentemente: Pelé=Rei do Futebol?

- As altas dimensionalidades geradas nas representações de conteúdo textual podem inviabilizar aplicações comerciais de SR baseados em conteúdo textual (Konstan & Riedl, 2012).

[10]Normalmente há um antagonismo entre os fãs das duas séries, mas isso não seria percebido por um sistema baseado em conteúdo textual.

Os primeiros trabalhos de SR baseados em Filtragem Colaborativa visavam justamente combater algumas das principais limitações da abordagem baseada em conteúdo. O trabalho de Goldberg et al. (1992) afirma que utilizar somente as representações dos documentos de seu perfil, como mecanismo de filtragem, não oferece para o usuário recomendações para a totalidade de seus interesses.

2.4.2 Abordagem Baseada em Filtragem Colaborativa

Diferentemente das técnicas que se baseiam em similaridade entre itens para recomendar, a Filtragem Colaborativa utiliza outro caminho para realização das recomendações. Seu princípio básico é identificar usuários semelhantes, ou seja, que tenham avaliado uma quantidade relevante de itens de modo semelhante e desta forma apresentar uns aos outros os itens distintos bem avaliados por cada um. Essa técnica chamada de "automatização do boca-a-boca" por Shardanand & Maes (1995) tem como princípio uma filtragem onde o trabalho de identificação do conteúdo relevante (ou de qualidade sob a ótica individual do usuário) pode ser realizado de modo particionado entre os utilizadores do sistema quando cada um registra sua avaliação a um item recuperado. Os algoritmos para Filtragem Colaborativa são os mais populares em aplicações de Sistemas de Recomendação (Bogers, 2009) e são divididos em duas categorias: baseado em memória e baseado em modelo.

Algoritmos Baseados em Memória

Estes algoritmos são assim denominados por sua característica de manter em memória os interesses do usuário para então realizar o cálculo de predição da avaliação do usuário para um item. Por conta desse comportamento reagente às interações do usuário no sistema estes algoritmos são também conhecidos como preguiçosos (*lazy*). Os algoritmos baseados em memória são, normalmente, adaptações do algoritmo *k-Nearest Neighbor* (kNN) que podem predizer uma avaliação por meio de duas abordagens: baseado em usuário ou baseado em item.

Baseados em Usuário Quando o algoritmo é baseado em usuário o método de predição de uma avaliação de um item por um usuário é iniciado, primeiro, identificando os vizinhos do usuário ativo[11]. Os métodos mais utilizados para calcular a similaridade entre dois usuários e, consequentemente, descobrir sua vizinhança são o *Coeficiente de Correlação de Pearson* (Rodgers & Nicewander, 1988) e *Similaridade Cosseno* (Berry et al., 1999). O primeiro método é comumente aplicado quando as avaliações dos usuários estão dispostas em uma escala numérica discreta (Su & Khoshgoftaar, 2009) normalmente entre 1 até 5 ou 1 até 10. Em web sites como o Netflix, MovieLens[12] e Amazon o usuário é convidado a avaliar os filmes ou produtos utilizando uma escala de estrelas (5 estrelas) que podem representar valores entre ruim e excelente. O segundo método, *Similaridade Cosseno*, é mais comumente utilizado para cálculo de similaridade entre dois usuários quando avaliações são do tipo binário (gostei ou não gostei) (Su & Khoshgoftaar, 2009). Neste caso, as preferências podem ser representadas como um modelo vetorial de valores 0 ou 1 para representar uma avaliação negativa ou positiva para um determinado item do sistema.

Identificados os vizinhos a próxima etapa é analisar as avaliações dos vizinhos para os itens desconhecidos ao usuário. Os k-vizinhos mais próximos determinam uma previsão da avaliação do item

[11] Usuário para quem se deseja realizar a predição.
[12] http://www.movielens.umn.edu/

desconhecido para o usuário ativo, normalmente, por meio da média entre as avaliações realizadas pelos vizinhos. Uma descrição detalhada desse tipo de algoritmo pode ser verificada no trabalho de Su & Khoshgoftaar (2009).

Baseados em Item Esse algoritmo foi proposto por Sarwar et al. (2001) e popularizou-se, principalmente, em lojas virtuais depois de sua adoção pela *Amazon.com* em 2003. Seu princípio é, selecionado um item, identificar seus vizinhos (outros itens). Os itens possuem em comum as avaliações dadas pelos usuários e por meio dessa informação calcula-se a similaridade entre eles. Novamente, o cálculo pode ser realizado por alguma medida de similaridade como *Cosseno*, por exemplo. Identificados os vizinhos uma aplicação comumente utilizada é a apresentação destes vizinhos, ordenados pelos grau de similaridade, quando um usuário acessa um determinado item. Normalmente, essas recomendações são visualizadas em uma loja virtual como: "quem se interessou por esse item também se interessou por estes outros itens".

Algoritmos Baseados em Modelo

Estes algoritmos têm como característica a criação de um modelo preditivo baseado nas avaliações dos usuários como fase preliminar às recomendações. O uso de algoritmos de Aprendizado de Máquina para construção desse modelo torna a etapa de treinamento mais custosa. Por conta desse comportamento estes algoritmos são também conhecidos como “impacientes” (*eager*). Entretanto, como o modelo é construído com antecedência o custo é compensado na tarefa de recomendação empregada. Apesar de menos popular que os algoritmos baseados em memória, há trabalhos importantes que utilizam os mais diversos algoritmos de Mineração de Dados como Classificadores, Agrupadores e Regras de Associação. Como pioneiro podemos citar o trabalho de Breese et al. (1998) que utiliza o classificador *Naïve Bayes*. Outro trabalho importante foi o de Koren (2008) que utiliza *Latent Factor Models* para reduzir a esparsidade da relação $usuário \times item$ quanto as avaliações. Vucetic & Obradovic (2005) apresentam uma proposta de algoritmo que utiliza regressão para identificar semelhança entre os itens. Em seguida, constrói um modelo que é utilizado por um algoritmo de FC para fornecer previsões de classificação para um usuário. Sarwar et al. (2000) utilizam Regras de Associação que são utilizadas por um algoritmo do tipo *TopK* derivando um algoritmo baseado em item.

Vantagens e Limitações

A grande popularização dos algoritmos de recomendação utilizando a abordagem de Filtragem Colaborativa se deve a algumas vantagens apresentadas em relação à recomendação que utiliza o conteúdo dos itens. A característica que confere a esta abordagem essas vantagens é o fato dela não depender de análise de conteúdo e dos problemas relacionados aos algoritmos de Mineração de Dados. Nessas vantagens podemos incluir a capacidade de filtrar qualquer tipo de conteúdo como livros, notícias, artigos científicos, músicas ou filmes sem a preocupação com questões como idioma e sentido ambíguo das palavras. Além disso, a filtragem é feita considerando conceitos complexos de serem representados por técnicas de processamento de textos como qualidade e estilo. E finalmente, é possível recomendar itens para os usuários que não possuam características semelhantes aos itens de seu perfil e possivelmente surpreendê-los (Herlocker et al., 2000).

Em contrapartida, existem muitos desafios pertinentes à Filtragem Colaborativa que são apontados em muitas pesquisas e foram catalogados no trabalho de Su & Khoshgoftaar (2009). A primeira questão é como lidar com a esparsidade da relação $usuário \times item$ que quando presente pode prejudicar a identificação dos vizinhos ou mesmo a qualidade dessa "vizinhaça". Em decorrência da esparsidade outro problema pode surgir: o problema da partida a frio (*cold-start problem*). Este problema possui duas categorias, a primeira, o problema de novo usuário (*new user problem*) e consiste em como recomendar para usuários que possuem poucas ou nenhuma avaliação sobre os itens. Obviamente, para estes usuários é difícil identificar usuários semelhantes que possam contribuir no processo de filtragem de conteúdo e, consequentemente, realizar a tarefa de recomendação desejada. A segunda categoria, o problema de novo item (*new item problem*), é referente a um novo item disponível no sistema que não possua avaliações dos usuários. Como a técnica social se baseia nas avaliações dos usuários um item que não possua avaliações não pode ser recuperado.

Outros problemas também apontados por Konstan & Riedl (2012) e Su & Khoshgoftaar (2009) são o problema da escalabilidade do sistema em decorrência da característica de muitos deles possuirem muito mais usuários que itens. Neste caso, manter o sistema escalável mesmo com a crescente quantidade de usuários é um grande desafio para algoritmos de Filtragem Colaborativa. É possível destacar também o problema conhecido como *Grey Sheep* que consiste nos usuários que concordam ou descordam totalmente com um conjunto de usuários e em consequência não se beneficiam de recomendações sociais. Há também usuários que possuem preferências bastante específicas o que os fazem pouco semelhantes aos demais usuários (*Black Sheep*). Esta característica torna a tarefa de recomendação direcionada a estes usuários bastante difícil para os algoritmos de recomendação baseados em Filtragem Colaborativa.

2.4.3 Abordagem Híbrida

Há um aspecto complementar para as duas abordagens principais de recomendação, pois justamente onde uma técnica limita-se a outra avança transformando este fato em uma oportunidade. Em Burke (2002) e Liu et al. (2010) são apresentadas aplicações que visam utilizar as duas abordagens de modo a complementá-las. Este tipo de sistema é denominado híbrido e pode, por exemplo, eliminar o problema de partida a frio da FC recomendando um item similar a outro visualizado, assim como surpreender o usuário apresentando itens de interesse de outro usuário com gosto similar. Em um trabalho posterior Burke (2007) apresenta uma taxonomia dos SR Híbridos, estes são comparados e apresentado alguns cenários de aplicação para os mesmos.

2.4.4 Abordagem Baseada em Contexto

Ao longo dos anos diversos trabalhos surgiram na tentativa de aprimorar os Sistemas de Recomendação, alguns utilizando novos algoritmos (Das et al., 2007) ou adicionando novos elementos ao recomendador. Muitas pesquisas como Adomavicius & Tuzhilin (2011), Anand & Mobasher (2007), Adomavicius et al. (2005), Lee et al. (2002) apontam uma outra abordagem denominada Baseada em Contexto. Nestes trabalhos, além de dados sobre os itens ou sobre os usuários, são utilizadas outras informações que agregam valor ao processo de filtragem. Em uma loja virtual que comercializa variedades é importante identificar no perfil do usuário preferências de curto prazo (um pai que compra um artigo para seu filho recém nascido) e as preferências de longo prazo (para um atleta oferecer artigos

esportivos). Além disso, informações sobre localização geográfica, orientação religiosa ou política dos usuários podem ajudar ao recomendador incrementar a qualidade das recomendações.

2.5 Tarefa de Recomendação

Aproveitando a capacidade dos Sistemas de Recomendação em ajudar o usuário no problema da *sobrecarga de informações* sua aplicabilidade tem ultrapassado os cenários de tipos diferentes de itens como: filmes e artigos científicos. É possível utilizar estes sistemas em diferentes cenários e de acordo com Herlocker et al. (2004) para avaliá-los é necessário compreender quais objetivos e tarefas eles realizam. Ainda segundo os pesquisadores este tipo de sistema possui 6 categorias de tarefas de recomendação. Veremos a seguir as principais tarefas.

- Anotação em contexto - essa categoria são de sistemas que apoiam o usuário na tomada de decisão baseado no contexto. Sugerir outros destinatários em uma mensagem de correio eletrônico, baseando-se na presença de outros destinatários, ou mesmo no assunto, é um exemplo de anotações em contextos utilizados em programas de e-mail.

- Encontre bons itens - neste cenário, basicamente o recomendador fornece ao usuário uma lista de itens ordenada baseada em seu interesse e limitada a uma quantidade K de itens. Na literatura costuma-se denominar esses algoritmos de recomendador do tipo *TopK*.

- Encontre todos os bons itens - é um caso especial para *encontre bons itens* onde todos os itens recuperados são apresentados ao usuário na ordem de relevância.

- Recomendação em sequência - alguns sistemas precisam prever uma sequência de itens que melhore a experiência do usuário. Para um pesquisador que deseja iniciar uma pesquisa em uma área o sistema poderia sugerir uma sequência de artigos a serem lidos que forneçam o embasamento teórico necessário para seu objetivo.

- Somente navegando - ao acessar um sistema de compartilhamento de vídeos[13] pode ser interessante ao usuário receber recomendações sem que haja um motivo em particular, neste caso a qualidade da apresentação da recomendação é mais importante que a qualidade da mesma.

- Encontre a recomendação confiável - usuários podem realizar ataques aos sistemas de recomendação alterando suas preferências simplesmente para identificar seu grau de confiança. Neste caso o usuário não está interessado nas recomendações, mas sim em testar o quão robusto é o sistema. Realizar recomendações para esses usuários que efetivamente apresentem itens relevantes é uma tarefa difícil (ver trabalhos como Lam & Riedl (2004), Chirita et al. (2005), Mobasher et al. (2007) sobre ataques em SR).

Para tarefas diferentes em que os recomendadores estão atuando diferentes maneiras de avaliá-los são necessárias. Este trabalho terá foco em uma tarefa em especial: encontrar bons itens. Nesta caso não estamos interessados em prever uma avaliação para um item ou anotá-lo em um contexto, mas sim disponibilizar uma lista de itens úteis ao usuário. Em uma loja virtual, por exemplo, esta

[13] e.g. o http://www.youtube.com/

tarefa seria oferecer uma lista com itens que possivelmente o usuário poderia colocar no carrinho de compras.

2.6 Experimentação

Os Sistemas de Recomendação devem se adaptar ao tipo de ambiente em que vão executar sua tarefa. Em nosso trabalho o recomendador deve identificar as preferências dos usuários e localizar novos itens presentes no sistema que possam ter relevância para o usuário em questão. Um projeto para construir um recomendador deve conter uma etapa para experimentação do sistema. Este passo visa mapear o comportamento do sistema na execução de sua tarefa, compará-lo com outras soluções (caso existam) ou mesmo encontrar a melhor configuração de seus parâmetros antes de entrar na fase de operação. A experimentação tem como objetivo a avaliação do recomendador e para isso devemos considerar duas categorias de avaliação: a avaliação do sistema (*offline*) e a avaliação do usuário (*online*) (Shani & Gunawardana, 2011; Voorhees, 2002).

2.6.1 Avaliação do Usuário

A avaliação do usuário consiste em disponibilizar o sistema (ou um protótipo) a um conjunto de usuários para operá-lo. Neste tipo de avaliação é comum adicionar ao sistema questões a serem respondidas antes, durante e depois do recomendador realizar sua tarefa. Normalmente, o conjunto desses usuários de teste é limitado a um pequeno número, logo questões qualitativas são mais pertinentes de serem aplicadas sobre algumas dimensões do recomendador como: interface com o usuário, qualidade, diversidade e utilidade da recomendação, confiança no sistema ou mesmo sua capacidade em surpreender o usuário.

A avaliação do usuário apresenta um custo maior no projeto do recomendador, pois depende da construção de artefatos de software adicionais e da presença do usuário. Considerado um dos primeiros trabalhos a desenvolver uma avaliação focada no usuário, *Beyond algorithms: An HCI perspective on recommender systems* de Swearingen & Sinha (2001) apresenta resultados interessantes sobre a influência de alguns aspectos da interface do usuário, como a forma de apresentação das recomendações. Porém, é um trabalho passível de crítica por utilizar não mais que 19 usuários na avaliação podendo tornar os resultados pouco confiáveis (Massa & Bhattacharjee, 2004).

Trabalhos posteriores a Swearingen & Sinha (2001) passaram a apresentar novos protocolos para testes *online*. Uma avaliação focada no usuário pode ser construída direcionando uma parte do fluxo de uso do sistema para soluções diferentes de recomendadores (Kohavi et al., 2009) e desta forma utiliza os dados de navegação em cada solução para comparar os algoritmos. No *15th ECML PKDD Discovery Challenge*, que ocorreu em 2013, houve uma etapa *online* de avaliação dos sistemas de recomendação de nomes próprios propostos por cada equipe participante do desafio[14]. Neste cada participante do desafio, que consistia em um recomendador de listas de nomes próprios, era escolhido randomicamente pelo sistema para apresentar suas recomendações aos usuários. Desde modo cada solução foi convidada a apresentar sua utilidade que foram comparadas pelo volume de cliques gerados por cada sistema.

[14]http://www.kde.cs.uni-kassel.de/ws/dc13/online/

2.6.2 Avaliação do Sistema

A avaliação do sistema é um protocolo de teste *offline* e, por isso, não depende da interação com usuários. Normalmente é um tipo de avaliação mais fácil de ser desenvolvida e pode fornecer dados para uma análise não só qualitativa, mas também quantitativa de algumas dimensões de um recomendador. Em uma avaliação *offline* devemos seguir três diretrizes básicas no experimento (Shani & Gunawardana, 2011):

Levantar as hipóteses Um experimento visa comprovar alguma hipótese ou ideia, logo antes de configurar o experimento é necessário levantar as hipóteses que se deseja comprovar. Neste trabalho, como vimos na Seção 1.1, pretende-se responder quais as diferenças entre um recomendador baseado em conteúdo e o social. Nossa hipótese é que estes sistemas possuem coberturas diferentes entre os usuários do sistema. Neste caso é possível que cada abordagem apresente melhores resultados para diferentes tipos de usuários e para isso será necessário um experimento que compare os resultados das recomendações.

Controlar as variáveis Ao realizar a comparação entre recomendadores para comprovação de alguma hipótese é necessário determinar as variáveis envolvidas e mantê-las fixas durante a experimentação. Em nosso experimento, detalhado no Capítulo 3, vamos comparar recomendadores de abordagens diferentes, porém o desafio de cada um deles ao comparar os resultados deve ser o mesmo. Neste caso variáveis como: conjunto de dados, quantidade e quais itens para predição devem ser as mesmas para cada recomendador.

Buscar a generalização Uma experimentação é uma tentativa de responder questões previamente levantadas (hipóteses) e para tal deve realizar procedimentos que evidenciem as respostas. Quando se constrói um recomendador é prudente testá-lo antes de operá-lo utilizando dados previamente coletados do sistema no qual este vai realizar sua tarefa. Neste caso, deve-se buscar nesta experimentação a maior generalidade possível. Os filtros no conjunto de dados utilizado e a variedade de usuários e itens devem ser exploradas com diferentes configurações de modo a obter resultados o mais gerais possíveis para o sistema proposto.

Definido o tipo de tarefa de recomendação para recomendar bons itens, um método de avaliação *offline* para este cenário é transformar a tarefa de recomendação em uma tarefa de predição de itens utilizando o histórico de avaliações dos usuários. Recuperando a lista de itens considerados interessantes para um usuário retira-se uma porção de itens desse conjunto e o recomendador deve prever, em uma lista, estes itens que foram escondidos do sistema. A lista de itens que são considerados como interessantes para o usuário é composta dos itens do sistema que foram no passado bem avaliados pelo usuário. Desta maneira o recomendador pode ser avaliado como um classificador binário e para isso cada item removido do perfil do usuário deve aparecer nas primeiras K posições da lista de recomendação, caso contrário a tarefa não foi bem executada. Na Tabela 2.1 podemos ver os possíveis resultados desta classificação (Shani & Gunawardana, 2011).

Tab. 2.1: Matriz de confusão para uma classificação binária.

	Recomendado	Não recomendado
Interessante	Verdadeiro-Positivo (#vp)	Falso-Negativo (#fn)
Não interessante	Falso-Positivo (#fp)	Verdadeiro-Negativo (#vn)

Utilizando a matriz de confusão da Tabela 2.1 podemos obter algumas das fórmulas comumente usadas para avaliação da tarefa proposta neste trabalho. Estas fórmulas consideram como *Interessante* os itens que foram removidos do perfil do usuário e o *Não interessante* o restante dos itens do catálogo do sistema.

Precisão É dada pela proporção entre a quantidade de itens recomendados que são do interesse do usuário (avaliados positivamente) e todos os itens que foram recomendados (tamanho total da lista de recomendação). Esta medida é comumente utilizada para avaliação de recomendadores do tipo "bons itens" e é dada por:

$$Precisão = \frac{\#vp}{\#vp + \#fp} \tag{2.5}$$

Segundo Shani & Gunawardana (2011) "em aplicações onde o número de recomendações apresentadas para o usuário é predeterminado, a medida mais útil de interesse é *Precision at K*", também conhecida por *P@K* (Baeza-Yates & Ribeiro-Neto, 1999). Neste caso, considera-se somente os K primeiros documentos da consulta e quando nenhum documento relevante é recuperado até a última posição da lista o resultado da precisão é zero. Em um Sistema de Recomendação do tipo *TopK* normalmente as listas de recomendação são ordenadas de acordo com a preferência dos usuários, neste caso é válido considerar a posição dos itens relevantes na lista e uma medida de precisão para isso é a *Average Precision* (AveP) definida como:

$$AveP = \frac{\sum_{k=1}^{n}(P(k) \times rel(k))}{R} \tag{2.6}$$

onde n é a quantidade de documentos recuperados na lista de recomendação e R é a quantidade de documentos relevantes, neste caso a quantidade de itens removidos do perfil do usuário para que o sistema faça a predição. K é a posição do documento no rank (lista) e a função $P(k)$ é a precisão no ponto de corte k da lista. A função $rel(k)$ indica valor 1 se o documento presente nesta posição é relevante e valor 0 se o documento da posição não for relevante. A média dos valores de *AveP* após todas as consultas é denominada *Mean Average Precision* (MAP). Essa medida é o resumo de valor único mais comumente usado na execução de um conjunto de consultas (Agichtein et al., 2006a) e utilizaremos essa medida como base comparativa entre os recomendadores testados. Em alguns momentos chamaremos essa medida simplesmente de MAP para simplificar a leitura.

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \tag{2.7}$$

onde Q é a quantidade de listas de recomendação geradas pelo sistema (chamadas de consultas em sistemas de RI). Se sempre for considerado um tamanho fixo para as lista de recomendação é possível definir *AveP at K* (AveP@K) e *MAP at K* (MAP@K) que são medidas mais adequadas em aplicações como um recomendador *TopK*.

Revocação (*Recall*) É dada pela proporção entre a quantidade de itens recomendados que são do interesse do usuário e todos os itens de interesse do usuário. Esta medida tem por princípio indicar a relevância dos bons itens recuperados na lista de recomendação e é dada por:

$$Recall = \frac{\#vp}{\#vp + \#fn} \tag{2.8}$$

Caso o tamanho da lista de recomendações também seja predeterminado podemos chamá-la de *Recall at K*, também conhecida por *R@K* (Baeza-Yates & Ribeiro-Neto, 1999). Um *trade off* entre estas medidas $Precisão \times Revocação$ é esperado. Seguramente, aumentar o tamanho da lista de recomendação pode melhorar a *Revocação*, porém é esperada uma queda em *Precisão*. Deve-se verificar que independentemente deste aspecto, um fator a ser considerado nas duas medidas é que o conjunto de dados de testes é restrito à experiência do usuário, portanto incompleto, pois não considera itens relevantes que ainda não foram avaliados pelo usuário. Desta forma há uma superestimação da *Revocação*, já que não é possível recuperar todos os itens relevantes com as abordagens de recomendação atuais, mas somente uma amostra representativa deles (Bellogín et al., 2010).

Uma avaliação para múltiplos usuários poderia calcular *Precisão* e *Revocação* em cada lista de tamanho K para cada usuário e em seguida calcular a média em cada K. O resultado pode ser plotado em um gráfico equivalente à curva ROC (do inglês *Receiver Operating Characteristic*) que é uma forma eficiente de demonstrar a relação, normalmente, antagônica entre *Precisão* e *Revocação*.

Cobertura É uma medida que visa analisar uma dimensão dos recomendadores sob o aspecto de qual percentual dos itens o Sistema de Recomendação é capaz de gerar recomendações ou o percentual dos itens disponíveis que de fato são sempre recomendados para um usuário. Ainda, segundo (Herlocker et al., 2004) *Cobertura* é uma medida do domínio dos itens que o recomendador é capaz de gerar recomendações. Para uma tarefa de recomendação do tipo "recomendar bons itens" uma lista de itens é apresentada ao usuário que, normalmente, é limitada a uma quantidade K dos itens mais relevantes. O *Catalog coverage* (CCOV) é uma medida que verifica a capacidade do recomendador em recuperar todo o conjunto de itens do sistema e definida como (Ge et al., 2010):

$$Catalog\ coverage = \frac{\left|\bigcup_{j=1}^{N} I_L^j\right|}{|I|} \tag{2.9}$$

onde I_L^j é o conjunto de itens recomendados pelo sistema a partir de listas L geradas pelo recomendador em j vezes durante um período de tempo. N denota o número total de recomendações realizadas durante o período analisado e I o conjunto de itens disponíveis no sistema, ou seja, o catálogo.

As medidas de *cobertura* são importantes para avaliar questões como o tratamento dado a novos usuários e/ou itens adicionados ao sistema e que ainda não fizeram muitas avaliações ou foram avaliados (*cold-start problem*). No trabalho de Bogers (2009) *Cobertura do Usuário* foi utilizada sob outra ótica, analisando qual é o percentual entre todos os usuários para os quais o recomendador foi capaz de gerar qualquer previsão. Ou seja, se as abordagens de recomendações testadas eram capazes de ao menos realizar sua tarefa de recuperar itens, independentemente, de sua precisão.

Outras Medidas Há outras medidas que avaliam outras dimensões dos Sistemas de Recomendação e estas auxiliam o pesquisador a perceber aspectos que vão além da precisão dos recomendadores em uma tarefa de predição. Outras medidas como *Confidence*, *Trust*, *Novelty*, *Serendipity*, *Diversity*, *Utility*, *Risk*, *Robustness*, *Privacy*, *Adaptivity*, *Scalability* estão disponíveis nos trabalhos de Shani & Gunawardana (2011) e Herlocker et al. (2004) e podem ser utilizadas para diferentes tarefas de recomendação.

Na avaliação *offline* de sistemas de recomendação, na qual usuários não estão avaliando os itens sugeridos, o algoritmo é forçadamente feito para considerar itens não avaliados como não úteis ao usuário. Ou seja, itens sugeridos que não estejam na lista de itens escondidos do histórico do usuário são avaliados como de não interesse deste. Esta hipótese pode não ser verdadeira, pois os recomendadores surgem como solução para o problema da sobrecarga de informações e o usuário pode não ter avaliado anteriormente um item recomendado justamente por não ter conhecimento de sua existência. Logo, este tipo de avaliação que mede a precisão das previsões é bastante questionada, principalmente no importante trabalho de Herlocker et al. (2004).

Em Shani & Gunawardana (2011), por exemplo, discute-se que os valores encontrados para Falso-Positivo (#fp) sejam demasiadamente estimados pelas medidas oriundas de RI e que em uma possível avaliação *on-line* (onde as recomendações são postas a prova dos usuários) o usuário poderia considerar útil um item recomendado. Em Konstan & Riedl (2012) são discutidos diversos aspectos das medidas e os motivos pelos quais os primeiros Sistemas de Recomendação eram avaliados somente em *back-test*. Ainda, neste artigo é enfatizado principalmente "o descompasso entre as medidas e a necessidade do usuário", por conta dos algoritmos de recomendação serem projetados para atender medidas impróprias para sua tarefa.

Embora criticada, ainda hoje em competições como: *Netflix Prize*[15], *Recommending Given Names*[16] e em experimentos científicos como nos trabalhos de Koren (2008) e Cremonesi et al. (2010) são utilizadas essas medidas para uma primeira avaliação dos algoritmos. Dessa forma as principais medidas de avaliação para experimentos *offline* na tarefa de "recomendar bons itens" são ainda obtidas utilizando as medidas de RI.

2.7 Recomendação por Conteúdo Textual

Nosso trabalho é uma pesquisa que está concentrada na área de Sistemas de Recomendação. De modo mais específico vamos detalhar aspectos da Abordagem Baseada em Conteúdo Textual comparando-a com a Abordagem Social. Nesta perspectiva serão desenvolvidos recomendadores por conteúdo que utilizando métodos de manipulação de textos realizarão a tarefa de "recomendar bons itens" aos usuários por meio de listas de recomendação. Além disso, um recomendador social deve ser construído para realizar Filtragem Colaborativa dos itens do sistemas e sugerir aos usuários itens de seus interesses de modo a ser um *baseline* em nossa pesquisa. Diante dos resultados dos algoritmos de cada abordagem buscaremos respostas sobre quais são as semelhanças, diferenças e como se comportam em diferentes contextos. Neste capítulo apresentamos alguns trabalhos que estão relacionados com nossa pesquisa por dois diferentes aspectos: por utilizarem conteúdo textual dos

[15] http://www.netflixprize.com/
[16] http://www.kde.cs.uni-kassel.de/ws/dc13/

itens para realizar uma determinada tarefa de recomendação e por realizar algum tipo de comparação entre recomendadores por conteúdo e social.

Na literatura analisada existem trabalhos que realizam comparações e avaliações em técnicas Baseadas em Conteúdo Textual. Na publicação *Syskill & Webert: Identifying interesting web sites* de Pazzani et al. (1996) a pesquisa realizada compara a acurácia de diversos algoritmos de Aprendizado de Máquina, mais precisamente classificadores, em uma tarefa de recomendação Baseada em Conteúdo de páginas Web. Dentre os algoritmos utilizados estão: Nearest Neighbor, ID3, Naïve Bayes, Rede Neural Perceptron (com *Backpropagation*). Neste trabalho há uma preocupação em generalizar os algoritmos fazendo-os atuar em diferentes domínios na web como páginas de filmes ou sobre biomedicina. Além disso, uma grande contribuição deste trabalho é verificar que diferentes classificadores apresentam resultados diferentes nos cenários utilizados ou mesmo quando se utiliza quantidades diferentes de exemplos de treinamento (páginas web presentes no sistema). A avaliação do recomendador é outro diferencial desse trabalho. Neste sentido o processo é feito por meio de uma avaliação do usuário no qual o mesmo deve informar para qual lista deve ser adicionada a página apresentada: *Hotlist*, *Luke Warm List* ou *Cold List*. Embora exiba bons resultados apresentados pelo classificador Bayes (e outros) com relação ao *baseline* (*Most Frequent*) a metodologia de avaliação apresenta um problema recorrente de avaliações de usuário: baixa quantidade de usuários para realizar a tarefa.

Em diversos sistemas não há uma maneira explicita para que os usuários informem suas impressões sobre um determinado item do sistema. No trabalho *Webwatcher: A learning apprentice for the world wide web* de Armstrong et al. (1995) o sistema aprende sobre as preferências dos usuários para realizar sua tarefa de recomendação considerando a seguinte hipótese: páginas web visitadas são consideradas como exemplos positivos e páginas web não visitadas são consideradas exemplos negativos. Esta hipótese normalmente é considerada em sistemas que não possuem avaliação explicita dos usuários. Esta hipótese pode ser questionável, mas necessária para a etapa de teste de sistema em diversos contextos. Outra questão relevante sobre o trabalho de Armstrong et al. (1995) é avaliar o recomendador por conteúdo utilizando diferentes técnicas de RI sobre o conteúdo das páginas web. Neste trabalho foram utilizadas as seguintes técnicas para identificar similaridade entre os documentos: Winnow, Wordstat e $TFIDF + Cosseno$. Como *baseline* o experimento utilizou um algoritmo de seleção aleatória dos itens para compor as recomendações. Todas as técnicas de RI foram superiores na avaliação de sistema aplicada em relação ao *baseline*, onde parte das atividades dos usuários foram utilizadas para treinamento e outra parte para teste. Outra preocupação importante deste trabalho é em relação ao *trade-off* que normalmente afeta sistemas de filtragem: $acurácia \times cobertura$. O autor enfatiza os diferentes comportamentos apresentados pelas diferentes técnicas de RI utilizadas.

Na tese de doutorado *Recommender Systems for Social Bookmarking* de Bogers (2009) há um capítulo especialmente dedicado à comparação de recomendadores Baseados em Conteúdo, Filtragem Colaborativa e híbridos. Este trabalho verifica a precisão dos recomendadores utilizando conteúdos diferentes (chamados de metadados) para conjuntos de dados diferentes (Bibsonomy[17] e Delicious[18]), além de dois algoritmos distintos para recomendar por conteúdo extraído do perfil dos usuários. Os resultados comparativos obtidos por Bogers na tarefa de recomendação de itens demonstram ligeira vantagem dos recomendadores por conteúdo em alguns cenários em relação ao melhor resultado de

[17] http://www.bibsonomy.org/
[18] https://delicious.com/

recomendação social obtido. Outro resultado importante é que a combinação das duas principais técnicas apresentou resultados superiores em algumas configurações do experimento sugerindo que a complementação dos resultados de cada uma das abordagens pode constituir em melhores resultados para o usuário.

Além da precisão das previsões, Bogers propõe avaliar *User coverage* que é o percentual de usuários ativos que o algoritmo recomendador é capaz de gerar alguma previsão. Entretanto, a medida de UCOV proposta neste trabalho é booleana, ou seja, considera qualquer quantidade de previsões como positivo (valor total da medida como 1) ou nenhuma previsão como negativo (valor zero para a medida). Desta forma não é possível perceber diferenças expressivas quando a tarefa de recomendar "bons itens" for executada em listas de tamanho fixo. Se em um sistema for desejável uma quantidade $K = 10$ de itens nas listas de recomendação e um algoritmo recupere somente 1 item e outro recuperar 7 itens, estes são avaliados com o mesmo peso pela medida proposta. Essa visão sobre os recomendadores pode se tornar míope em um cenário com muitas ocorrências semelhantes ao exemplo dado comparando-os da mesma forma. Este trabalho tem forte influência em nossa pesquisa, principalmente na abordagem de avaliação da tarefa de recomendação com diferentes conteúdos separadamente. Neste trabalho cada atributo recuperado dos itens do sistema são utilizados individualmente de modo a perceber sua influência no processo de filtragem.

Estes trabalhos citados apresentam uma preocupação em comum: generalizar os algoritmos de recomendação por conteúdo de modo a atuar em diferentes tipos de itens ou que possuam diferentes tipos de conteúdos (filmes, artigos científicos, páginas web sobre biomedicina, dentre outros). Entretanto, é comum a construção de experimentos para atender a somente um tipo de item foco de algum sistema. *Profiling with the INFOrmer text filtering agent* de Sorensen et al. (1997) foi um trabalho publicado sobre o sistema *INFOrmer*, que, como os próprios autores definem, é um agente inteligente de filtragem de conteúdo para artigos de notícias da USENET. Este trabalho define o sistema de filtragem com as seguintes etapas: pré-processamento, onde as técnicas de remoção de *Stopwords*, *Stemming* e *Sentence Boundary Disambiguation* foram utilizadas. Em seguida, é criada uma representação semântica para todos os documentos por meio de uma rede onde os nós são os termos extraídos dos documentos e as ligações suas relações detectadas dentro dos textos (distância entre os termos). Após a criação da nova representação dos documentos um algoritmo de comparação foi criado para recuperar e ordenar novos documentos de acordo com a similaridade ao perfil dos usuários. Um ponto importante desse sistema é o modulo denominado *Relevance Feedback* que solicita ao usuário suas impressões sobre cada artigo apresentado e utilizando desse mecanismo aprende mais sobre as preferências do usuário. O sistema é testado em termos de *Precision* e *Recall* e uma de suas contribuições é detalhar um processo de filtragem capaz de superar problemas recorrente aos recomendadores por conteúdo: sinonímia e polissemia que acabam por atrapalhar a experiência do usuário.

Notícias é um tipo de conteúdo bastante dinâmico e aspectos temporais devem ser considerados em uma recomendação. A relevância de uma notícia pode ser questionada quando tenha sido passado algum período após a ocorrência do fato. Em *Terms of a feather: Content-based news recommendation and discovery using twitter* de Phelan et al. (2011) é proposto um sistema de filtragem baseado em conteúdo, denominado *Buzzer*, que analisa termos extraídos do *Twitter*[19] do usuário para filtrar notícias e intercalar em uma lista de notícias geradas pelo recomendador. O trabalho propõe quatro estratégias para recuperação de notícias relevantes utilizando *tweets* públicos e de amigos combinando

[19]https://twitter.com/

com os artigos RSS pessoais e de comunidades. Além deste trabalho apresentar um recomendador que realiza a fusão de resultados diferentes de filtragem por conteúdo, sua abordagem de avaliação é bastante particular. Para realizar os testes um novo sistema foi desenvolvido para cada uma das estratégias de recomendação e por meio de testes com os usuários seus comportamentos de uso com o sistema foram mapeados. Os resultados obtidos são discutidos por meio da quantidade de cliques gerados por cada uma das abordagens.

Tutoriais é outro tipo de conteúdo dinâmico como notícias. Textos como "faça você mesmo" normalmente é de interesse momentâneo para o usuário. Em *Using content-based filtering for recommendation* de Van Meteren & Van Someren (2000) é proposta uma solução para recomendar esse tipo de item utilizando *Rocchio's algorithm* e aspectos de como manter o perfil de usuário são discutidos. Um ponto particular deste trabalho é sua forma de avaliação do sistema que funciona com a criação de três usuários fictícios, onde o perfil de cada uma deles é construído por meio de itens de categorias diferentes de tutoriais. Uma porção de itens foram recomendados para cada um dos usuários e os mesmos foram testados em termos de *Precision* e *Recall*. Essa forma de avaliação alia teste de sistema com teste de usuário, sendo que o usuário não é real. Além disso a quantidade de perfis simulados é baixa de modo a não cobrir uma quantidade relevante de tipos de usuários com preferências distintas. Nesta pesquisa os autores destacam as influências negativas das técnicas de RI utilizadas, em que, por exemplo, documentos similares são descritos por meio de palavras sintaticamente diferentes. Eles sugerem que isso ocorre, principalmente, pelo curto tamanho dos documentos recuperados e caso fosse possível obter mais informações sobre os itens este problema poderia ser minimizado.

No trabalho de Mooney & Roy (2000) denominado *Content-based book recommending using learning for text categorization* foi construído um recomendador que executa a tarefa de recomendar bons livros para os usuários. O sistema utiliza uma série de atributos recuperados por um *Web Crawler*[20] que minerou o site da *Amazon*. De forma a garantir conteúdo relevante para realizar sua tarefa o sistema restringe sua execução a livros que possuam ao menos uma revisão de usuário ou resumo ou um comentário de usuário. Neste trabalho são descritas as técnicas de RI utilizadas para a construção do recomendador que se concentram em técnicas clássicas como a representação BOW e o classificador *Naïve Bayes*. Uma descoberta interessante dessa pesquisa é sobre os tipos de conteúdo que podem aprimorar as previsões realizadas pelo recomendador: conteúdo colaborativo. Este termo é aplicado para os conteúdos dos livros que foram adicionados pelos usuários do sistema (por exemplo: um comentário). Os resultados apresentados mostram que ao utilizar este tipo de conteúdo a precisão do recomendador aumenta para o conjunto de dados testado.

Diferente de tipos de itens como livros e artigos científicas os Filmes constituem em um tipo bem particular de item que pode apresentar um grau maior de dificuldade para recomendadores por conteúdo. Normalmente, este tipo de item tem pouco conteúdo associado e questões de estilo podem influenciar no processo de filtragem. Utilizar somente atributos como título podem ajudar muito pouco[21] e por isso no sistema *INTIMATE* (Mak et al., 2003) foi adicionado ao vetor de características dos itens a sinopse para melhorar a representação dos filmes. Neste trabalho Mak et al. (2003) apresentou o resultado de testes de sistema (*Accuracy*, *Recall*, *Precision* e *F1-Measure*) para diferentes tipos de representação de conteúdo (BOW, *Nouns* e *Noun Phrases*), diferentes tipos de extração de

[20] *Web Crawler* é um programa de computador que navega na web de modo automatizado para recuperar conteúdo.

[21] Como selecionar novos filmes similares ao filme denominado *Pi* (http://www.imdb.com/title/tt0138704)?

features (DF, IG e MI) e diferentes algoritmos de classificação (*k-Nearest Neighbor*, *Decision Trees* e *Naïve Bayes*). Em todas as variações de configuração testadas para filmes extraídos do *Internet Movie Database* (IMDb)[22] os resultados foram semelhantes.

Discussão sobre as Pesquisas Relacionadas Os Sistemas de Recomendação Baseado em Conteúdo Textual utilizam das mais diversas técnicas de RI como observamos em algumas pesquisas apresentadas neste capítulo. Entretanto, há uma concentração de pesquisas em soluções que utilizam duas técnicas específicas no processo de filtragem: representação dos documentos utilizando representação BOW e aprendizado de novos interesses dos usuários por meio de algoritmos de classificação como o *Naïve Bayes*. Considerando essas pesquisas seria possível concluir que é suficiente o uso das técnicas clássicas para realizar a tarefa de recomendação desejada e que as grandes dificuldades em utilizar este tipo de abordagem são: qual conteúdo utilizar e como obter esse conteúdo. Em muitos sistemas existe pouco conteúdo para ser explorado por estes recomendadores e a exemplo do trabalho de Mooney & Roy (2000) é necessário criar critérios para adicionar um item no conjunto de itens passíveis de recomendação (conjunto a ser filtrado). Mesmo diante das dificuldades as pesquisas listadas nesta seção apresentam bons resultados para a abordagem estudada neste trabalho. Além disso é válido destacar que estas pesquisas não colocam a filtragem Baseada em Conteúdo como antagonista da Filtragem Colaborativa. Inclusive Van Meteren & Van Someren (2000) sugere, ao final de seu trabalho, que o sistema *PRES* seja evoluído para combinar recomendações das duas abordagens de modo a cobrir diferentes perfis de usuários. As pesquisas sobre sistemas híbridos que associam as duas abordagens são exemplos de trabalhos que aprimoram os resultados das recomendações explorando a característica complementar, afirmada por alguns pesquisadores, entre as duas técnicas.

[22] http://www.imdb.com/

Capítulo 3

Construção do Experimento

No Capítulo 2 abordamos o tema Sistemas de Recomendação apresentando as duas principais abordagens, os tipos de tarefas comumente executadas por esses sistemas e estratégias de como avaliá-los. Nosso trabalho visa explorar a abordagem que utiliza o conteúdo dos itens, ou seja, informações que os caracterizam e assim executar a tarefa de recomendar bons itens. Utilizando uma avaliação de sistema buscamos responder questões que possam indicar a utilidade deste tipo de técnica e entender, de modo mais claro, as diferenças desta para a técnica de filtragem social.

Neste trabalho vamos utilizar os dados textuais sobre os itens do sistema manipulando-os como textos e verificando quais desses dados são relevantes para a tarefa proposta. Nossos dados serão obtidos de três sistemas: *MovieLens*, *IMDb* e *CiteULike* como veremos mais detalhes sobre eles na Seção 3.1. Em seguida, na Seção 3.2, vamos apresentar um algoritmo de Filtragem Colaborativa clássico e logo em seguida, na Seção 3.3, dois algoritmos de Recomendação Baseado em Conteúdo Textual, como serão feitas suas configurações e uso durante o experimento. Na Seção 3.4 apresentamos os filtros aplicados nos conjuntos de dados e as configurações finais dos dados utilizados no experimento. Os protocolos de avaliação utilizados em nossa experimentação serão detalhados na Seção 3.5. Finalizando este capítulo de nosso trabalho comparativo vamos apresentar quais medidas serão aplicadas em nosso experimento na Seção 3.6.

3.1 Conjuntos de Dados

Desde o Capítulo 2 apresentamos a decisão de pesquisa sobre o tipo de tarefa a ser executada pelos recomendadores estudados neste trabalho. Ao optar por "recomendar bons itens", consequentemente, começamos a definir nossa metodologia experimental. Uma primeira preocupação neste ponto é obter um conjunto de dados que possibilite análises qualitativas e se possível quantitativas sobre os recomendadores. Na Seção 2.6.2 vimos que um experimento deve buscar a generalização o quanto possível. Em nosso trabalho vamos submeter nossos algoritmos de recomendação a dois diferentes contextos de recomendação: filmes e artigos científicos. Além disso, tipos diferentes de itens possuem dados diferentes a serem tratados por nossas soluções baseadas em conteúdo. Vejamos primeiro como obtemos os dados de filmes e em seguida os dados sobre artigos científicos.

3.1.1 MovieLens e IMDb

O projeto *MovieLens*[1] é uma das iniciativas do grupo de pesquisa *GroupLens*[2] para recomendação de filmes. Este projeto é utilizado como ferramenta de pesquisa para avaliação do usuário de questões relacionadas à Sistemas de Recomendação, como: recomendadores sociais, recomendadores baseados em marcadores (*tagging*) e interface com o usuário.

Quando o usuário faz seu registro no sistema ele recebe primeiro uma lista aleatória de filmes para que o mesmo faça uma avaliação para cada um deles. As avaliações são computadas por meio de uma escala numérica entre 0 e 5 representada por estrelas. Neste caso é possível notas com fração de 0.5 (metade de uma estrela) e quanto mais estrelas possuir melhor é a avaliação dos usuários quanto ao filme. A tarefa de recomendação original do sistema é prever a avaliação de um usuário para um filme antes que o mesmo o assista. Para isso, o sistema compara as avaliações de um usuário com a de outros usuários que tenham realizado avaliações semelhantes para os mesmos filmes. Ao final o sistema identifica as notas dadas pelos usuários semelhantes (vizinhos) e infere a nota que o usuário daria ao assistir o filme. Na Figura 3.1 é apresentada a página inicial do sistema *MovieLens*. Este projeto gerou um dos principais conjunto de dados para pesquisas em Sistemas de Recomendação.

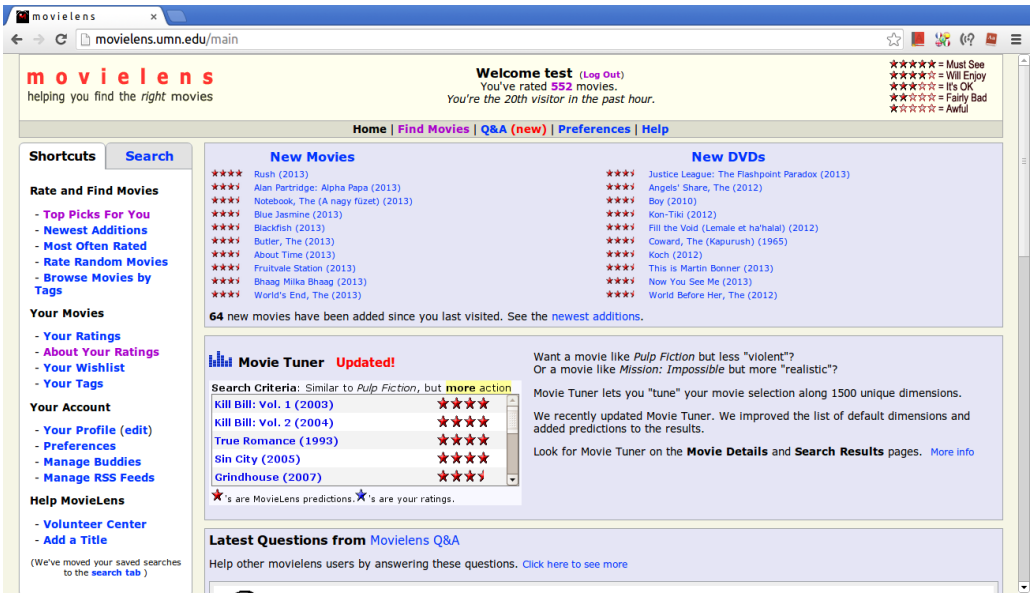

Fig. 3.1: Página inicial do sistema *MovieLens* que realiza a tarefa de recomendação de filmes.

O *GroupLens* disponibiliza para a comunidade três conjuntos de dados[3] para pesquisas em Sistemas de Recomendação. A construção desses conjuntos foi feita utilizando a captura de dados de uso do sistema e posterior filtro. Durante essa pesquisa estavam disponíveis as seguintes configurações para os conjuntos de dados (valores aproximados): *MovieLens 100k Data Set* com 100.000 avaliações de 1.000 usuários em 1.700 filmes, *MovieLens 1m Data Set* com 1 milhão de avaliações de 6.000 usuários em 4.000 filmes e *MovieLens 10m Data Set* com 10 milhões de avaliações e 100.000 marcadores aplicados em 10.000 filmes por 72.000 usuários.

[1] http://www.movielens.umn.edu/
[2] http://www.grouplens.org/projects
[3] Os conjuntos de dados estão disponíveis para download em http://www.grouplens.org/node/73

Propriedade	**MovieLens – 1m**
#usuários	6040
#itens	3883
#atividades	1000209
proporção #itens/#usuários	0,6428
média #itens por usuário	165,5968
média #usuários por item	269,8880
max #itens por usuário	2313
max #usuários por item	3428
min #itens por usuário	20
min #usuários por item	1
esparsidade $usuário \times item$	0.95735

Tab. 3.1: Características do Conjunto de Dados MovieLens com os filtros originais.

Adotamos na nossa pesquisa o *MovieLens 1m Data Set* por possuir o *Título* dos filmes (conteúdo utilizado para identificar o filme na web como veremos adiante). Suas características apresentadas na Tabela 3.1 foram extraídas da atividade dos usuários do sistema a partir do ano 2000. Esse conjunto de dados foi uma contribuição dos pesquisadores Shyong Lam e Jon Herlocker que filtraram e limparam os dados do sistema gerando os seguintes arquivos:

- ratings.dat - arquivo no formato "UserID::MovieID::Rating::Timestamp" e cada usuário tem ao menos 20 avaliações registradas.
- users.dat - arquivo no formato "UserID::Gender::Age::Occupation::Zip-code" no qual as informações demográficas são fornecidas voluntariamente pelos usuários sem checagem da veracidade.
- movies.dat - arquivo no formado "MovieID::Title::Genres" e os títulos incluem o ano do *release*.

Originalmente, o conjunto de dados não dispõe de muito conteúdo que possa caracterizar os itens (filmes), somente título, gênero e ano do *release*. Para tratar este problema utilizamos os dados dos filmes disponíveis e construímos um *Web crawler*[4] para pesquisar na web informações sobre os títulos dos filmes capturando mais dados que possam melhor representá-los. Para tornar mais simples o trabalho do nosso *Web crawler* reduzimos as buscas ao IMDb que é um conjunto de dados *online* mantido de modo colaborativo. Este sistema oferece aos usuários informações detalhadas sobre filmes, séries de TV, atores e atrizes, diretores e roteiristas. O sistema permite que qualquer usuário envie novas edições de material relacionado com o tema do site. Entretanto, todos os dados são verificados antes de ficar *online*, mas ainda assim admite-se eventuais erros. Assim como no *MovieLens* os usuários também são convidados a avaliar qualquer filme em uma escala de 1 a 10, e os totais são convertidos em uma classificação média ponderada que é exibida ao lado de cada título. O site também conta com um Fórum e um sistema colaborativo de *tagging* que estimula outro modo

[4] "...é um programa de computador que navega pela World Wide Web de uma forma metódica e automatizada. Outros termos para Web crawlers são indexadores automáticos, bots, web spiders, Web robot, ou Web scutter" Wikipédia.

de classificação dos filmes por rótulos definidos pelos usuários. Na Figura 3.2 apresentamos a página inicial do site IMDb.

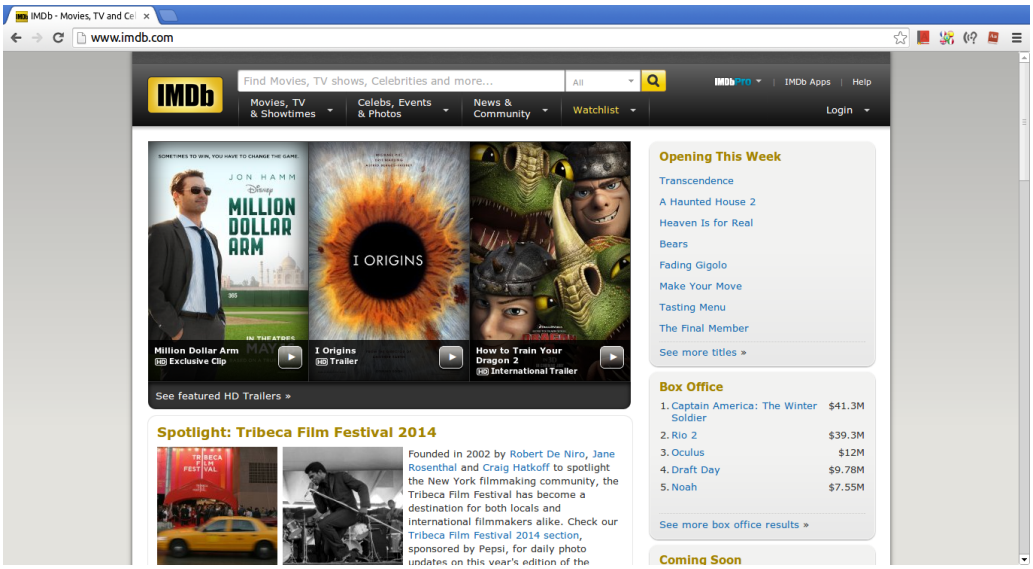

Fig. 3.2: Página inicial do sistema *IMDb* que disponibiliza um banco de dados sobre filmes, séries de TV, atores, diretores e relacionados.

3.1.2 CiteULike

O *CiteULike*[5] é um site na web que disponibiliza para seus usuários um serviço para organizar sua biblioteca de referências científicas. O serviço é baseado na tecnologia de *Social Bookmarking*, no qual o usuário é capaz de adicionar, editar, anotar e compartilhar apontadores para documentos na web. Cada usuário cadastrado no sistema pode gerenciar seu perfil manipulando apontadores para artigos, relatórios, dissertações ou qualquer outro tipo de documento científico de modo colaborativo com os demais usuários. Dessa forma, um artigo adicionado por um usuário pode ser adicionado a biblioteca de outro e também ter suas características modificadas com adição de novos metadados à citação. Na Figura 3.3 apresentamos um *screenshot* da página inicial com o perfil de um usuário do *CiteULike*.

Os dados das atividades dos usuários no sistema *CiteULike* estão disponíveis na página web http://www.citeulike.org/faq/data.adp. Diferentemente do *MovieLens* o conjunto de dados não possui qualquer tipo de filtro de seleção de dados e os dados brutos são disponibilizados pelos mantenedores do sistema para pesquisas.

No *CiteULike* o usuário não é convidado a avaliar os itens de seu perfil ou qualquer outro presente no sistema. É possível, no entanto, definir o interesse do usuário de forma implícita: qualquer artigo científico presente na biblioteca do usuário deve ser de seu interesse (ver conjuntos $d+$ e $d-$ apresentados na Seção 2.4.1). Na Tabela 3.2 apresentamos a configuração inicial do *dump* obtido do sistema *CiteULike*. O download do arquivo *dump* denominado *current.bz2* foi realizado em Outubro de 2013

[5]http://www.citeulike.org/

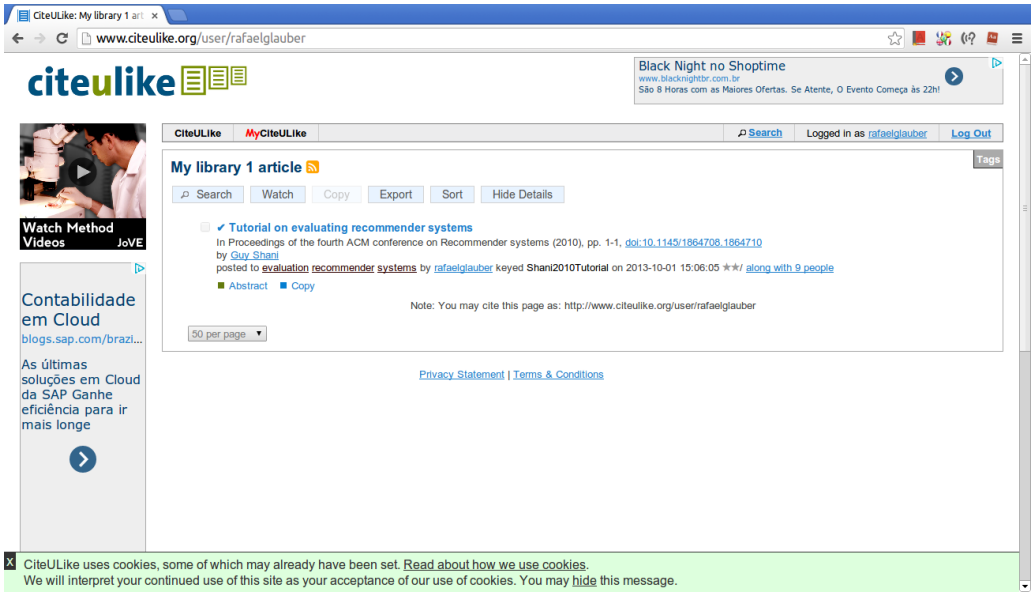

Fig. 3.3: Página do sistema *CiteULike* que permite os usuários criar bibliotecas de artigos científicos.

Propriedade	**CiteULike**
#usuários	132377
#itens	4398320
#atividades	19723576
proporção #itens / #usuários	33,2257
média #itens por usuário	43,0617
média #usuários por item	1,2960
max #itens por usuário	71510
max #usuários por item	1092
min #itens por usuário	1
min #usuários por item	1
esparsidade $usuário \times item$	0,99999

Tab. 3.2: Características do Conjunto de Dados CiteULike sem aplicação de qualquer filtro.

e conta com atividades dos usuários desde 30 de Maio de 2004. Os registros no arquivo seguem o seguinte formato delimitado pelo pipe ("|"):

- Um identificador (número inteiro) para o artigo que foi registrado na biblioteca do usuário.
- Uma representação de 32 caracteres em hexadecimal do nome do usuário que realizou a atividade. Este dado é resultado do código *hash* do *username* do usuário utilizando o algoritmo MD5.
- A data e a hora que ocorreu o registro no sistema.
- O *tag* (marcador) usado pelo usuário no registro para classificar o artigo científico.

Caso o usuário utilize mais de um *tag* significa que o registro será feito por mais de uma linha no arquivo *dump*. Artigos adicionados sem informar qualquer marcador ou com somente um marcador são registrados em uma única linha. Assim como o *MovieLens* foi necessário construir um *Web crawler* para obter mais dados sobre as publicações presentes nas bibliotecas dos usuários. Para isso foi utilizado o identificador dos apontadores como chave de busca no próprio sistema.

Discussão sobre os Conjuntos de Dados A decisão de utilizar dois conjuntos de dados distintos visa a avaliação em diferentes domínios e situações. Porém, para atender a essa decisão metodológica é necessário adaptar os conjuntos de dados para serem utilizados em nossos algoritmos de recomendação. O *MovieLens* disponibiliza três configurações fixas para os conjuntos de dados que já estão filtrados. Esse aspecto fornece a este conjunto de dados a possibilidade de comparar nossos resultados com outros trabalhos de forma direta com base nas medidas que serão utilizadas. Em contrapartida, o conjunto de dados *CiteULike* não teve qualquer tipo de filtragem o que confere a este artefato maiores possibilidades de trabalhar diferentes aspectos dos sistemas de recomendação. Problemas como da *partida a frio* não são bem analisados utilizando *MovieLens* por conta dos filtros aplicados, mas podem ser trabalhados com *CiteULike* desde que os filtros aplicados a este conjunto possibilitem realizar a tarefa de recomendação desejada para usuários e/ou itens que possuam poucas avaliações. Outro aspecto que deve ser tratado é o tipo de avaliação realizada nos dois sistemas: escala numérica e booleana. Para que os mesmos recomendadores que serão descritos nas Seções 3.2 e 3.3 sejam utilizados sem qualquer tipo de adaptação tomamos a seguinte decisão de projeto: os filmes avaliados no *MovieLens* com nota entre 0 e 3 formam o conjunto $d-$ dentro do perfil do usuário. Neste caso assumimos que somente os filmes com avaliação entre 4 e 5 são realmente relevantes para o usuário e devem formar o conjunto $d+$. De modo análogo no *CiteULike* todos os itens presentes na biblioteca do usuário formam o conjunto $d+$ e o conjunto $d-$ é vazio, uma vez que o usuário não manifesta explicitamente seu desinteresse pelos itens do conjunto de dados.

3.2 Recomendador Social

Todos os recomendadores utilizados em nosso experimento para a tarefa de recomendar bons itens para os usuários do *MovieLens* e *CiteULike* são baseados em memória. Mais precisamente nosso *baseline* é um recomendador social como descrito no trabalho de Su & Khoshgoftaar (2009). A Filtragem Colaborativa pode ser baseada em usuário ou em item e optamos pela recomendação por meio da similaridade entre os usuários. Quando baseado no usuário, o sistema recomenda bons itens para um usuário ativo identificando, primeiro, os k usuários mais semelhantes utilizando alguma medida de similaridade (Breese et al., 1998) entre os perfis dos usuários. Os itens existentes nos perfis dos usuários mais próximos formam um grupo que, possivelmente, interessam ao usuário ativo. Os itens mais populares dentro deste grupo formam uma lista ordenada pela popularidade entre os vizinhos que é apresentada ao usuário. Este tipo de algoritmo pode ser dividido em três grandes atividades: calcular a similaridade entre os usuários, em seguida, selecionar os vizinhos do usuário ativo e ao final identificar os itens mais populares entre os vizinhos. Vejamos detalhes de cada uma delas:

Calcular Similaridade entre os Usuários A primeira etapa envolve localizar usuários com interesses semelhantes ao usuário ativo. O cálculo de similaridade depende de como o usuário é modelado e qual o tipo de dado referente às avaliações dos itens estão disponíveis. Em nosso trabalho consideramos somente o conjunto $d+$ para modelar as preferências dos usuários. Dessa forma podemos utilizar o mesmo algoritmo para os dois conjuntos de dados, pois o *CiteULike* não possui o conjunto $d-$. Desta forma o perfil do usuário forma um vetor binário em que cada dimensão representa um item diferente dentre os itens do sistema. Em seguida é associado o valor 1 a cada item pertencente ao conjunto $d+$ do usuário e as demais posições recebem valor 0. O cálculo de similaridade entre dois usuários é determinada utilizando a medida *Cosseno* vista na Equação 2.4.

Seleção de Vizinhos Depois de calcular a similaridade entre o usuário ativo e os demais usuários do sistema são escolhidos os *k-vizinhos* mais próximos do usuário ativo utilizando o grau de similaridade calculado. Utilizar somente os melhores vizinhos preserva a ideia que somente os usuários com preferência mais parecida com o usuário ativo são capazes de sugerir itens interessantes (princípio do algoritmo de classificação kNN).

Popularidade dos Itens Escolhidos os vizinhos é iniciada a etapa de recuperação dos itens que farão parte da lista de recomendação. Nesta etapa é verificado quais itens pertencentes aos perfis dos vizinhos do usuário ativo são mais populares. Cada vizinho atribui um voto aos itens de seu perfil com peso igual ao seu grau de similaridade ao usuário ativo. Todos os K vizinhos votam e os votos são acumulados para cada item. Ao final todos os itens votados são ordenados pelo total da votação obtida no pleito de popularidade. Os itens mais votados que não estejam presentes no perfil do usuário (supostamente desconhecidos ao usuário) são apresentados em uma lista do tipo *TopK*.

Discussão sobre Filtragem Colaborativa A Filtragem Colaborativa, desde seu surgimento na década de 90, se tornou um padrão para os Sistemas de Recomendação. Quando configurada para recuperar itens baseada na relação $usuário \times usuário$ tem como princípio apresentar aos usuários itens que outros usuários de preferência semelhante tenham demonstrado alto grau de interesse. Este fundamento confere uma característica a este tipo de recomendador difícil de ser reproduzida por técnicas por conteúdo: questões subjetivas podem ser preservadas no processo de filtragem. Pesquisadores como Konstan & Riedl (2012) e Adomavicius & Tuzhilin (2005) atribuem a este princípio a vantagem desta abordagem. Um exemplo dessa "subjetividade" é o antagonismo existente entre os fãs das séries *Star wars* e *Star trek*. Por meio do conteúdo se encontraria similaridade (50% para o título), mas em um sistema colaborativo se verificaria, possivelmente, que estes fãs possuem gostos antagônico. Este trabalho não visa questionar este tipo de afirmação por parte dos pesquisadores, até por que resultados superiores de precisão são normalmente conferidos aos recomendadores sociais em relação aos baseados em conteúdo. Porém, buscamos mostrar que um Sistema de Recomendação com somente este princípio pode ser pobre aos interesses do usuário quanto a outros aspectos relevantes e que serão avaliados ao final deste trabalho.

3.3 Recomendadores Baseados em Conteúdo Textual

Os algoritmos de recomendação Baseados em Conteúdo são filtros de informações que utilizam técnicas de RI e AM para realizar sua tarefa. Para recomendação utilizando conteúdo que é foco de nosso trabalho utilizamos o conteúdo textual presente nos itens do perfil dos usuários para modelar suas preferências e realizar a tarefa de recomendação desejada. Propomos trabalhar com dois diferentes algoritmos, Filtragem por Agregação dos Itens do Perfil e Filtragem pela Similaridade dos Itens do Perfil, que realizam a filtragem por conteúdo e serão detalhados nas Seções 3.3.1 e 3.3.2. Nossa hipótese é que mesmo sendo a mesma abordagem estes algoritmos podem recuperar conteúdos diferentes devido aos métodos distintos de executar a recomendação.

Como base para nossos dois algoritmos é necessário, primeiro, criar um modelo vetorial que represente qualquer item do sistema. A representação visa tratar o conteúdo dos itens, mesmo dados estruturados como o Gênero do filme (e.g., Drama, Romance...) também como dados semiestruturados. A representação *Bag-Of-Words* amplamente utilizada como representação de textos em sistemas de RI será utilizada pelos dois algoritmos para realizar suas inferências de similaridade entre os documentos (itens do sistema). Na etapa de pré-processamento realizamos os seguintes filtros: remoção de *Stopwords* e remoção de termos com $DF = 1$. Estes dois filtros garantem que palavras sem valor semântico não sejam utilizados para definir similaridade entre itens. Além de descartar termos que ocorrem exclusivamente em um item, não sendo útil portanto para definir similaridade entre itens e permitindo também diminuir substancialmente o espaço vetorial a ser processado (problema da calda longa).

3.3.1 Filtragem por Agregação dos Itens do Perfil

Filtragem por Agregação dos Itens do Perfil (AIP) é um algoritmo simples de recomendação baseado em conteúdo que tem como princípio um filtro de conteúdo por palavra chave. Na Figura 3.4 apresentamos uma abstração de como funciona a geração das recomendações através deste algoritmo. O modelo que representa as preferências do usuário é um conjunto de palavras chave obtidas a partir de todos os itens do perfil do usuário. A ideia é que o interesse do usuário pode ser modelado juntando todas as palavras advindas dos itens do seu perfil e, posteriormente, identificando as palavras mais relevantes entre elas. Este algoritmo pode ser subdividido em três etapas principais executadas sequencialmente. A primeira, denominada Criar Modelo de Representação dos Documentos, em seguida Extrair as Palavras Chave do Perfil do Usuário e ao final Calcular Similaridade com o Modelo e Compor Lista de Recomendação. Vejamos mais detalhes sobre cada uma dessas etapas.

Criar Modelo de Representação dos Documentos Essa etapa realiza a transformação dos itens, representados por seus conteúdos textuais (título, descrição, comentários...) em uma nova representação baseada em BOW. Neste ponto o conteúdo de todos os itens do perfil do usuário (novamente destacamos que só é utilizado o conjunto $d+$) formam um único documento (documento perfil do usuário) e este é transformado em um vetor de palavras que recebe como peso o valor calculado de $TF \times IDF$.

Extrair as Palavras Chave do Perfil do Usuário Essa etapa é semelhante ao processo de *Feature Selection* (Seleção de Características) como descrito no relatório de Joachims (1996). Por meio dos

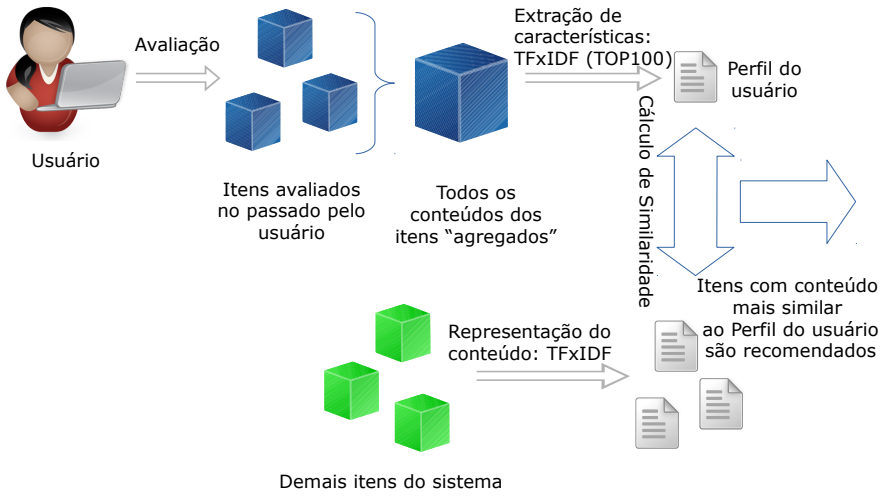

Fig. 3.4: Abstração da geração das recomendações através do algoritmo AIP.

valores atribuídos ao vetor que representa o perfil do usuário, no qual cada palavra possui o valor de $TF \times IDF$, construímos um ranking das palavras mais representativos e os N termos mais significativos são extraídos do ranking para formar um vetor de palavras que modelam as preferências do usuário.

Este algoritmo necessita de um parâmetro K para determinar a quantidade de termos que definem as preferências dos usuários. Em nosso trabalho fixamos $K = 100$ para essa variável. Esta foi uma escolha que busca redução de dimensionalidade do modelo do usuário e leva em consideração a hipótese de que, assim como em tarefas de RI, um SR não necessita de uma grande quantidade de características para recuperar conteúdo relevante. Trabalhos de Recuperação de Informação sobre *Feature Selection* como os de Jing et al. (2002) e Yang & Pedersen (1997) demonstram a importância deste tema e da influência que a definição do valor de N pode ter em nosso algoritmo. No trabalho de Liu et al. (2003) verificamos o impacto na precisão dos algoritmos de agrupamento ao determinar diferentes percentuais na obtenção das características dos documentos. Vimos ainda que é possível obter altos valores de precisão mesmo utilizando uma quantidade pequena de características dos documentos. Obviamente, um trabalho de otimização seria válido sobre essa variável, mas isso não é foco de nosso trabalho e deixamos essa questão em aberto para futuros trabalhos de pesquisa sobre o tema.

Calcular Similaridade com o Modelo e Compor Lista de Recomendação Este é um procedimento final do algoritmo que calcula a similaridade entre o modelo do usuário e os demais documentos (itens do sistema). Utilizando as medidas de similaridade *Cosseno* ou *Jaccard* é possível identificar o grau de similaridade (valor numérico) com todos os itens ainda não avaliados pelo usuário. Qualquer item que possua similaridade maior que zero forma um ranking de itens que são utilizados para criar a lista de recomendação. A lista de recomendação é um corte do ranking em algum ponto

e esta lista em seguida é entregue ao usuário.

3.3.2 Filtragem pela Similaridade dos Itens do Perfil

Filtragem pela Similaridade dos Itens do Perfil (SIP) é outro algoritmo de recomendação Baseado em Conteúdo Textual construído para nosso experimento que visa aplicar outro tipo de recuperação de conteúdo relevante para o usuário com relação ao algoritmo AIP. Na Figura 3.5 apresentamos uma abstração de como funciona a geração das recomendações através deste algoritmo. A ideia central deste algoritmo é que cada item do perfil do usuário é um eleitor e os demais itens do sistema são os candidatos. Uma "votação" ocorre calculando a similaridade entre os votantes e os candidatos e o grau de similaridade obtido entre o votante e o candidato é utilizado como peso no voto. Os candidatos que acumulem mais votos são recuperados como uma recomendação. Este algoritmo é divido em três etapas principais e abaixo veremos cada uma delas.

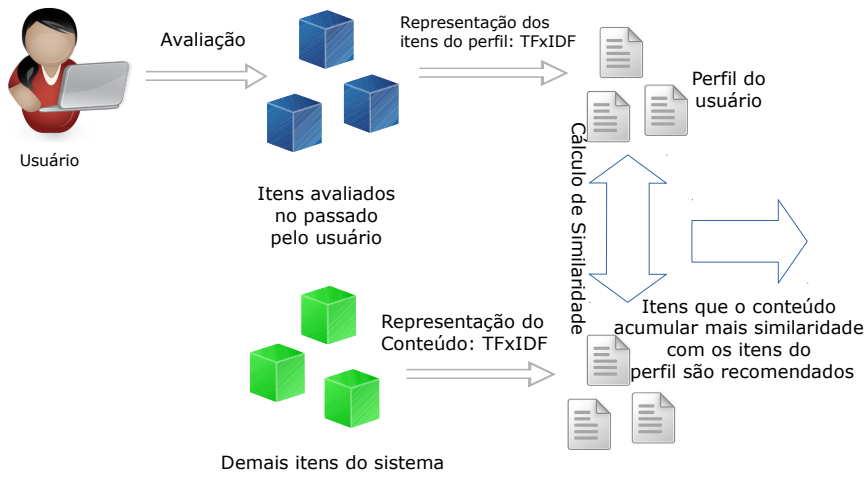

Fig. 3.5: Abstração da geração das recomendações através do algoritmo SIP.

Criar Modelo de Representação dos Documentos Essa etapa, assim como no algoritmo AIP, realiza a transformação dos itens, representados por seus conteúdos textuais (título, descrição, comentários...) em uma nova representação baseada em BOW. A diferença é que os itens do perfil do usuário são tratados individualmente no momento e não agregados como no AIP. Cada documento é transformado em um vetor com todas as suas palavras que recebe o valor calculado de $TF \times IDF$.

Calcular Similaridade entre Eleitores e Candidatos Esta é uma fase na qual o algoritmo calcula a similaridade entre cada item do perfil do usuário e os demais documentos (itens do sistema). Utilizando as medidas de similaridade *Cosseno* ou *Jaccard* é possível identificar o grau de similaridade

(valor numérico) com todos os itens ainda não avaliados pelo usuário. Todos os pares de documentos que possuam similaridade maior que zero são armazenados como entradas da próxima etapa.

Compor Lista de Recomendação A etapa final do algoritmo é identificar quais os itens do sistema que ainda não foram avaliados pelo usuário, mas que podem ser interessantes para o mesmo. Este processo é realizado por meio de uma votação, na qual cada item do perfil do usuário (conjunto $d+$) é um eleitor e os demais itens do sistema são candidatos no pleito. Cada eleitor vota em K candidatos atribuindo o grau de similaridade entre eles (calculado na etapa anterior). Os candidatos mais votados, ou seja, que acumularam mais votos (similaridades acumuladas) formam um ranking que é utilizado para gerar listas de recomendação.

Este segundo algoritmo de recomendação Baseado em Conteúdo solicita a definição da variável K que limita a quantidade de candidatos que um eleitor pode atribuir seu voto. Este parâmetro visa evitar a recomendação de itens que possuam pouca similaridade com as preferências do usuário ou que foram aproximados por palavras sem significado semântico para a caracterização dos itens. Entretanto, temos novamente um problema de otimização assim como o primeiro algoritmo por conteúdo. Neste ponto nossa decisão foi utilizar $K = 50$ definindo que somente os 50 documentos mais similares a cada item do perfil do usuário podem participar do pleito que define a lista de recomendação.

Discussão sobre os Recomendadores por Conteúdo Textual Aos recomendadores baseados em similaridade do conteúdo se confere fortes críticas relacionadas a sua incapacidade de surpreender o usuário recuperando um item de alto grau de interesse e que seus atributos não possuam similaridade com qualquer outro avaliado anteriormente pelo usuário. Entretanto, na experiência diária do usuário é possível que determinadas características, mesmo não sendo as únicas, sejam relevantes e determinantes para uma escolha do usuário. Em um sistema de locação de filmes o fato de uma grande quantidade de avaliações de um usuário terem sido direcionadas para um conjunto específico de atores/diretores pode indicar um alto grau de interesse em novas películas estreladas por estes indivíduos. Assim como em um sistema de apontadores para publicações acadêmicas perceber uma preferência por um determinado editorial pode apresentar uma oportunidade de apresentar novos conteúdos ao usuário. Desta forma pretendemos evidenciar melhor as características desse tipo de abordagem e utilizando algoritmos distintos perceber se mesmo utilizando a mesma abordagem é possível recuperar ou organizar diferentes recomendações para um mesmo usuário em um determinado momento do tempo.

3.4 Filtro de Dados

Muitos sistemas que utilizam recursos de recomendação são acessados por milhares de usuários todos os dias. *Google News*, *Amazon.com* e *Netflix* são exemplos de plataformas na web com grande volume de acessos dos usuários. As atividades dos usuários desses sistemas representam um volume de dados muito grande. Normalmente, as pesquisas sobre Sistemas de Recomendação utilizam amostras de conjuntos de dados reais para realizar a experimentação dos recomendadores. Por exemplo, no trabalho de McNee (2006) as experimentações são conduzidas utilizando pequenas representações dos conjuntos da dados disponíveis buscando generalizar a totalidade do problema que se deseja resolver.

Quando optamos por utilizar os dados disponíveis do *MovieLens* o processo de filtragem foi realizado pelos pesquisadores que os disponibilizaram. Neste caso optou-se inicialmente por não alterar os dados para manter nosso trabalho comparável com outras pesquisas. Porém, ao integrar os dados do *MovieLens* ao *IMDb* ocorreram diversos problemas para identificar os títulos dos filmes no *IMDb*. Na Tabela 3.3 apresentamos alguns exemplos de problemas que ocorreram no *Web crawler* e que precisaram ser tratados.

Título MovieLens	**Título IMDb**	**Problema Ocorrido**
Three Amigos! (1986)	¡Three Amigos! (1986)	Adicionado um caractere especial
101 Dalmatians (1961)	One Hundred and One Dalmatians (1961)	Título escrito por extenso
Two Moon Juction (1988)	Two Moon Junction (1988)	Escrita errada
Bamba, La (1987)	La Bamba (1987)	Transposição do artigo
Dorado, El (1967)	El Dorado (1966)	Transposição do artigo
Three Ages, The (1923)	Three Ages (1923)	Omissão do artigo
Story of G.I. Joe, The (1945)	Story of G.I. Joe (1945)	Omissão do artigo
Tarantella (1995)	Tarantella (1996)	Diferença no ano do release
Dorado, El (1967)	El Dorado (1966)	Diferença no ano do release
Santa Claus: The Movie (1985)	Santa Claus (1985)	Título alternativo
Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	Sunset Blvd. (1950)	Título alternativo
Open Your Eyes (Abre los ojos) (1997)	Abre los ojos (1997)	Título em diferentes idiomas
Dear Diary (Caro Diario) (1994)	Caro diario (1994)	Título em diferentes idiomas

Tab. 3.3: Alguns dos problemas ocorridos na busca dos títulos no *IMDb*.

Além dos problemas de localização dos títulos que foram tratados, dois títulos não foram identificados (arquivo movies.dat):

- 1697::Big Bang Theory, The (1994)::Crime
- 3027::Slaughterhouse 2 (1988)::Horror

Possivelmente, ocorreram modificações na base de dados do sistema que impediram a identificação no sistema IMDb e, por isso, estes dois títulos foram removidos do conjunto de dados. Consequentemente, também removemos as avaliações dos usuários para os dois títulos não identificados no IMDb. Outro problema foi a duplicidade de títulos com identificadores diferentes, vejamos:

- 811::Bewegte Mann, Der (1994)::Comedy
- 860::Maybe, Maybe Not (Bewegte Mann, Der) (1994)::Comedy
- 1741::Midaq Alley (Callejón de los milagros, El) (1995)::Drama

- 1795::Callejón de los milagros, El (1995)::Drama

Optamos por remover o título 811 do conjunto de dados final devida a sua menor incidência nas avaliações dos usuários. O título 1741 foi outro removido, mas por outro critério, pois este apresenta mais avaliações dos usuários, porém o usuário 195 possui avaliação para os dois títulos duplicados. Desta forma optamos por manter a última avaliação do usuário que foi para o título 1795. Outra adaptação necessária neste conjunto de dados é a remoção das avaliações negativas (avaliação ≤ 3) como justificado anteriormente. Nossas implementações de algoritmos, tanto de Filtragem Colaborativa, quanto de Filtragem Baseada em Conteúdo, levam somente em consideração o conjunto $d+$ na tentativa de generalizar ao máximo as avaliações das abordagens nos diferentes contextos de avaliação dos algoritmos.

Nosso outro conjunto de dados a ser explorado, *CiteULike*, possui características distintas do *MovieLens* e é disponibilizado sem qualquer tipo de filtro por seus mantenedores. Para realizar os testes de sistemas neste contexto vamos realizar um conjunto de filtros definidos como se segue:

Filtro de Período É o primeiro critério de filtro nos dados. Na Figura 3.6 verificamos a distribuição das atividades dos usuários ao longo dos anos, desde a criação da plataforma. As atividades apresentadas são extraídas de uma cópia do banco de dados contendo os registros até Outubro de 2013. Mesmo considerando parte das atividades de 2013 não registradas é possível verificar que o pico de atividades do sistema foi no ano de 2010 e que nos primeiros anos o volume de atividades, apesar de crescente, era baixo. Optamos por utilizar os dados de atividades dos usuários que entraram no sistema a partir de 2013. Consideramos que somente usuários que possuem atividades registradas no sistema a partir do primeiro dia de Janeiro de 2013 são analisados. Usuários que possuem atividades antes e depois de 2013 foram descartados, pois um princípio básico de nossos filtros de dados é manter inalterável o perfil dos usuários que serão submetidos aos recomendadores desenvolvidos neste trabalho. Aqui entendemos que estudar somente os novos usuários pode diminuir bastante o volume de dados, viabilizando as experimentações que propomos, mas mantendo a tendência de uso do sistema atual e para os próximos anos.

Filtro de Tamanho de Perfil Na Figura 3.7 apresentamos um gráfico com a distribuição dos usuários sobre a quantidade de itens de seu perfil. O gráfico foi construído com os eixos X e Y com base logarítmica para melhor compreensão da distribuição. Podemos dessa forma verificar que há uma pequena concentração de usuários com perfis com grande volume de itens em suas bibliotecas. Alguns usuários apresentam valores na ordem de 55000 publicações. Indo em sentido contrário há uma grande concentração de usuários com poucos itens presentes em suas bibliotecas. Acreditamos que para muitos desses usuários não houve interesse em permanecer utilizando o sistema e destes pouca informações sobre suas preferências está disponível. Desta forma estabelecemos dois cortes na amostra resultante do Filtro de Período: utilizaremos somente usuários com no mínimo 2 (duas) publicações em sua biblioteca e com no máximo 1000 (mil). Este corte elimina uma pequena quantidade de usuários com grandes quantidades de itens no perfil, além de restringir as recomendações para os usuários que ao menos possuam duas publicações adicionadas ao seu perfil. Como veremos ainda neste capítulo, a quantidade mínima será variável ao longo dos experimentos por conta de nosso protocolo de teste detalhado na Seção 3.5.

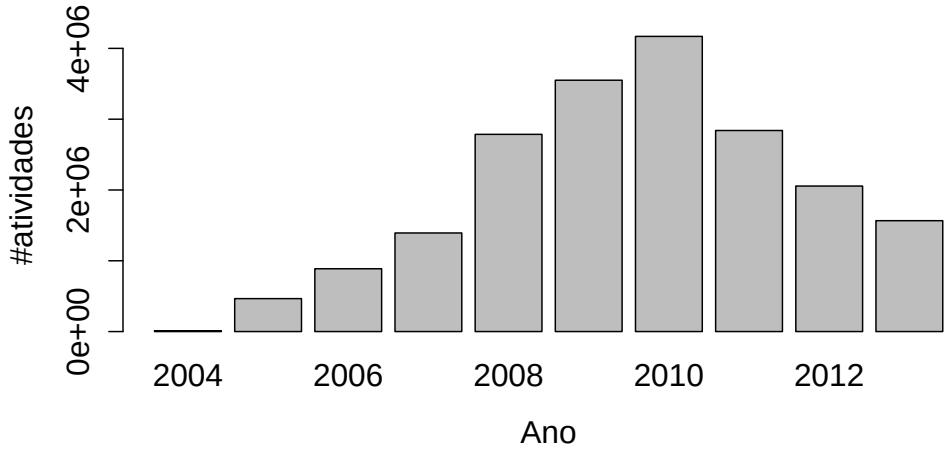

Fig. 3.6: Distribuição das atividades dos usuários no sistema *CiteULike* nos anos.

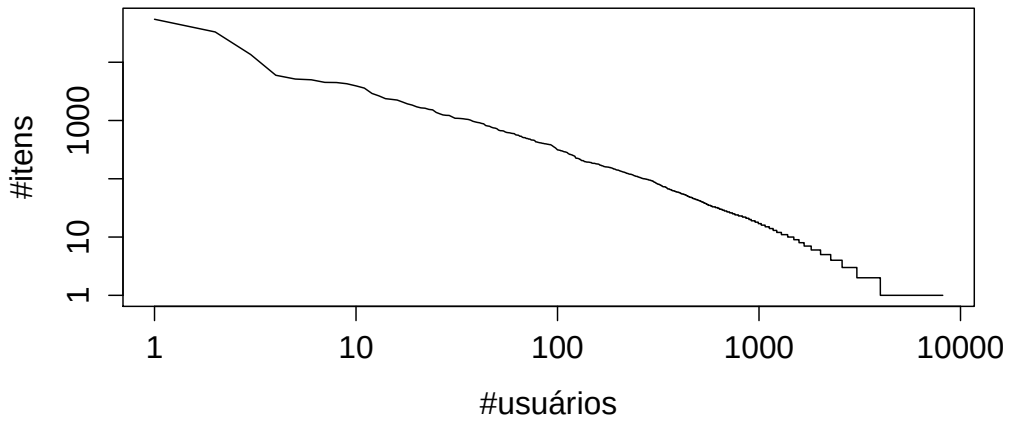

Fig. 3.7: Distribuição do tamanho do perfil dos usuários que entraram no sistema *CiteULike* em 2013.

Filtro de Idioma Em Sistemas de Recomendação Baseados em Conteúdo o princípio da construção das recomendações é feito utilizando a busca por itens similares às preferências dos usuários. O cálculo de similaridade é normalmente realizado por meio de comparações de tokens/palavras e por este fator estes recomendadores são fortemente influenciados pelo idioma no qual os atributos dos itens do sistema estão registrados. Sistemas como o *CiteULike* que permitem apontadores para publicações de qualquer idioma apresentam uma barreira aos recomendadores por conteúdo. Neste ponto decidimos remover os usuários que apresentem pelo menos 1 (uma) publicação em idioma diferente do Inglês. Para detecção de qual idioma está a publicação utilizamos o trabalho de Shuyo (2010) *Language Detection Library for Java* e utilizando os atributos de título e resumo das publicações

realizamos a classificação dos itens. Acreditamos que um possível tratamento para essa questão de diversidade de idioma, presente no *CiteULike*, seria a construção de recomendadores que atuariam somente nos grupos de itens de cada idioma. Como o idioma Inglês é responsável pelo maior volume de publicações mundialmente aceitas nosso trabalho se concentra exclusivamente nesta porção dos dados. Os demais usuários/publicações foram eliminados da amostra.

Filtro de Página Inválida Os conjuntos de dados que são criados da extração de dados de sistemas reais podem conter dados inválidos. Muitos registros recuperados das atividades dos usuários podem significar entradas inválidas no sistema em decorrência de mudanças no modelo de dados do sistema ou mesmo pelo baixo grau de normalização do modelo de dados. Uma etapa necessária para nossos recomendadores por conteúdo é obter dados dos itens do sistema. Para o *CiteULike* nosso *Web crawler* utilizou os códigos identificadores das publicações para pesquisar os artigos no sistema. Entretanto, em algumas buscas nosso *Web crawler* recebeu código 404 (página não encontrada). Não se conhece a origem dessas ocorrências, mas analisando os dados do ano 2013 não mais que 10 usuários apresentaram esta situação. Desta forma removemos estes usuários da amostra final utilizada.

Após todos os tratamentos realizados nos dois conjuntos de dados utilizados, *MovieLens* e *CiteULike*, apresentamos na Tabela 3.4 o resultado dos filtros e a configuração final dos conjuntos de dados.

Propriedade	**MovieLens – 1m**	**CiteULike**
#usuários	6038	3000
#itens	3533	32563
#atividades	575279	34037
proporção #itens / #usuários	0,5847	10,8543
média #itens por usuário	95,2764	11,3456
média #usuários por item	162,8301	1,0452
max #itens por usuário	1435	947
max #usuários por item	2853	20
min #itens por usuário	1	2
min #usuários por item	1	1
esparsidade $usuário \times item$	0,9730	0,9996

Tab. 3.4: Características dos conjuntos de dados utilizados após filtros.

Cada conjunto de dados recuperado possui atributos distintos, pois os atributos dos filmes diferem das publicações. O processo de extração de dados para os recomendadores baseados em conteúdo textual utilizou as páginas recuperadas pelo *Web crawler* realizando a análise das páginas web para cada um dos sistemas: *IMDb* e *CiteULike*. No primeiro sistema, com informações específicas sobre filmes e seriados, foram recuperados os dados apresentados na Tabela 3.5.

Os filmes recuperados no sistema *IMDb* são releases com data inferior ou igual ao ano 2000. Isso confere aos mesmos uma grande quantidade de informações, de modo colaborativo, que foram atribuídas ao longo dos anos. *Avaliação do usuário*, por exemplo, é um atributo recuperado no qual os usuários do sistema redigem suas impressões sobre o filme de modo textual. Os demais usuários podem indicar se a avaliação dos demais usuários o ajudaram ou não na escolha do filme. Recuperamos a avaliação mais popular entre os usuários para utilizar como atributo em nossos recomendadores. Os

Atributo	**Cobertura (%)**
Título	100
Sinopse	98,69
Palavras chave	99,88
Gênero	100
Atores principais	99,88
Escritores	98,07
Diretores	99,63
Avaliação do usuário	99,77
Ano do *release*	99,03

Tab. 3.5: Cobertura dos atributos extraídos pelo *Web crawler* no *IMDb*.

demais campos como *Título* e *Gênero*, com 100% dos filmes possuindo este atributo, são dados intrínsecos sobre os filmes e séries, assim como todos os demais. Em nossa avaliação o *Ano do release* foi removido por considerar que um usuário não deve apreciar um filme simplesmente pelo fato de ter apreciado outro que foi lançado no mesmo ano. Em futuros trabalhos seria possível elaborar uma abordagem na qual esse atributo fosse considerado para localizar grupos de usuários que compartilham o interesse por filmes antigos ou lançamentos. Seria possível, por exemplo, melhor identificar as preferências dos usuários e utilizar esse tipo de informação para aprimorar as recomendações.

O processo de extração de dados no sistema *CiteULike* segue o mesmo princípio de recuperar dados intrínsecos sobre as publicações postadas nas bibliotecas dos usuários, exceto pelo atributo *Tags* que representa todos os marcadores utilizados de modo colaborativo no item. Na Tabela 3.6 apresentamos a cobertura de cada atributo recuperado para o conjunto de dados final deste sistema.

Atributo	**Cobertura (%)**
Título	100
Resumo	81,35
Autores	97,65
Nome da publicação	78,31
Editor	38,50
Fonte	97,27
Tags (marcadores)	81,13
Ano da publicação	94,59

Tab. 3.6: Cobertura dos atributos extraídos pelo *Web crawler* no *CiteULike*.

Discussão sobre a Configuração Final dos Conjuntos de Dados Analisando individualmente os dois conjuntos de dados é possível identificar semelhanças de características entre o conjunto de dados inicial e o conjunto de dados final. Relacionado ao *MovieLens* os filtros aplicados modificaram pouco as características contabilizadas do conjunto de dados. Com relação ao *CiteULike*, por uma melhor acomodação do experimento a amostra final representa um pequeno percentual do conjunto original. Mesmo assim, algumas das principais características deste conjunto de dados foram preservadas. Primeiro, a baixa quantidade de usuários e a grande quantidade de itens com a consequente grande desproporção entre esses dois conjuntos. Segundo, a baixa quantidade média de itens no perfil dos

usuários que, associada à grande quantidade de itens, leva a uma altíssima esparsidade na relação $usuário \times item$.

Os dois conjuntos de dados, *MovieLens* e *CiteULike*, apresentam configurações muitos discrepantes entre si. Enquanto o *MovieLens* possui características interessantes para qualquer das duas abordagens de recomendação empregadas neste trabalho, seja social ou por conteúdo, o *CiteULike* não. O *MovieLens* possui uma esparsidade menor dos dados, além de um conjunto de itens menor e uma média de itens por usuário maior em comparação com o conjunto de dados *CiteULike*. Enquanto podemos afirmar que o conjunto de dados do *MovieLens* é um ambiente amigável para os algoritmos de recomendação, também podemos afirmar que os dados do *CiteULike* representam um desafio para as principais abordagens. Esparsidade para o recomendador social é um aspecto que dificulta muito a identificação de vizinhos, etapa fundamental dos principais filtros colaborativos. E grande quantidade de itens para o filtro por conteúdo que torna os cálculos de similaridade entre os documentos custosos computacionalmente. A expectativa ao final é analisar o comportamento das duas abordagens em cada um dos ambientes e o resultado desta experiência esclarecer melhor as diferenças e similaridades entre as duas abordagens principais de recomendação.

3.5 Avaliação Empírica

Em projetos de algoritmos de recomendação é comum a construção de *teste retroativo*, também conhecido como *avaliação do sistema* ou *avaliação offline* como parte do método de avaliação do sistema. Este método, melhor detalhado na Seção 2.6.2, está presente nos trabalhos de Breese et al. (1998) e Herlocker et al. (1999) e de muitos outros pesquisadores. Mesmo possuindo um princípio básico de ocultação de algumas das atividades dos usuários, para posterior predição, as configurações utilizadas nos diversos experimentos não é exatamente um consenso entre os trabalhos presentes em nossa revisão bibliográfica.

Em uma tarefa de recomendação do tipo "recomendar bons itens", foco de nosso trabalho, não existe um protocolo predefinido que, por exemplo, padronize a quantidade de itens que devem ser removidos do perfil do usuário ativo. Desde modo avaliaremos nossos recomendadores em configurações em que cada usuário terá de seu perfil removidos as porções 1, 5 e 10 itens (Given 1, Given 5 e Given 10). Isso significa que para cada usuário ativo realizaremos as avaliações da tarefa empregada com um conjunto de treinamento sem as ocorrências dos itens retirados do perfil que formam o conjunto de teste. Este procedimento não significa que os itens removidos do perfil do usuário não continuam no conjunto de treinamento, porém estes não estarão no perfil do usuário ativo, mas podem estar no perfil de outros usuários. Este método experimental, variando a quantidade de itens removidos do perfil do usuário, foi aplicado no trabalho de Breese et al. (1998) de modo a analisar o desempenho dos algoritmos com configurações diferentes em um mesmo conjunto de dados.

Em sistemas como o Goodreads[6] é solicitado ao usuário uma quantidade mínima de 20 avaliações (neste caso pelo menos 20 itens no perfil do usuário) para que o recomendador inicie a tarefa proposta. O conjunto de dados *MovieLens* em todas suas configurações originais (100K, 1M e 10M) estabelece como filtro a mesma quantidade mínima de itens avaliados por cada usuário recuperado. Entretanto, em aplicações reais é comum um baixo interesse dos usuário em fornecer suas impressões sobre os itens ou mesmo a necessidade de oferecer recomendações para novos usuários. No sistema *Netflix*

[6]https://www.goodreads.com/

de modo a combater o problema de *partida a frio* o usuário é convidado na criação de sua conta a avaliar uma lista inicial de filmes, mas este procedimento não é obrigatório. Ainda assim sistemas do mundo real devem cumprir com sua função sob qualquer tipo de circunstância. De modo a simular este problema nossa proposta é que para cada rodada de avaliação quando se remove uma porção de itens do perfil do usuário no mínimo a mesma quantidade deve permanecer em seu perfil. Portanto, quando nosso protocolo estiver configurado para Given 1 os usuários devem possuir, no mínimo 1 item em seu perfil para treinamento. Com Given 5 outros mesmos 5 itens devem existir no perfil do usuário e assim por diante. Nas Tabelas 3.7 e 3.8 apresentamos cada conjunto de dados para cada configuração deste protocolo de remoção de itens do perfil do usuário que foi utilizado.

Propriedade	**Given 1**	**Given 5**	**Given 10**
#usuários	6037	5950	5180
#itens	3533	3532	3526
#atividades	575278	574617	562798
média #itens por usuário	95,2920	96,5742	108,6482
média #usuários por item	162,8299	162,6888	159,6137
max #itens por usuário	1435	1435	1435
max #usuários por item	2853	2832	2608
min #itens por usuário	2	10	20
min #usuários por item	1	1	1

Tab. 3.7: Características do conjunto de dados *MovieLens* com cada configuração do protocolo de remoção de itens do perfil.

Propriedade	**Given 1**	**Given 5**	**Given 10**
#usuários	3000	813	380
#itens	32563	25122	19728
#atividades	34037	25914	20150
média #itens por usuário	11,3456	31,8745	53,0263
média #usuários por item	1,0452	1,0315	1,0213
max #itens por usuário	947	947	947
max #usuários por item	20	9	4
min #itens por usuário	2	10	20
min #usuários por item	1	1	1

Tab. 3.8: Características do conjunto de dados *CiteULike* com cada configuração do protocolo de remoção de itens do perfil.

Outra configuração importante dos conjuntos de dados utilizados é a distribuição de quantidade de itens no perfil dos usuários. É possível verificar essa configuração, primeiro, para as três configurações do protocolo de retirada de itens do perfil dos usuários no *MovieLens* (Given 1, 5 e 10) nas Figuras 3.8, 3.9 e 3.10. Os três histogramas foram criados somando, para cada usuário, a quantidade de itens de seu perfil que fazem parte do conjunto $d+$. Ou seja, consideramos para essa distribuição somente os itens considerados relevantes para cada usuário do sistema. Para facilitar a análise da distribuição os eixos x (quantidade de usuários) e y (quantidade de itens) foram plotados em escala

logarítmica. O comportamento predominante nas três configurações é uma quantidade pequena de usuários que possuem a quantidade mínima de itens no perfil para Given 1, Given 5 e Given 10. Além disso, como anteriormente verificado na Tabela 3.7, cada configuração do conjunto de dados *MovieLens* difere pouco suas características se comparado os valores das características encontradas no *CiteULike*.

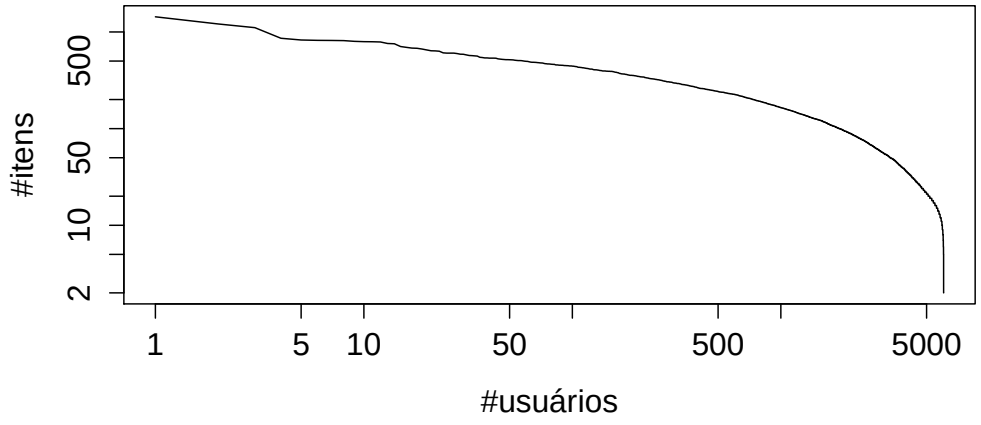

Fig. 3.8: Distribuição do tamanho do perfil dos usuários no conjunto de dados *MovieLens* em Given 1.

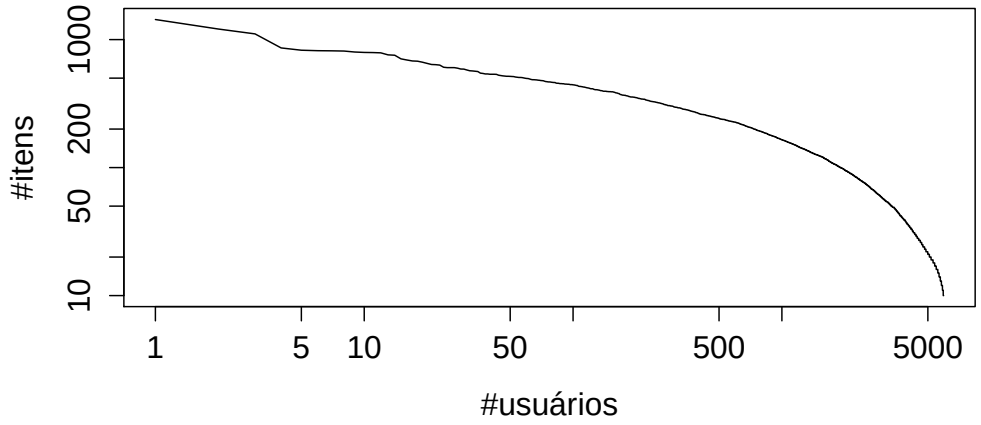

Fig. 3.9: Distribuição do tamanho do perfil dos usuários no conjunto de dados *MovieLens* em Given 5.

Apresentamos também a distribuição do tamanho do perfil dos usuários no conjunto de dados *CiteULike*. É possível verificar um comportamento muito diferente das distribuições quando comparadas com as distribuições encontradas em *MovieLens*. A principal característica das distribuições

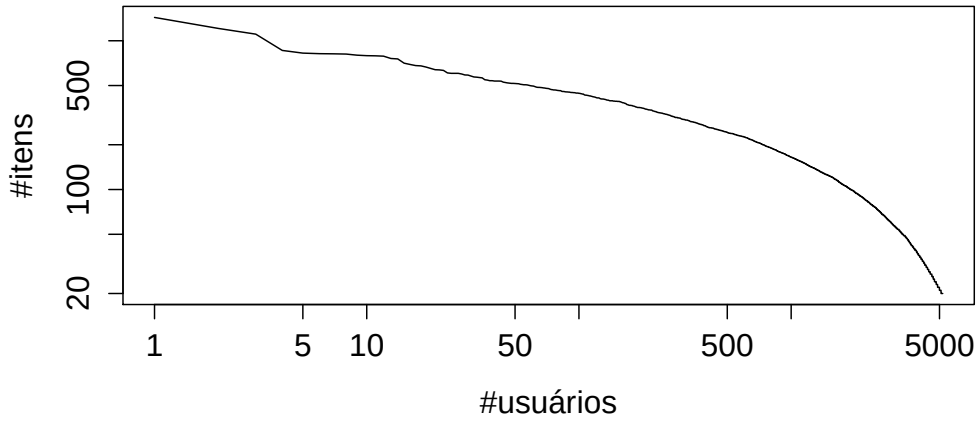

Fig. 3.10: Distribuição do tamanho do perfil dos usuários no conjunto de dados *MovieLens* em Given 10.

apresentadas para o *CiteULike* é uma maior concentração de usuários com tamanho de perfil considerados pequenos. Por exemplo, verificando a distribuição para Given 1, na qual é possível haver usuários com somente 2 itens no perfil, há uma grande concentração de usuários com tamanhos entre 2 e 10 itens. Usuários com estas quantidades de itens no perfil fornecem poucas informações sobre suas preferências tornando a tarefa de recomendação mais difícil (*user-cold-start*).

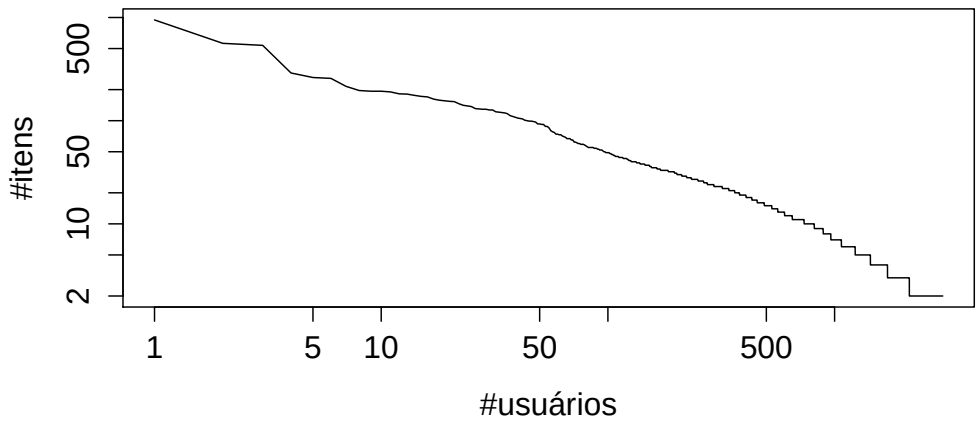

Fig. 3.11: Distribuição do tamanho do perfil dos usuários no conjunto de dados *CiteULike* em Given 1.

Herlocker et al. (2004) sugere que em uma tarefa de predição na qual está presente o *timestamp* dos registros é possível que as últimas atividades do usuário formem o conjunto de teste de modo

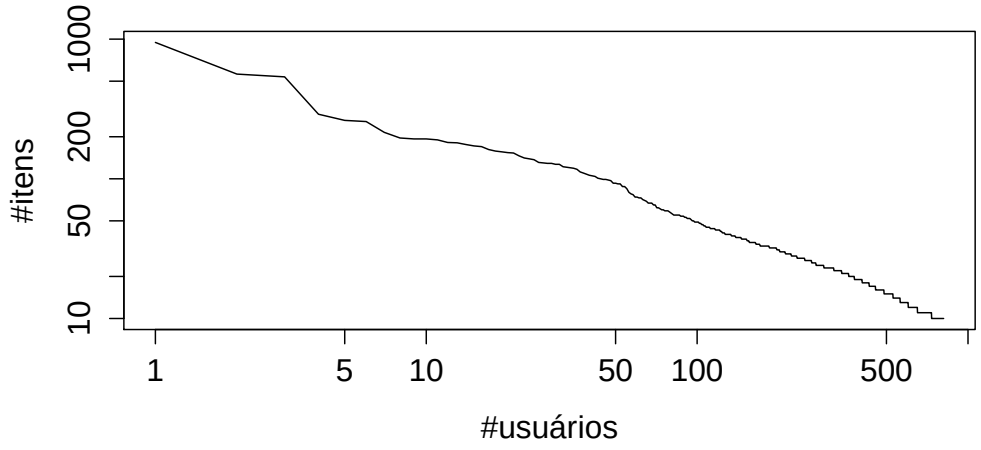

Fig. 3.12: Distribuição do tamanho do perfil dos usuários no conjunto de dados *CiteULike* em Given 5.

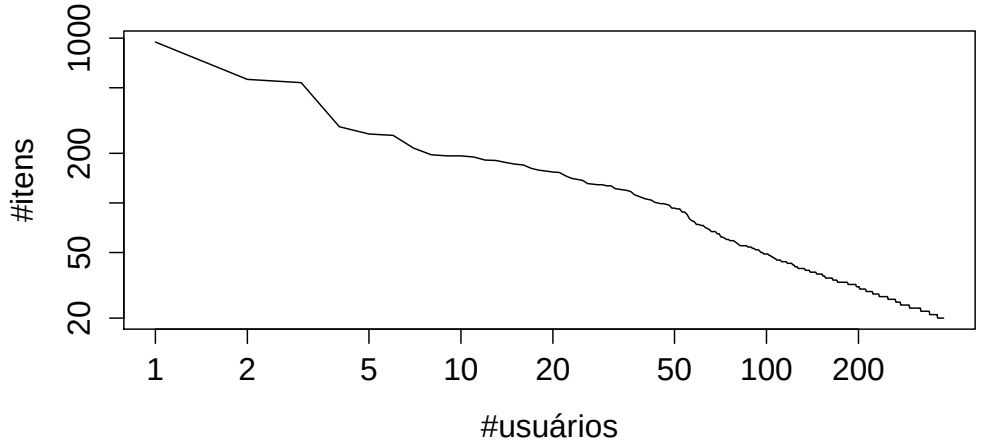

Fig. 3.13: Distribuição do tamanho do perfil dos usuários no conjunto de dados *CiteULike* em Given 10.

a simular o mais fielmente o comportamento do sistema. Esse comportamento é fundamental em recomendadores no qual o aspecto sequencial da tarefa é um requisito. Predizer uma sequência de musicas para uma *playlist* ou sugerir uma sequência de artigos para um estudante iniciar os estudos em uma área do conhecimento seriam exemplos nos quais o fator "ordem das recomendações" teria grande importância. Este aspecto não é foco de nosso trabalho e por isso as porções de itens do perfil do usuário foram removidos aleatoriamente.

Em uma tarefa de classificação os algoritmos empregados podem ter seus resultados influenciados por diversos aspectos dos conjuntos de dados analisados. Na etapa de treinamento os exemplos dispo-

níveis tem forte impacto na classificação dos itens de teste e por este motivo a busca da generalização dos algoritmos de Aprendizado de Máquina é quesito fundamental em seu projeto. Predizer preferências dos usuários é equivalente a uma tarefa de classificação e por isso o modelo preditivo criado deve ser generalizado para que aplicado em novos conjuntos de dados seu comportamento seja o mais estável quanto possível. O método de *cross-validation* denominado *k-fold* (Kohavi, 1995) é bastante popular e está presente nos trabalhos de Cremonesi & Turrin (2009), Bogers (2009) e Gantner et al. (2011). Este método consiste em dividir o conjunto total de dados em k subconjuntos mutualmente exclusivos do mesmo tamanho. Em seguida, um subconjunto é utilizado para teste e os $k-1$ outros subconjuntos são novamente reunidos e utilizados como um novo conjunto de treinamento que produz um novo modelo preditivo que deve ser avaliado (por meio da medida de avaliação desejada como vimos na Seção 2.6.2). Este processo é realizado k vezes de modo que todos os subconjuntos sejam testados. Ao final deste procedimento calcula-se a média da medida de avaliação empregada ao recomendador obtendo um resultado mais consistente sobre o algoritmo analisado. Em nosso trabalho definimos $k = 10$, logo cada subconjunto de dados representa 10% do conjunto de dados total.

3.6 Medidas para Avaliação

Estabelecido o protocolo de testes de nossos recomendadores por conteúdo textual e social é necessário selecionar quais as medidas que devem ser utilizadas para avaliar o comportamento desses algoritmos. Neste trabalho buscamos indícios mais claros sobre o comportamento das duas principais abordagens de recomendação e para isso precisamos analisa-las sobre diversas dimensões.

3.6.1 Precisão

Mean Average Precision foi escolhida como nossa principal medida de avaliação para os recomendadores. Esta medida avalia a lista de recomendação com base na posição que o item considerado relevante (neste caso removido do perfil do usuário) está localizado. Neste caso os melhores resultados são obtidos logo no início da lista, mas essa medida também pontua listas com acertos ao final. Em nossa avaliação estabelecemos tamanhos fixos para as listas de recomendação (10, 20, 30, 50 e 100) e assim a pontuação do MAP fica limitada a esta configuração. Por conta desse comportamento chamaremos essa medida ao longo do trabalho de *MAP at K* (MAP@K). Como veremos nas próximas seções esta configuração de limitar o tamanho das listas vai interferir na definição de diversas medidas utilizadas.

3.6.2 Cobertura

As características dos conjuntos de dados podem tornar a tarefa de recomendação empregada mais ou menos difícil para os diferentes algoritmos de recomendação de cada abordagem. Em um sistema no qual a relação $usuário \times item$ é muito esparsa, é muito difícil encontrar usuários com preferências semelhantes e, consequentemente, utilizar técnicas sociais para realizar a tarefa de recomendação. Em outro contexto, no qual não haja dados sobre os itens ou os mesmos sejam insuficientes para caracterizá-los, uma abordagem por conteúdo também pode falhar em sua tarefa. Para analisar a capacidade de um recomendador que executa a tarefa de recomendar "bons itens" em conseguir

apresentar recomendações aos usuários utilizamos o *User coverage* (*UCOV at K* ou simplesmente UCOV quando conveniente). Definimos essa medida como:

$$User\ Coverage = \frac{1}{|U|} \times \sum_{u=1}^{|U|} \frac{|L_u|}{K} \tag{3.1}$$

onde $|L_u|$ é o tamanho da lista de recomendação gerada para o usuário u e U é o conjunto de usuários ativos que devem receber listas de recomendação de tamanho K. Esta medida visa analisar a capacidade dos algoritmos testados em conseguir realizar sua tarefa de compor uma lista completa de recomendação para cada usuário. A expectativa é que situações de *partida a frio* para usuários com quantidade limitada de avaliações apresente resultados ruins de precisão, tanto para abordagens sociais, quanto para baseada em conteúdo. Entretanto, as abordagens por conteúdo tendem a conseguir recuperar itens mesmo em um cenário desfavorável como este que pouca informação sobre as preferências do usuário está disponível.

Em nossa metodologia experimental os recomendadores são chamados para realizar a tarefa de recuperar bons itens para cada usuário ativo uma única vez e assim o sistema é avaliado. Desta forma *Cobertura de Catálogo* (*CCOV at K* ou somente CCOV) para listas de tamanho máximo K deve ser adaptada em nosso experimento com base na proposta apresentada na Seção 2.6.2 da seguinte forma:

$$Cobertura\ de\ Catálogo = \frac{|L_{u=1} \bigcup L_{u=2} \bigcup ... \bigcup L_{u=|U|}|}{|I|} \tag{3.2}$$

onde I é o conjunto de itens disponíveis no sistema, ou seja, o catálogo. L é a lista de recomendação que é apresentada para cada usuário $u \in U$, onde U é o conjunto de usuários ativos. Como vimos no Capítulo 2 *Cobertura de Catálogo* é aplicada para um conjunto de recomendações realizadas para um mesmo usuário. Desta forma é medida, para cada usuário, a capacidade do sistema em cobrir o catálogo. A definição apresentada neste trabalho é uma modificação que altera o conceito original da medida de analisar cada usuário e avalia a cobertura considerando as recomendações para todos usuários. Com nossa modificação a quantidade de consultas realizadas pode ter maior impacto no resultado da medida, ou seja, quanto mais consultas (mais usuários para teste) maior será o valor obtido. Entretanto, para comparações entre algoritmos nos parece viável a modificação podendo indicar comportamentos diferentes entre as abordagens.

3.6.3 Similaridade de Recomendação

As principais medidas de avaliação dos Sistemas de Recomendação são medidas utilizadas por trabalhos em Recuperação de Informação e as mais tradicionais possuem a característica de testar as consultas uma a uma para calcular seu respectivo resultado. Entretanto, na nossa questão de pesquisa **Q3** buscamos identificar quais as semelhanças e diferenças entre as duas principais abordagens. Utilizando somente medidas como o MAP não é possível deixar claro esse aspecto, pois só trata o resultado em termos de acertos de previsão. Por exemplo, na tarefa de predição para o protocolo de remoção de itens em Given 10 dois algoritmos podem acertar 5 itens cada um. Neste caso os dois teriam a mesma precisão de $5/10$, mas este resultado pode encobrir uma questão: são os mesmos 5 itens acertados pelos dois algoritmos? Ou mais precisamente, em se tratando de abordagens tão distintas quanto BC e FC estes acertariam em suas previsões os mesmos itens? Além disso, todas as

previsões (incluindo os *Falsos-Positivos*) realizadas por cada um dos algoritmos das abordagens são semelhantes ou diferentes?

Por meio da tarefa de recomendação empregada pelos algoritmos deste experimento propomos calcular a similaridade entre as listas de recomendação, utilizando a *Similaridade Jaccard* vista na Seção 2.4.1. Este procedimento é realizado tomando a lista de recomendação gerada por cada um dos algoritmos, duas a duas, para o mesmo usuário ativo realizando o cálculo para a equação de *Jaccard*. Esta medida fornece o grau de similaridade entre duas listas de recomendação por meio de valores entre 0 e 1. Caso as listas não possuam nenhum item em comum o resultado é zero e pode apresentar valores maiores a medida que a quantidade de itens em comum cresce. Um aspecto importante desta análise é que a ordem nos quais os itens estão ordenados não é considerada no cálculo, por conta da natureza da medida de similaridade utilizada que realiza a operação com os conjuntos sem verificar a ordenação dos mesmos. Acreditamos que em pequenas listas de tamanho 10 ou 20 não é muito impactante o fator ordenação para a nossa análise e por isso o uso da *Similaridade Jaccard* pode ser mais adequada que o uso de outro tipo de medida como as de *correlação de rank*.

Encontrar similaridade entre as listas de recomendação é o ponto de partida para nossa análise de similaridade entre os recomendadores. Após essa etapa, propomos identificar o comportamento dos algoritmos em relação aos acertos das previsões. Plotamos um gráfico que conta para cada posição da lista de recomendação quantos acertos foram realizados por cada algoritmos entre todas as recomendações geradas. As curvas geradas por este gráfico definem em quais regiões das listas de recomendação cada um dos algoritmos possuem mais ou menos acertos de previsão indicando mais um comportamento relevante em nossa análise. Ao final, são verificadas das listas de recomendação as proporções de acertos gerados pelos recomendadores dois-a-dois. Nesta análise, tomamos dois recomendadores (suas listas de recomendação geradas para os usuários ativos) e separamos os acertos das previsões em três conjuntos distintos: os acertos exclusivos do algoritmo A, os acertos exclusivos do algoritmo B e a interseção dos acertos de A e B. O primeiro conjunto indica quais são as previsões acertadas exclusivamente pelo algoritmo A. O segundo conjunto é análogo ao primeiro e o terceiro são as previsões acertadas pelos dois algoritmos. A análise desses três conjuntos indica se há superposição dos acertos das previsões nas recomendações geradas por algum dos algoritmos. Ou seja, se algum dos algoritmos é suficiente para acertar todas as previsões quanto é possível pelos três algoritmos testados neste trabalho.

Discussão sobre as Medidas de Avaliação Abordamos os tipos de testes e medidas de avaliação no Capítulo 2 deste trabalho e levantamos diversas dimensões que podem ser avaliadas em um sistema de recomendação. Entretanto, escolhemos algumas dimensões a serem avaliadas e nossa decisão metodológica leva em consideração os seguintes aspectos: primeiro, em uma tarefa do tipo recomendar "bons itens" é inevitável o uso de uma medida como MAP de modo a verificar quais recomendadores são mais ou menos eficientes em realizar previsões. Mesmo considerando outras medidas como NDCG (Ravikumar et al., 2011), o MAP tem se tornado um padrão para testes de sistemas deste tipo. Segundo, extrapolando a questão da precisão das previsões, consideramos aspectos como diversidade e capacidade de cobrir o catálogo como aspectos relevantes e que podem ajudar a esclarecer algumas questões levantadas na introdução de nosso trabalho. Obviamente que as demais dimensões merecem atenção, mas abordar as mais diversas medidas foge do escopo de nosso trabalho. Outro ponto importante de nossa avaliação é a comparação dos recomendadores por outro viés: quão similares são as listas de recomendação geradas pelos três algoritmos analisados? Esta avaliação pode ajudar a

evidenciar melhor aspectos complementares que são sugeridos em trabalhos como o de Burke (2002), mas que são pouco evidenciados, principalmente por algum experimento. Considerando ambientes virtuais como *Netflix* e *Amazon.com* é possível desfrutar de recomendações das mais diversas tarefas e abordagens sugerindo que nenhuma abordagem sozinha, até o momento, é capaz de maximizar a experiência do usuário a níveis satisfatórios.

Capítulo 4

Resultados Obtidos

Neste capítulo apresentamos os resultados obtidos utilizando a metodologia experimental detalhada no Capítulo 3. Os resultados obtidos foram explorados para cada conjunto de dados utilizado (*MovieLens* e *CiteULike*) e suas variações no protocolo de remoção de itens. Além disso, os algoritmos foram testados para diferentes dimensões como apresentadas na Seção 3.6 e organizamos os resultados na seguinte ordem: primeiro os resultados de precisão utilizando a medida MAP na Seção 4.1. Em seguida, na Seção 4.2, apresentamos os resultados de *Cobertura de Usuário* nos dois conjuntos de dados utilizados. Na Seção 4.3 exploramos os resultados de *Cobertura do Catálogo* para em seguida apresentar as medidas de *Similaridade* entre os algoritmos de recomendação na Seção 4.4. Na Seção 4.5 é apresentada uma comparação de resultados de MAP entre os algoritmos de recomendação SIP e AIP descritos na Seção 3.3 utilizando outra medida de similaridade, além de peso binário na representação BOW. Ao final deste capítulo, na Seção 4.6, é feita uma discussão geral sobre os resultados obtidos neste trabalho respondendo as questões de pesquisa apresentadas no Capítulo 1.

4.1 Resultados de Precisão das Previsões

No Capítulo 3 definimos a medida MAP como nossa principal medida diante do tipo de tarefa desempenhada por nossos recomendadores. Em se tratando de "recomendar bons itens", em um teste de sistema, acertar os itens escondidos do perfil do usuário, e nas primeiras colocações, demonstra a capacidade de um algoritmo de recomendação em aprender as preferências do usuário e se utilizar disto para fazer previsões. Essa é uma dimensão importante para um recomendador deste tipo e discutiremos os resultados dessa medida para cada conjunto de dados explorado em nossa pesquisa.

4.1.1 Precisão em MovieLens

Na Tabela 4.1 contém os resultados para os dois algoritmos de recomendação baseados em conteúdo textual: Filtragem por Agregação dos Itens do Perfil e Filtragem pela Similaridade dos Itens do Perfil. Os resultados são referentes ao conjunto de dados *MovieLens* e a tarefa empregada é para predizer somente 1 item removido do perfil do usuário (Given 1). O algoritmo que apresenta os melhores resultados é o SIP quando utilizados *Todos* os atributos concatenados. Entretanto, um resul-

tado chama atenção que é a concatenação dos atributos de Atores, Escritores e Diretores (*1+2+3*) que apresentam resultados de MAP próximos ao melhor resultado. Alguns atributos tiveram valores de precisão muito ruins, dentre eles *Gênero* com o pior resultado no algoritmo SIP e *Palavras chave* com o pior resultado no algoritmo AIP. O atributo *Sinopse* apresenta comportamento interessante quando analisada a precisão dos dois algoritmos de recomendação BC Textual: o algoritmo SIP apresenta resultados próximos aos melhores dessa configuração e resultados ruins, próximos aos piores resultados, para o algoritmo AIP. O atributo *Gênero* apresenta os piores resultados para os dois algoritmos, principalmente para o SIP. Os atributos *Revisão* e *Sinopse* tem resultados relevantes para o algoritmo SIP, mas apresentam resultados ruins para o algoritmo AIP.

	Given 1									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,0030	0,0003	0,0033	0,0006	0,0035	0,0009	0,0037	0,0012	0,0042	0,0015
Sinopse	0,0088	0,0006	0,0095	0,0012	0,0098	0,0015	0,0101	0,0018	0,0105	0,0022
Palavras	0,0051	0,0003	0,0056	0,0005	0,0058	0,0008	0,0061	0,0010	0,0064	0,0014
Gênero	0,0008	0,0013	0,0011	0,0015	0,0013	0,0017	0,0015	0,0020	0,0018	0,0024
Atores(1)	0,0089	**0,0019**	0,0094	0,0025	0,0096	0,0027	0,0099	0,0031	0,0104	0,0037
Escritores(2)	0,0056	0,0014	0,0066	0,0022	0,0070	0,0026	0,0074	0,0030	0,0079	0,0036
Diretores(3)	0,0038	**0,0019**	0,0045	0,0025	0,0049	0,0027	0,0054	0,0031	0,0061	0,0037
Revisão	0,0075	0,0011	0,0083	0,0018	0,0086	0,0021	0,0090	0,0025	0,0094	0,0029
1+2+3	0,0104	0,0015	0,0116	0,0026	0,0121	0,0032	0,0126	0,0039	0,0131	0,0045
Todos	**0,0116**	0,0017	**0,0125**	**0,0031**	**0,0130**	**0,0037**	**0,0135**	**0,0042**	**0,0139**	**0,0048**

Tab. 4.1: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o conjunto de dados *MovieLens* em Given 1 com Similaridade Cosseno.

Os resultados apresentados em Given 5 para o *MovieLens* seguem a mesma tendência do Given 1 como podemos observar na Tabela 4.2. Novamente *Todos* os atributos concatenados geram melhor MAP e próximo destes valores estão os valores da concatenação de *1+2+3*. Porém, um novo atributo também merece destaque: *Sinopse*. Nesta configuração do experimento este atributo mostra resultado superior ao de *1+2+3*, em TOP10, ultrapassando seu desempenho observado para Given 1.

Assim como em Given 1 os resultados obtidos em Given 10, apresentados na Tabela 4.3, confirmam que para o conjunto de dados *MovieLens* o melhor algoritmo entre as duas propostas apresentadas na Seção 3.3 é o SIP. Além disso, podemos inferir que pelos resultados apresentados a concatenação de todos os atributos seguido da concatenação *1+2+3* são os melhores atributos em se tratando de precisão das previsões. Em Given 10 o atributo *Sinopse* não mantém seu valor de MAP superior ao de *1+2+3* em MAP@10 como apresentado em Given 5 com relação a Given 1. Esta variação de resultados para *Sinopse* e *1+2+3* contrasta com o comportamento estável de *Todos* os atributos concatenados que sempre apresenta resultado superior aos demais. Outro comportamento perceptível nos resultados é o aumento do valor de MAP com o crescimento das listas. Porém, este crescimento não pode ser considerado muito significativo se verificarmos as diferenças de MAP entre os tamanhos das listas 10 e 20, 20 e 30, 30 e 50 e 50 e 100. Este comportamento está presente em todos os atributos testados e sugere que o algoritmo SIP, um dos representantes da abordagem BC,

concentra seus acertos no início da lista.

	Given 5									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,0032	0,0004	0,0036	0,0008	0,0037	0,0011	0,0039	0,0014	0,0045	0,0019
Sinopse	0,0114	0,0008	0,0122	0,0015	0,0126	0,0018	0,0130	0,0023	0,0135	0,0027
Palavras	0,0056	0,0003	0,0061	0,0005	0,0064	0,0008	0,0067	0,0011	0,0071	0,0016
Gênero	0,0022	0,0015	0,0026	0,0017	0,0028	0,0019	0,0031	0,0023	0,0035	0,0028
Atores(1)	0,0099	0,0020	0,0105	0,0025	0,0108	0,0029	0,0112	0,0034	0,0117	0,0041
Escritores(2)	0,0061	0,0016	0,0072	0,0023	0,0078	0,0027	0,0084	0,0032	0,0091	0,0039
Diretores(3)	0,0038	0,0020	0,0045	0,0025	0,0050	0,0029	0,0056	0,0034	0,0066	0,0041
Revisão	0,0098	0,0011	0,0107	0,0019	0,0111	0,0023	0,0115	0,0027	0,0120	0,0033
1+2+3	0,0109	0,0017	0,0123	0,0028	0,0129	0,0034	0,0135	0,0042	0,0143	0,0051
Todos	**0,0132**	**0,0022**	**0,0144**	**0,0036**	**0,0149**	**0,0043**	**0,0155**	**0,0051**	**0,0162**	**0,0060**

Tab. 4.2: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o conjunto de dados *MovieLens* em Given 5 com Similaridade Cosseno.

	Given 10									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,0023	0,0003	0,0027	0,0006	0,0029	0,0009	0,0031	0,0013	0,0038	0,0018
Sinopse	0,0107	0,0009	0,0114	0,0015	0,0118	0,0019	0,0123	0,0024	0,0129	0,0031
Palavras	0,0059	0,0002	0,0064	0,0004	0,0067	0,0007	0,0071	0,0011	0,0075	0,0015
Gênero	0,0015	0,0012	0,0019	0,0015	0,0020	0,0017	0,0023	0,0021	0,0028	0,0027
Atores(1)	0,0095	0,0018	0,0103	0,0024	0,0106	0,0027	0,0110	0,0033	0,0117	0,0042
Escritores(2)	0,0057	0,0014	0,0069	0,0022	0,0076	0,0027	0,0084	0,0032	0,0093	0,0041
Diretores(3)	0,0029	0,0018	0,0036	0,0024	0,0042	0,0027	0,0050	0,0033	0,0063	0,0042
Revisão	0,0086	0,0011	0,0095	0,0018	0,0100	0,0022	0,0105	0,0027	0,0111	0,0033
1+2+3	0,0110	0,0016	0,0127	0,0028	0,0135	0,0036	0,0144	0,0046	0,0154	0,0058
Todos	**0,0119**	**0,0021**	**0,0133**	**0,0035**	**0,0140**	**0,0043**	**0,0148**	**0,0054**	**0,0157**	**0,0067**

Tab. 4.3: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o conjunto de dados *MovieLens* em Given 10 com Similaridade Cosseno.

Na Tabela 4.4 estão os resultados de MAP para o algoritmo de Filtragem Colaborativa descrito na Seção 3.2 para o conjunto de dados *MovieLens*. Para qualquer das configurações exploradas, neste conjunto de dados, o resultado de MAP para Filtragem Colaborativa é superior. Outro comportamento perceptível é o maior taxa de crescimento do valor de MAP quando se aumenta o tamanho das listas de recomendação em relação aos algoritmos da abordagem BC. Esse comportamento de crescimento é melhor apresentado na Figura 4.1. Cada linha deste gráfico trata de um algoritmo em uma configuração do conjunto de dados *MovieLens*, por exemplo: G1SIP são os valores de MAP

em Given 1 para o algoritmo SIP. G10FC são os valores de MAP em Given 10 para o algoritmo de Filtragem Colaborativa e assim por diante. Outro comportamento marcante observado e que difere as duas abordagens, nas configurações do experimento para o protocolo de remoção de itens, é que o algoritmo de FC melhora a medida que possui mais itens a serem previstos. Ou seja, Given 5 apresenta resultados de MAP melhor que Given 1 e Given 10 apresenta resultados de MAP superiores aos encontrados em Given 5. Quando analisados os resultados dos algoritmos SIP e AIP se observa que Given 5 apresenta resultados de MAP maiores que Given 1. Porém, em Given 10 há uma queda nos resultados, principalmente no algoritmo SIP.

	Given 1	Given 5	Given 10
MAP@10	0,0230	0,0252	0,0253
MAP@20	0,0283	0,0335	0,0358
MAP@30	0,0308	0,0382	0,0423
MAP@50	0,0332	0,0438	0,0507
MAP@100	0,0353	0,0496	0,0606

Tab. 4.4: Valores de MAP obtidos por Filtragem Colaborativa para o Conjunto de Dados *MovieLens*.

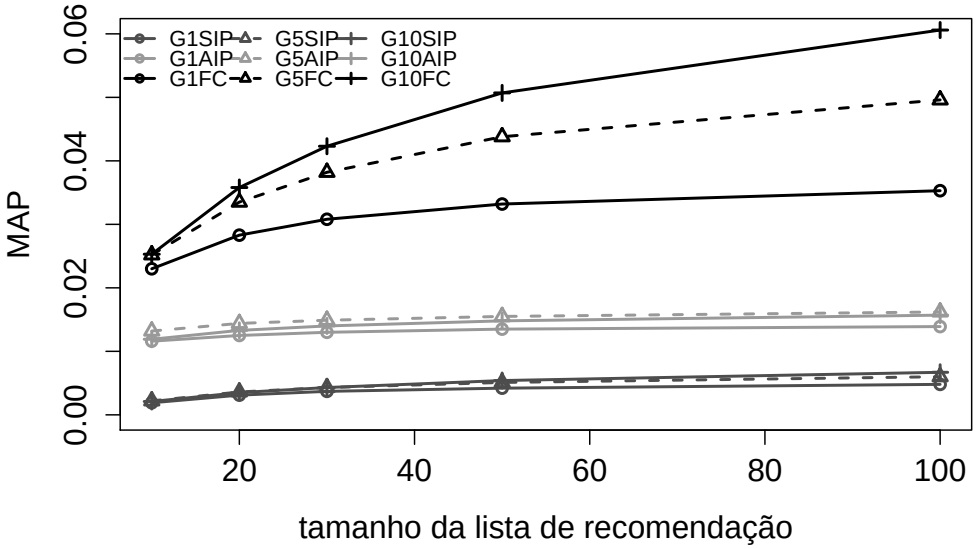

Fig. 4.1: Comportamento do MAP entre os melhores resultados dos algoritmos SIP, AIP (Todos os atributos) e FC nas três configurações do Conjunto de Dados *Movilens*.

A tarefa de recomendação de filmes constitui um desafio amplo para as pesquisas em Sistemas de Recomendação. Utilizando dois algoritmos de recomendação Baseados em Conteúdo Textual, por conta da natureza desta abordagem, se cria uma expectativa de que quanto maior o volume de dados disponíveis melhor o seu desempenho. Isso é comprovadamente apresentado nos resultados de MAP

para o atributo *Todos* que concatena todos os atributos recuperados do *IMDb*. Entretanto, os valores apresentados pelo atributo *Atores Principais* e a concatenação de *Atores+Escritores+Diretores* são resultados expressivos para essa abordagem. Definitivamente, os resultados demonstram que utilizando algoritmos de recomendação por conteúdo textual os atributos dos itens possuem relevância diferente. Os baixos resultados para o atributo *Gênero* é um exemplo. Um usuário não deve gostar de um determinado filme, exclusivamente, por que possui o mesmo gênero de outro bem avaliado anteriormente. Além disso, existem muitos filmes com o mesmo gênero tornando a tarefa de filtragem difícil por não possibilitar a distinção dos filmes. Contrária a essa situação dos resultados para *Gênero*, os bons resultados para o atributo *Atores Principais* sugere que muitos usuários apreciam diversos filmes estrelados pelos mesmos atores. Esses resultados de MAP apresentados pelo algoritmo SIP, apesar de inferiores aos apresentados por FC, começam a responder nossa primeira questão de pesquisa (**Q1**). Para recomendação de filmes, considerando as atividades dos usuários do conjunto de dados *MovieLens*, é possível realizar a tarefa de "recomendar bons itens" por meio de conteúdo textual. Porém, a escolha de qual conteúdo utilizar para realizar as recomendações é uma decisão importante no projeto do recomendador.

4.1.2 Precisão em CiteULike

Como apresentamos no Capítulo 3 o conjunto de dados do *CiteULike* apresenta características muito distintas em relação ao *MovieLens*. Neste trabalho nos propomos a fornecer como entrada para os mesmos algoritmos estes dois conjuntos de dados para observação de seu comportamento em cenários muito diferentes. Na Tabela 4.5 apresentamos os resultados para os algoritmos da abordagem BC textual SIP e AIP em Given 1.

	Given 1									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,1116	0,0542	0,1148	0,0601	0,1162	0,0623	0,1172	0,0637	0,1181	0,0646
Resumo	0,0851	0,0536	0,0865	0,0596	0,0871	0,0609	0,0881	0,0621	0,0898	0,0628
Autores	0,1109	0,0684	0,1119	0,0711	0,1124	0,0717	0,1129	0,0721	0,1133	0,0723
Publicação	0,0436	0,0393	0,0451	0,0419	0,0457	0,0424	0,0464	0,0430	0,0468	0,0434
Editor	0,0093	0,0074	0,0099	0,0082	0,0100	0,0084	0,0102	0,0086	0,0104	0,0088
Fonte	0,0706	0,0397	0,0725	0,0425	0,0732	0,0435	0,0739	0,0443	0,0745	0,0448
Tags	0,1693	**0,1367**	0,1729	**0,1411**	0,1741	**0,1423**	0,1748	**0,1432**	0,1753	**0,1437**
Todos	**0,1931**	0,1096	**0,1975**	0,1174	**0,1990**	0,1196	**0,2006**	0,1211	**0,2020**	0,1219

Tab. 4.5: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 1 com Similaridade Cosseno.

Os dois atributos que apresentam maior precisão para os recomendadores por conteúdo são *Tags* e *Todos*, sendo que neste caso *Tags*, especificamente, para o algoritmo AIP e *Todos* para o SIP. Porém, os demais campos também apresentam valores de MAP interessantes diante da característica desse conjunto de dados, no qual, muitos usuários possuem poucos itens em seu perfil. Os atributos *Título* e *Autores* com valores de MAP acima de $0,1$ para o algoritmo SIP seguidos dos atributos *Resumo* e

Fonte também merecem destaque para essa configuração do experimento. O destaque negativo é para o atributo *Editor* que apresenta resultados ruins para os dois algoritmos. Outro aspecto importante dos valores de MAP apresentados é que apesar do algoritmo SIP ainda apresentar melhor resultado de MAP com relação ao algoritmo AIP, essa diferença é bem inferior a apresentada para o conjunto de dados *MovieLens*. Entretanto, observamos esse comportamento somente nos dois atributos de melhor resultado de MAP, pois nos demais (exceto *Publicação*) o algoritmo SIP apresenta grande superioridade na escala de valores utilizada.

Na Tabela 4.6, para a configuração Given 5, novamente o atributo *Editor* tem o pior resultado de MAP e *Tags*, seguido de *Todos*, apresentam os melhores resultados de MAP. Além disso, observamos uma mudança entre Given 1 e Given 5 apresentado nesta tabela: o algoritmo AIP passa a apresentar os melhores resultados e o atributo *Tags* na melhor colocação com uma diferença considera relevante para os valores apresentados.

Na Seção 3.5 do Capítulo 3 apresentamos as configurações de cada um dos conjuntos de dados em Given 1, 5 e 10. Observa-se que a quantidade de usuários, no *CiteULike*, sofre grande redução entre essas configurações, principalmente entre Given 1 e Given 5. Este fato devida a grande presença de usuários com tamanho de perfil considerados pequenos (entre 2 e 10 itens avaliados positivamente). Associado a este cenário é perceptível a queda nos valores de MAP entre Given 1 e Given 5 o que sugere que os dois algoritmos BC Textual, SIP e AIP, possuem alta precisão de previsões para esses usuários nos quais pouco se conhece suas preferências.

	Given 5									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,0587	0,0167	0,0654	0,0322	0,0684	0,0405	0,0718	0,0469	0,0747	0,0508
Resumo	0,0294	0,0281	0,0335	0,0511	0,0347	0,0592	0,0364	0,0648	0,0425	0,0685
Autores	0,0565	0,0310	0,0622	0,0452	0,0650	0,0491	0,0667	0,0515	0,0683	0,0528
Publicação	0,0212	0,0301	0,0258	0,0359	0,0277	0,0379	0,0295	0,0399	0,0315	0,0415
Editor	0,0023	0,0051	0,0034	0,0059	0,0038	0,0063	0,0042	0,0068	0,0045	0,0072
Fonte	0,0450	0,0140	0,0510	0,0216	0,0533	0,0248	0,0549	0,0280	0,0568	0,0307
Tags	**0,1135**	**0,1172**	**0,1412**	**0,1502**	**0,1483**	**0,1593**	**0,1531**	**0,1641**	**0,1558**	**0,1666**
Todos	0,1063	0,0657	0,1293	0,1131	0,1365	0,1278	0,1418	0,1352	0,1475	0,1393

Tab. 4.6: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 5 com Similaridade Cosseno.

Nos valores de MAP obtidos em Given 10 e apresentados na Tabela 4.7 o comportamento é mantido em relação ao Given 5. Novamente os dois principais conteúdos são *Tags* e *Todos* e o algoritmo AIP, definitivamente, apresenta resultados superiores de MAP para este conjunto de dados. Novamente *Título* e *Resumo* também apresentam bons resultados de precisão, assim como *Publicação* e *Editor* com os piores resultados nesta dimensão de avaliação. Outro padrão de comportamento detectado é a diminuição dos valores de MAP em relação a Given 5 e que já ocorreu entre Given 1 e 5, mas em uma proporção menor. Essa menor diminuição pode ser justificada também pela menor diminuição da quantidade de usuário, mas reforça a ideia que os algoritmos SIP e AIP são mais precisos entre usuários que possuem menor quantidade de itens em seus perfis. Outra percepção desses resultados é

que na atividade de recuperar uma quantidade maior de itens relevantes, no conjunto de dados *CiteU-Like*, o algoritmo AIP apresenta melhores resultados. Para a tarefa de prever somente 1 único item o algoritmo SIP apresentou melhores resultados. E ainda um aspecto interessante é que em alguns dos demais campos (*Título*, *Autores* e *Fonte*) o comportamento é semelhante ao padrão encontrado nos resultados de MAP do *MovieLens*: SIP com resultado superior a AIP.

	Given 10									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,0466	0,0041	0,0573	0,0212	0,0629	0,0373	0,0680	0,0519	0,0732	0,0617
Resumo	0,0197	0,0135	0,0251	0,0506	0,0272	0,0701	0,0293	0,0849	0,0396	0,0936
Autores	0,0472	0,0161	0,0578	0,0319	0,0618	0,0438	0,0655	0,0510	0,0689	0,0544
Publicação	0,0171	0,0234	0,0244	0,0330	0,0286	0,0376	0,0328	0,0419	0,0361	0,0453
Editor	0,0026	0,0044	0,0040	0,0053	0,0045	0,0059	0,0051	0,0067	0,0058	0,0073
Fonte	0,0336	0,0063	0,0431	0,0155	0,0485	0,0225	0,0530	0,0292	0,0569	0,0347
Tags	**0,0796**	**0,0966**	**0,1284**	**0,1558**	**0,1509**	**0,1825**	**0,1629**	**0,1964**	**0,1694**	**0,2030**
Todos	0,0670	0,0471	0,1055	0,1093	0,1264	0,1519	0,1403	0,1714	0,1514	0,1819

Tab. 4.7: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 10 com Similaridade Cosseno.

Como verificamos ao final da Seção 3.4 o conjunto de dados *CiteULike* apresenta uma esparsidade muito elevada na relação $usuário \times item$ e esta característica é determinante na identificação dos vizinhos em um algoritmo do tipo kNN utilizado em Filtragem Colaborativa. Neste cenário são esperados os resultados inferiores de MAP para o algoritmo de abordagem social testado neste experimento como podemos analisar na Tabela 4.8.

	Given 1	Given 5	Given 10
MAP@10	0,0091	0,0080	0,0014
MAP@20	0,0095	0,0096	0,0020
MAP@30	0,0096	0,0100	0,0023
MAP@50	0,0097	0,0104	0,0026
MAP@100	0,0098	0,0108	0,0029

Tab. 4.8: Valores de MAP obtidos por Filtragem Colaborativa para o Conjunto de Dados *CiteULike*.

Em grande contraste ao que ocorreu nos resultados de MAP no conjunto de dados *MovieLens*, os valores apresentados para *CiteULike* demonstram a ineficiência de abordagens sociais em contextos como de uma biblioteca virtual ou um sistema de apontadores: grande quantidade de itens e baixíssima colaboração entre os usuários. A análise da Figura 4.2 reafirma a questão na qual neste cenário o uso exclusivo de técnicas sociais é de difícil aplicação, pois localizar usuários com interesses semelhantes (com as mesmas publicações) implicaria em isolar os usuários em grupos de co-citações restringindo muito as possibilidades de recuperação de novos itens. Outro ponto importante de discussão é sobre o atributo que apresentou os melhores resultados: *Tags*. Apesar deste resultado ter

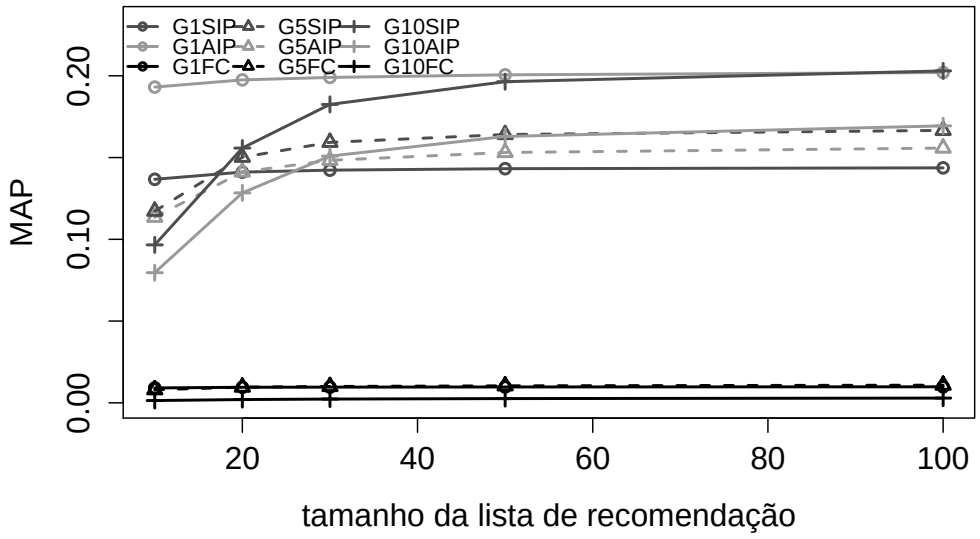

Fig. 4.2: Comportamento do MAP entre os melhores resultados dos algoritmos SIP, AIP (Todos os atributos) e FC nas três configurações do Conjunto de Dados *CiteULike*.

sido gerado por um algoritmo da abordagem por conteúdo, por princípio, este atributo é social. Como explicado no Capítulo 3 este atributo é recuperado sobre os apontadores que os usuários utilizam em suas bibliotecas e que é criado de forma colaborativa. Ou seja, ao informar um novo marcador para uma entrada de publicação em seu perfil o usuário é apresentado a sugestões (um Sistema de Recomendação) de possíveis marcadores. Desta maneira poderíamos considerar este resultado como fruto de um sistema híbrido, onde um algoritmo BC recebe como entrada dados de um sistema colaborativo.

4.2 Resultados de Cobertura dos Usuários

Em Sistemas de Recomendação uma das tarefas mais empregadas por estes programas são a apresentação de listas de itens que podem interessar ao usuário. Como anteriormente discutido, *Netflix* e *Amazon* são dos mais famosos representantes deste tipo de aplicação dos algoritmos de recomendação (não se limitando somente a este tipo de tarefa). Neste contexto acreditamos que avaliar a capacidade destes algoritmos em apresentar uma quantidade desejada de itens ao usuário é um aspecto importante de seu comportamento. Obviamente, em plataformas comerciais não seria uma opção deixar o usuário sem receber sugestões de itens interessantes, mesmo que pouco se soubesse sobre ele (*user cold-start*). E portanto, será avaliado o índice UCOV que indica a capacidade do sistema compor uma lista completa de recomendação para cada um dos usuários.

4.2.1 Cobertura dos Usuários em MovieLens

Apresentamos na Tabela 4.9 os resultados de UCOV para o *MovieLens* em Given 1. Com essa configuração de experimento os algoritmos de recomendação BC possuem ampla capacidade de realizar a tarefa sugerida. Exceto para o atributo *Título* que para alguma quantidade de usuários não consegue recuperar a quantidade desejada de itens para a lista de recomendação. Além disso, quando o tamanho solicitado da lista é 100 (TOP100) os algoritmos SIP e AIP começam a não atingir índice $1,0000$ para UCOV devida a grande quantidade de itens esperada pela medida. Outro aspecto relevante dessa medida de avaliação, como apresentado nas Tabelas 4.10 e 4.11, o comportamento é constante para qualquer das 3 quantidade de itens removidos do perfil do usuário ativo o que confere para a abordagem BC ótima capacidade de geração das listas de recomendação para qualquer dos atributos analisados.

	Given 1									
	UCOV@10		UCOV@20		UCOV@30		UCOV@50		UCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,9995	0,9997	0,9993	0,9994	0,9989	0,9989	0,9980	0,9976	0,9923	0,9927
Sinopse	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998	1,0000
Palavras	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998	1,0000
Gênero	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998	1,0000
Atores(1)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998	0,9996
Escritores(2)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998	0,9999
Diretores(3)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9994	0,9996
Revisão	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998	1,0000
1+2+3	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998	1,0000
Todos	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998	1,0000

Tab. 4.9: Valores de UCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 1 com Similaridade Cosseno.

UCOV também foi analisado para a Filtragem Colaborativa e o resultado apresentado na Tabela 4.12. Em nossa metodologia experimental, detalhada no Capítulo 3, discutimos sobre as configurações de filtro realizadas neste conjunto de dados e por quê este contexto se apresenta como favorável para qualquer das abordagens. Nesta dimensão todos os algoritmos apresentaram o resultado máximo para a medida devido a esparsidade considera pequena, na qual todos os itens possuem avaliação dos usuários e todos os usuários possuem uma quantidade relevante de itens em seus perfis. E mais precisamente sobre os resultados conferidos aos algoritmos SIP e AIP, quando analisamos o índice UCOV para cada um dos atributos podemos fazer relação aos valores de cobertura apresentados na Seção 3.4. Para o conjunto de dados *MovieLens* os atributos dos filmes tiveram grande cobertura na recuperação de dados realizada pelo *Web crawler*, ou seja, quase 100% dos filmes possuem todos os seus atributos com dados válidos o que possibilita aos recomendadores por conteúdo localizar itens semelhantes com relativa facilidade.

Nossa medida de UCOV visa identificar se a tarefa de recomendação testada está sendo executada pelo recomendador. De forma direta, medidas como MAP não podem indicar esta dimensão, pois é

	Given 5									
	UCOV@10		UCOV@20		UCOV@30		UCOV@50		UCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,9999	1,0000	0,9995	0,9997	0,9989	0,9991	0,9967	0,9971	0,9859	0,9879
Sinopse	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Palavras	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Gênero	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Atores(1)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9998
Escritores(2)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Diretores(3)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9996	0,9998
Revisão	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
1+2+3	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Todos	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

Tab. 4.10: Valores de UCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 5 com Similaridade Cosseno.

	Given 10									
	UCOV@10		UCOV@20		UCOV@30		UCOV@50		UCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	1,0000	1,0000	1,0000	1,0000	0,9999	1,0000	0,9993	0,9998	0,9947	0,9959
Sinopse	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Palavras	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Gênero	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Atores(1)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Escritores(2)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Diretores(3)	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Revisão	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
1+2+3	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Todos	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

Tab. 4.11: Valores de UCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 10 com Similaridade Cosseno.

possível obter valores de MAP superiores em TOP10 apresentando listas menores que 10. E encontrar valores ruins de MAP em listas de 10 itens em TOP10. Em uma aplicação real seria indesejado o usuário esperando navegar em uma lista de 10 ou 20 itens e o recomendador apresentar somente 4 ou 5 itens. Este cenário pode empobrecer a experiência do usuário e tornar o uso do sistema desinteressante ao longo do tempo.

	Given 1	Given 5	Given 10
UCOV@10	1,0000	1,0000	1,0000
UCOV@20	1,0000	1,0000	1,0000
UCOV@30	1,0000	1,0000	1,0000
UCOV@50	1,0000	1,0000	1,0000
UCOV@100	1,0000	1,0000	1,0000

Tab. 4.12: Valores de UCOV obtidos por Filtragem Colaborativa para o Conjunto de Dados *MovieLens*.

4.2.2 Cobertura dos Usuários em CiteULike

Aplicamos a mesma medida proposta de *User Coverage* no conjunto de dados *CiteULike* como podemos verificar nas Tabelas 4.13 , 4.14 e 4.15. Para Given 1 os melhores resultados do índice UCOV são atribuídos para os atributos *Autores*, *Tags* e *Todos* com pequena queda para as listas de recomendação de tamanho 50 e 100 (TOP50 e TOP100). Os valores apresentados, assim como nas configurações do conjunto de dados *MovieLens*, são valores próximos ao máximo de $1,0000$. Porém, é possível observar um conjunto maior de resultados diferentes do máximo valor possível o que sugere, neste contexto, que a tarefa começa a ficar mais difícil para os algoritmos por conteúdo. Este comportamento deve ser justificado pela menor cobertura dos atributos na recuperação de dados realizada pelo *Web crawler*. Verificamos na Seção 3.4 que atributos como *Fonte* estão presentes em somente 38,50% dos itens do sistema. A falta de dados para a realização dos cálculos de similaridade entre os itens dificulta a tarefa de filtragem de dados realizada pelos algoritmos de recomendação BC Textual.

	Given 1									
	UCOV@10		UCOV@20		UCOV@30		UCOV@50		UCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,9990	0,9993	0,9983	0,9985	0,9971	0,9975	0,9890	0,9950	0,8379	0,9884
Resumo	0,9997	0,9997	0,9997	0,9997	0,9997	0,9997	0,9931	0,9997	0,8211	0,9997
Autores	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9941	1,0000	0,8352	1,0000
Publicação	0,8602	0,9798	0,8481	0,9657	0,8382	0,9542	0,8196	0,9376	0,6579	0,9082
Editor	0,6650	0,9914	0,6626	0,9881	0,6597	0,9850	0,6507	0,9804	0,5015	0,9719
Fonte	0,9719	0,9831	0,9718	0,9774	0,9718	0,9755	0,9660	0,9740	0,8132	0,9728
Tags	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9927	1,0000	0,7111	1,0000
Todos	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9942	1,0000	0,8356	1,0000

Tab. 4.13: Valores de UCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 1 com Similaridade Cosseno.

Os resultados apresentados para Given 5 e Given 10 sugerem uma melhora no índice de UCOV provavelmente influenciada pela remoção dos usuários com poucos itens do perfil. Embora quando se analisa a precisão dos recomendadores a presença desses usuários eleva os valores de MAP, em contrapartida, significa uma redução da cobertura. Entendemos que criar listas para uma tarefa TOP50

ou TOP100 é bastante difícil quando se possui poucos itens avaliados no perfil. Ainda para essas duas configurações os atributos *Publicação*, *Editor* e *Fonte* apresentam os piores resultados com uma quantidade maior de usuários que não tiveram listas totalmente preenchidas.

	Given 5									
	UCOV@10		UCOV@20		UCOV@30		UCOV@50		UCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9999	1,0000
Resumo	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9994	1,0000	0,9829	1,0000
Autores	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9951	1,0000
Publicação	0,9623	0,9974	0,9570	0,9920	0,9540	0,9879	0,9493	0,9828	0,9018	0,9744
Editor	0,8594	0,9948	0,8574	0,9925	0,8560	0,9908	0,8516	0,9885	0,7692	0,9843
Fonte	0,9951	0,9951	0,9951	0,9951	0,9951	0,9951	0,9950	0,9951	0,9905	0,9951
Tags	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9993	1,0000	0,8591	1,0000
Todos	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9917	1,0000

Tab. 4.14: Valores de UCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 5 com Similaridade Cosseno.

	Given 10									
	UCOV@10		UCOV@20		UCOV@30		UCOV@50		UCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9995	1,0000
Resumo	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9995	1,0000	0,9837	1,0000
Autores	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9952	1,0000
Publicação	0,9868	1,0000	0,9868	1,0000	0,9868	1,0000	0,9860	0,9991	0,9490	0,9937
Editor	0,8895	0,9953	0,8868	0,9933	0,8851	0,9911	0,8823	0,9894	0,8227	0,9881
Fonte	0,9947	0,9947	0,9947	0,9947	0,9947	0,9947	0,9947	0,9947	0,9918	0,9947
Tags	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9996	1,0000	0,8703	1,0000
Todos	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9864	1,0000

Tab. 4.15: Valores de UCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 10 com Similaridade Cosseno.

Na Tabela 4.16 estão os resultados de UCOV para Filtragem Colaborativa. Assim como em MAP, UCOV também é prejudicada na abordagem social por conta da grande esparsidade de dados na matriz de usuários por itens. Devida a esta característica encontrada no conjunto de dados *CiteULike* a etapa de identificação dos vizinhos do algoritmo de FC não é capaz de recuperar muitos outros usuário (em muitos casos nem ao menos 1 vizinho) e por isso recuperar itens para realizar a tarefa de recomendação testada. Este comportamento observado nesta dimensão indica que mesmo considerando um tamanho de lista pequeno ($N = 10$) FC não é capaz de recuperar uma quantidade satisfatória de itens interessantes para os usuários.

	Given 1	Given 5	Given 10
UCOV@10	0,1864	0,2532	0,2474
UCOV@20	0,1640	0,2395	0,2474
UCOV@30	0,1475	0,2243	0,2398
UCOV@50	0,1253	0,2006	0,2184
UCOV@100	0,0921	0,1560	0,1754

Tab. 4.16: Valores de UCOV obtidos por Filtragem Colaborativa para o Conjunto de Dados *CiteU-Like*.

O conjunto de dados *CiteULike* utilizado em nosso trabalho é uma amostra que visa representar ao máximo o mundo real. Quando utilizamos o protocolo Given 1 existem usuários com somente 1, 2 ou 3 itens no perfil dos usuários testados (ver Figuras 3.11, 3.12 e 3.13 da Seção 3.4 com a distribuição dos usuários por quantidade de itens no perfil). Neste cenário a tarefa de "recomendar bons itens" se torna muito difícil, principalmente para abordagens sociais quando o sistema apresenta um forte *user cold-start*, no qual, pouco se conhece sobre as preferências dos usuários. Além disso, o sistema *CiteULike* apresenta também grande *item cold-start* devida a grande esparsidade apresentada na relação $usuário \times item$. Essa circunstância é muito desfavorável para a abordagem social, mas parece indiferente para os algoritmos que utilizam o conteúdo textual dos itens para realizar as recomendações devido aos resultados apresentados por eles para MAP e UCOV. Os altos valores de UCOV e MAP para este conjunto de dados confere à abordagem BC Textual um ótimo desempenho nesta tarefa em grande contraste aos resultados do algoritmo de Filtragem Colaborativa.

4.3 Resultados de Cobertura do Catálogo

Propomos em nossa metodologia experimental duas dimensões para *Coverage* e a segunda, *Catalog Coverage*, visa observar a capacidade dos algoritmos em recuperar todo o catálogo (todos os itens cadastrados no sistema) entre todas consultas realizadas pelo sistema. Considerando um Sistema de Recomendação que visa ajudar os usuários no problema da sobrecarga de informações um algoritmo bem avaliado em um teste de sistema teria *Precision* (capacidade de prever novas preferências) e *Coverage* (capacidade de recuperar itens poucos avaliados) com os maiores valores possíveis.

4.3.1 Cobertura do Catálogo para MovieLens

Na Tabela 4.17 apresentamos os primeiros resultados para CCOV com os dois algoritmos por conteúdo propostos com protocolo Given 1. Nesta configuração do experimento o algoritmo AIP apresenta uma melhor cobertura do catálogo utilizando o atributo *Palavras chave* em relação ao algoritmo SIP que apresentou melhor cobertura com o atributo *Atores+Escritores+Diretores*. A superioridade do algoritmo AIP neste medida de avaliação, na configuração analisada, ocorre em todos os tamanhos de listas que foram testadas. Um primeiro aspecto desse resultado é que desde o protocolo Given 1 até Given 10 (apresentados adiante) veremos que aumentar o tamanho das listas de recomendação (de TOP10 até TOP100) não representa mudança significativa nos valores de CCOV do modo definido neste trabalho. Nos resultados para o atributo *Título* com o algoritmo SIP o valor

de CCOV é mantido em $0,6625$ e somente em listas de recomendação de tamanho 100 (TOP100) o valor muda para $0,6626$. Este comportamento sugere que somente as primeiras posições das listas de recomendação são as responsáveis por cobrir o catálogo e garantir a máxima quantidade de itens que o recomendador é capaz de recuperar do sistema. Os piores resultados, tanto para SIP quanto para AIP, ficam atribuídos para o atributo *Gênero*. Para o algoritmo AIP o desempenho é bastante ruim se comparado com os demais atributos e comparando com os valores de precisão obtidos com este atributo verificamos que, definitivamente, este atributo tem pouca relevância quando utilizado exclusivamente.

	Given 1									
	CCOV@10		CCOV@20		CCOV@30		CCOV@50		CCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,6625	0,8871	0,6625	0,8871	0,6625	0,8872	0,6625	0,8872	0,6626	0,8872
Sinopse	0,7203	0,9150	0,7203	0,9150	0,7203	0,9150	0,7204	0,9151	0,7205	0,9152
Palavras	0,7935	**0,9549**	0,7935	**0,9549**	0,7935	**0,9550**	0,7935	**0,9550**	0,7936	**0,9550**
Gênero	0,5354	0,2822	0,5354	0,2822	0,5354	0,2822	0,5354	0,2822	0,5357	0,2812
Atores(1)	0,7616	0,8656	0,7616	0,8656	0,7617	0,8656	0,7618	0,8656	0,7620	0,8656
Escritores(2)	0,7360	0,8849	0,7360	0,8850	0,7360	0,8850	0,7360	0,8850	0,7361	0,8850
Diretores(3)	0,6957	0,8656	0,6957	0,8656	0,6957	0,8656	0,6957	0,8656	0,6961	0,8656
Revisão	0,6471	0,7363	0,6471	0,7363	0,6471	0,7363	0,6471	0,7363	0,6471	0,7365
1+2+3	**0,8048**	0,9391	**0,8048**	0,9391	**0,8049**	0,9391	**0,8049**	0,9391	**0,8052**	0,9392
Todos	0,7277	0,8513	0,7277	0,8513	0,7277	0,8513	0,7277	0,8513	0,7277	0,8514

Tab. 4.17: Valores de CCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 1 com Similaridade Cosseno.

Nas Tabelas 4.18 e 4.19 apresentamos os resultados de CCOV para *MovieLens* em Given 5 e Given 10. Assim como em Given 1 o algoritmo AIP apresenta os melhores resultados e o atributo de *Palavras chave* com o melhor resultado de CCOV nos algoritmos de BC textual seguido da combinação dos atributos *Atores+Escritores+Diretores*. Outro atributo com resultado também expressivo é *Sinopse* e, como observado neste atributo, o crescimento das listas de recomendação continua não representando um ganho nesta dimensão (uma pequena mudança já em TOP100). Observando os resultados entre essas três configurações do conjunto de dados *MovieLens* a pequena diferença entre quantidade de usuários pouco influência nos resultados de CCOV. Como visto em Given 5 ocorreu uma pequena melhora em CCOV para o algoritmo SIP com atributo *1+2+3*. Desta forma concluímos que para essa medida aumentar o tamanho da lista de recomendação ou mesmo remover alguns poucos usuários tem pouca influência no resultado.

Na Tabela 4.20 estão os valores obtidos de CCOV para Filtragem Colaborativa. Observamos que, apesar desse algoritmo apresentar maior precisão nas previsões, possui baixo desempenho com relação aos resultados encontrados pelos algoritmos BC nesta dimensão de avaliação. A concentração das abordagens sociais em itens populares favorece sua maior precisão, mas também limita sua filtragem sobre os itens que foram bem avaliados por um conjunto maior de usuários.

No trabalho de Salter & Antonopoulos (2006) *Catalog coverage* é utilizada como uma medida para avaliar *Diversity* do recomendador. Esta dimensão analisa a capacidade de diversificar as reco-

	Given 5									
	CCOV@10		CCOV@20		CCOV@30		CCOV@50		CCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,6727	0,8884	0,6727	0,8884	0,6727	0,8884	0,6727	0,8884	0,6729	0,8885
Sinopse	0,7441	0,9284	0,7441	0,9284	0,7441	0,9284	0,7441	0,9284	0,7444	0,9286
Palavras	0,8129	**0,9554**	0,8129	**0,9554**	0,8129	**0,9554**	0,8130	**0,9554**	0,8133	**0,9555**
Gênero	0,5566	0,3246	0,5566	0,3246	0,5566	0,3246	0,5566	0,3246	0,5572	0,3257
Atores(1)	0,7749	0,8696	0,7749	0,8696	0,7749	0,8696	0,7749	0,8696	0,7751	0,8697
Escritores(2)	0,7578	0,8893	0,7578	0,8893	0,7578	0,8893	0,7578	0,8893	0,7581	0,8894
Diretores(3)	0,7189	0,8696	0,7190	0,8696	0,7190	0,8696	0,7190	0,8696	0,7196	0,8697
Revisão	0,6624	0,7722	0,6624	0,7722	0,6624	0,7723	0,6624	0,7723	0,6626	0,7726
1+2+3	**0,8209**	0,9393	**0,8209**	0,9393	**0,8209**	0,9393	**0,8209**	0,9393	**0,8210**	0,9394
Todos	0,7415	0,8733	0,7415	0,8733	0,7415	0,8733	0,7415	0,8734	0,7416	0,8736

Tab. 4.18: Valores de CCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 5 com Similaridade Cosseno.

	Given 10									
	CCOV@10		CCOV@20		CCOV@30		CCOV@50		CCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,6531	0,8808	0,6531	0,8808	0,6531	0,8809	0,6531	0,8809	0,6532	0,8809
Sinopse	0,6896	0,8901	0,6896	0,8902	0,6896	0,8902	0,6897	0,8903	0,6898	0,8903
Palavras	0,7730	**0,9457**	0,7731	**0,9457**	0,7731	**0,9457**	0,7731	**0,9457**	0,7732	**0,9458**
Gênero	0,5136	0,2562	0,5136	0,2558	0,5136	0,2558	0,5136	0,2558	0,5136	0,2528
Atores(1)	0,7435	0,8494	0,7435	0,8495	0,7435	0,8495	0,7436	0,8495	0,7440	0,8496
Escritores(2)	0,7108	0,8713	0,7109	0,8713	0,7109	0,8714	0,7109	0,8714	0,7112	0,8716
Diretores(3)	0,6714	0,8494	0,6714	0,8495	0,6715	0,8495	0,6715	0,8495	0,6718	0,8496
Revisão	0,6376	0,6985	0,6376	0,6985	0,6376	0,6985	0,6376	0,6985	0,6377	0,6986
1+2+3	**0,7825**	0,9211	**0,7825**	0,9211	**0,7825**	0,9211	**0,7826**	0,9212	**0,7829**	0,9214
Todos	0,7133	0,8216	0,7134	0,8216	0,7134	0,8216	0,7134	0,8217	0,7135	0,8218

Tab. 4.19: Valores de CCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 10 com Similaridade Cosseno.

mendações, ou seja, recuperar diferentes itens para os usuários em uma tarefa de "recomendar bons itens". Entretanto, não consideramos apropriado analisar os resultados de CCOV obtidos nesta seção desta forma. A medida de CCOV proposta em Ge et al. (2010) contabiliza um conjunto de consultas realizadas para o mesmo usuário e nossa medida contabiliza somente 1 consulta realizada para cada usuário ativo do teste. Ainda assim, é esperado uma queda de precisão com o aumento da cobertura (ou diversidade) o que constitui em um dos problemas das técnicas sociais como apontado no trabalho de Lathia et al. (2010). Localizar a harmonia entre essas duas medidas é um problema de pesquisa não abordado em nosso trabalho, mas os resultados apontados no conjunto de dados *MovieLens* apontam para esta questão. Mesmo concentrando esta análise somente aos dois algoritmos por conteúdo

	Given 1	Given 5	Given 10
CCOV@10	0,3269	0,3233	0,3016
CCOV@20	0,3269	0,3233	0,3016
CCOV@30	0,3269	0,3233	0,3016
CCOV@50	0,3267	0,3233	0,3015
CCOV@100	0,3268	0,3231	0,3013

Tab. 4.20: Valores de CCOV obtidos por Filtragem Colaborativa para o Conjunto de Dados *MovieLens*.

textual (AIP e SIP) o algoritmo que apresentou melhor precisão (SIP) teve pior desempenho quando avaliado CCOV reafirmando este *trade-off*.

4.3.2 Cobertura do Catálogo para CiteULike

Na Tabela 4.21 apresentamos os resultados de CCOV para o conjunto de dados *CiteULike* em Given 1. Nesta dimensão o algoritmo AIP tem o melhor desempenho em relação ao algoritmo SIP e o atributo *Título* e a concatenação de *Todos* os atributos apresentam os melhores resultados. Os piores resultados atribuímos ao atributo *Editor*. Novamente, acreditamos que a baixa quantidade de itens que possuem este atributo (cerca de 38%) prejudica não somente a precisão de um recomendador baseado neste conteúdo, como também outras medidas como CCOV. Acompanhando *Editor* atributos como *Tags* e *Publicação* também apresentam resultados ruins. Especialmente o atributo *Tags* que já apresentou bons resultados de precisão se observa uma queda de mais de 53% para SIP e 28% para AIP em CCOV com relação aos melhores resultados.

	Given 1									
	CCOV@10		CCOV@20		CCOV@30		CCOV@50		CCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0,4202	**0,5013**	0,4202	**0,5014**	0,4203	**0,5014**	0,4202	**0,5016**	0,4206	**0,5021**
Resumo	0,3377	0,4500	0,3376	0,4501	0,3375	0,4503	0,3374	0,4505	0,3374	0,4510
Autores	0,2993	0,4151	0,2993	0,4152	0,2994	0,4152	0,2993	0,4153	0,2994	0,4156
Publicação	0,2716	0,3538	0,2716	0,3539	0,2717	0,3539	0,2716	0,3539	0,2718	0,3540
Editor	0,0972	0,1694	0,0973	0,1695	0,0973	0,1696	0,0972	0,1696	0,0973	0,1697
Fonte	0,3773	0,4686	0,3774	0,4687	0,3775	0,4688	0,3774	0,4690	0,3778	0,4695
Tags	0,1981	0,3569	0,1982	0,3570	0,1982	0,3571	0,1981	0,3572	0,1982	0,3574
Todos	**0,4231**	0,5010	**0,4232**	0,5012	**0,4233**	0,5013	**0,4232**	0,5015	**0,4237**	0,5019

Tab. 4.21: Valores de CCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 1 com Similaridade Cosseno.

Nas Tabelas 4.22 e 4.23 apresentamos os resultados em Given 5 e Given 10 para os algoritmos BC textual utilizando como entrada os dados do *CiteULike*. Em Given 5 o atributo *Título* apresenta o melhor resultado para qualquer dos dois algoritmos em contraste aos resultados de Given 1, no

qual a concatenação de *Todos* os atributos que apresentou o melhor resultado para SIP. Outro aspecto constante nestes resultados de CCOV é que o aumento de tamanho das listas de recomendação não constituem em acentuado crescimento dos resultados. E em alguns dos atributos, por exemplo *Publicação* com AIP, é possível perceber o resultado aproximadamente constante entre algumas variações de tamanho de listas de recomendação. Além disso, observamos que em Given 5 há uma queda nos resultados (de 45% no melhor resultado de SIP e 50% no melhor resultado de AIP) que acreditamos está associada a redução da quantidade de usuários de teste. Consequentemente, há menos listas de recomendação para realizar o calculo de cobertura do catálogo.

	Given 5									
	CCOV@10		CCOV@20		CCOV@30		CCOV@50		CCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	**0,2311**	**0,2502**	**0,2313**	**0,2504**	**0,2314**	**0,2506**	**0,2318**	**0,2511**	**0,2327**	**0,2523**
Resumo	0,1655	0,2437	0,1655	0,2439	0,1655	0,2442	0,1655	0,2447	0,1663	0,2459
Autores	0,1635	0,2171	0,1635	0,2174	0,1636	0,2175	0,1638	0,2178	0,1647	0,2187
Publicação	0,1664	0,2071	0,1665	0,2073	0,1666	0,2073	0,1666	0,2074	0,1672	0,2079
Editor	0,0783	0,1271	0,0783	0,1273	0,0784	0,1274	0,0784	0,1276	0,0784	0,1285
Fonte	0,2096	0,2379	0,2097	0,2382	0,2099	0,2383	0,2102	0,2388	0,2112	0,2397
Tags	0,1411	0,2174	0,1413	0,2177	0,1415	0,2179	0,1417	0,2181	0,1421	0,2188
Todos	0,2281	0,2404	0,2283	0,2406	0,2285	0,2408	0,2288	0,2413	0,2295	0,2423

Tab. 4.22: Valores de CCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 5 com Similaridade Cosseno.

	Given 10									
	CCOV@10		CCOV@20		CCOV@30		CCOV@50		CCOV@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	**0,1531**	**0,1662**	**0,1535**	**0,1666**	**0,1539**	**0,1671**	**0,1545**	**0,1678**	**0,1563**	**0,1696**
Resumo	0,1007	0,1623	0,1005	0,1627	0,1006	0,1631	0,1007	0,1637	0,1025	0,1654
Autores	0,1181	0,1512	0,1183	0,1517	0,1184	0,1520	0,1188	0,1526	0,1199	0,1539
Publicação	0,1157	0,1409	0,1159	0,1412	0,1161	0,1414	0,1166	0,1419	0,1177	0,1433
Editor	0,0690	0,0999	0,0690	0,1000	0,0690	0,1001	0,0690	0,1002	0,0693	0,1004
Fonte	0,1416	0,1578	0,1418	0,1582	0,1421	0,1585	0,1426	0,1591	0,1441	0,1606
Tags	0,1025	0,1548	0,1027	0,1551	0,1030	0,1554	0,1035	0,1560	0,1047	0,1574
Todos	0,1521	0,1641	0,1523	0,1644	0,1525	0,1647	0,1531	0,1653	0,1546	0,1666

Tab. 4.23: Valores de CCOV obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 10 com Similaridade Cosseno.

Assim como os valores obtidos no conjunto de dados *MovieLens* para o *CiteULike* detectamos uma menor capacidade de cobrir o catalogo no algoritmo de Filtragem Colaborativa. Neste conjunto de dados um algoritmo puramente social é pouco capaz de realizar a tarefa de "recomendar bons itens". Como discutido desde os resultados de precisão a grande esparsidade de dados devida a baixa

colaboração entre os usuários, neste tipo de sistema, torna muito difícil a localização de usuários com preferências semelhantes. Dessa forma, encontrar vizinhos que podem contribuir com a filtragem de itens interessantes para um usuário é muito difícil para esse tipo de algoritmo. Logo, desde a nossa primeira medida avaliada, MAP, é possível confirmar que algoritmos exclusivamente sociais não constituem de uma boa solução para esta configuração de conjunto de dados.

	Given 1	Given 5	Given 10
CCOV@10	0,0618	0,0408	0,0277
CCOV@20	0,0616	0,0409	0,0281
CCOV@30	0,0618	0,0407	0,0280
CCOV@50	0,0618	0,0410	0,0284
CCOV@100	0,0618	0,0414	0,0287

Tab. 4.24: Valores de CCOV obtidos por Filtragem Colaborativa para o Conjunto de Dados *CiteULike*.

Na primeira discussão sobre CCOV alertamos para o *trade-off* entre *Precision* e *Coverage*. A análise dos resultados para o conjunto de dados *CiteULike* deve seguir outro caminho. Apesar dos algoritmos BC apresentarem altos valores para MAP o que poderia sugerir a queda de resultado desta abordagem para CCOV, aqui cabe outro tipo de interpretação: a quantidade de itens do sistema é superior ao encontrado nos dados do *MovieLens* e com um catálogo muito maior é esperada uma queda na nossa proposta de *Catalog coverage*. Além disso, a quantidade de listas de recomendação geradas no *CiteULike* é inferior a quantidade de listas geradas para o *MovieLens* devida a quantidade de usuários disponíveis para teste. Este aspecto tem forte influência na nossa medida que é uma adaptação da proposta de Ge et al. (2010) para *Catalog coverage* em uma tarefa do tipo "recomendar bons itens". Ao considerarmos somente uma lista de recomendação para cada usuário de teste e o conjunto de listas para todos os usuários como denominador deste índice (ver detalhes na Seção 3.6) permitimos que este fator influencie nos resultados. Mesmo assim poderíamos considerar os resultados de CCOV para os dois algoritmos SIP e AIP como resultados expressivos se fosse possível comparar os resultados dos dois conjuntos de dados para essa medida.

4.4 Resultados de Similaridade entre Listas de Recomendação

Como vimos na Seção 2.7 do Capítulo 2 em trabalhos como o de Sorensen et al. (1997) quando algoritmos de recomendação são testados em modo *offline* as comparações são realizadas, normalmente, em termos de *Precisão* e *Revocação* como sistemas de Recuperação de Informação. Porém, consideramos essa abordagem limitada por não avaliar a possibilidade complementar das abordagens de recomendação. Em aplicações comerciais de Sistemas de Recomendação é comum identificar diversas abordagens convivendo e complementando as deficiências umas das outras. Por este motivo acreditamos que explorar a similaridade entre os resultados dos algoritmos pode fornecer uma nova visão sobre as abordagens e adicionar novas questões sobre as diferenças entre a Filtragem Colaborativa e a Baseada em Conteúdo.

4.4.1 Similaridade para MovieLens

Os resultados dos cálculos de similaridade obtidos foram plotados em gráficos de barras. A comparação foi realizada para a concatenação de *Todos* os atributos recuperados pelo *Web crawler* construído em nosso experimento. E, Cada barra trata da comparação entre dois recomendadores identificados na legenda do gráfico. No eixo X de cada gráfico é identificado quais são os dois algoritmos que estão sendo comparados e para qual configuração do protocolo de retirada de itens do perfil ocorre a comparação (Given 1, 5 e 10). A legenda é lida da seguinte forma: SA1 é a comparação do algoritmo SIP com o algoritmo AIP em Given 1. SF1 o algoritmo SIP com o algoritmo de FC em Given 1 e AF1 o algoritmo AIP e FC também em Given 1. As demais legendas são análogas a esta definição. O eixo Y indica qual o resultado do calculo de similaridade (*Jaccard*) entre as listas de recomendação geradas por cada um dos dois algoritmos testados. Cada barra do gráfico indica o grau de similaridade para um determinado tamanho de lista. Comparamos os algoritmos em diferentes tamanhos de listas: 10, 20, 30, 50 e 100 que denominamos TOP10, TOP20, TOP30, TOP50 e TOP100.

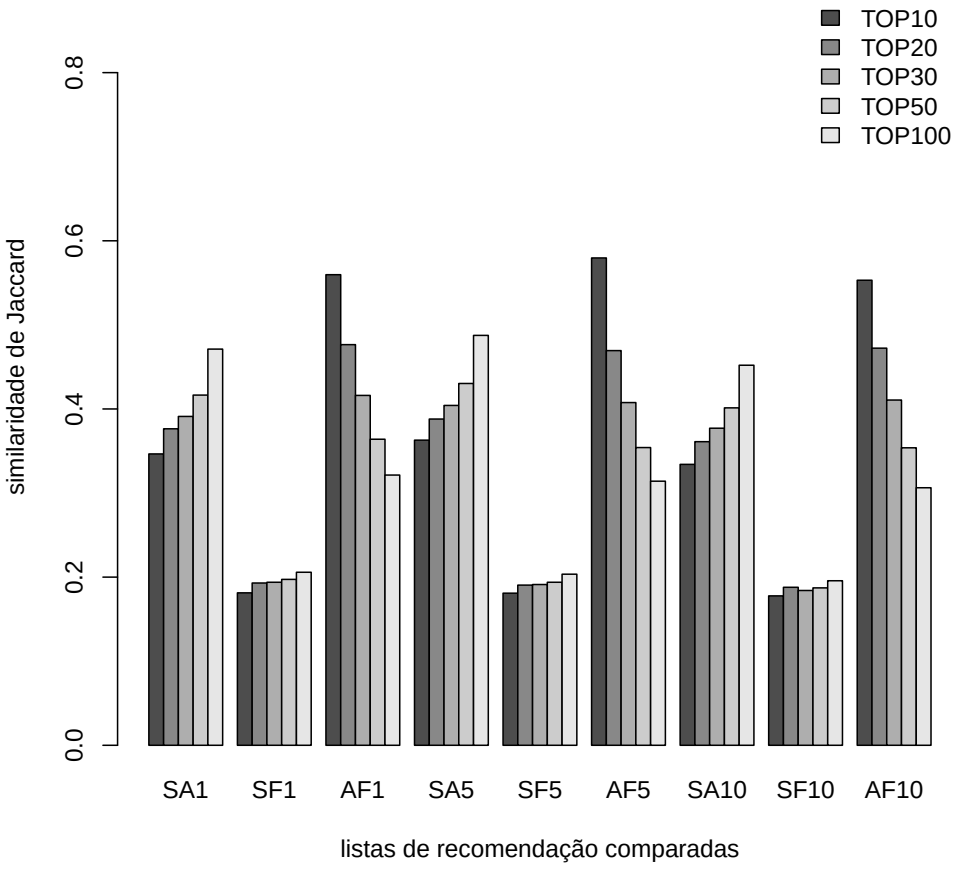

Fig. 4.3: Similaridade entre as recomendações geradas com o conjunto de dados *MovieLens* para Todos os atributos do Filme.

Como esperado, os dois algoritmos de recomendação BC apresentam grande grau de similaridade entre as listas de recomendação geradas por cada um. Além disso, observando os resultados de precisão (MAP) superiores para o algoritmo SIP em relação ao algoritmo AIP, no conjunto de dados *MovieLens*, se esperava uma similaridade superior deste algoritmo na comparação dos algoritmos BC Textual em relação a Filtragem Colaborativa. Outro ponto é sobre a similaridade entre AIPxFC que é maior em listas menores. Comportamento esse oposto a similaridade SIPxAIP que cresce a medida que as listas de recomendação crescem de tamanho.

Verificamos que há similaridade entre as listas de recomendação geradas pelos três algoritmos testados neste trabalho. Diante deste cenário um outro ângulo precisa ser inspecionado: os algoritmos acertam as mesmas previsões? Para iniciar esta análise são apresentados nas Figuras 4.4, 4.5 e 4.6 os resultados de *Quantidade de Acertos por Posição* na lista de recomendação. Estes gráficos demonstram em qual posição das listas de recomendação os algoritmos acertam considerando todos os usuários de teste. Podemos destacar destes gráficos é que notoriamente a Filtragem Colaborativa possui maior capacidade de acerto das previsões, pois mantem suas posições de acerto sempre em superioridade sobre qualquer dos dois algoritmos BC Textual. Além disso, a FC tem grande concentração de acertos nas primeiras posições da lista com uma queda desses acertos ao longo do crescimento das listas de recomendação. Outro comportamento importante é que o algoritmo SIP apresenta maior valor de MAP que o AIP e os três gráficos justificam esse resultado: esse algoritmo possui uma boa taxa de acerto logo no início das suas listas de recomendação. Ao final das listas seu desempenho é muito semelhante ao algoritmo AIP em termos de precisão das previsões.

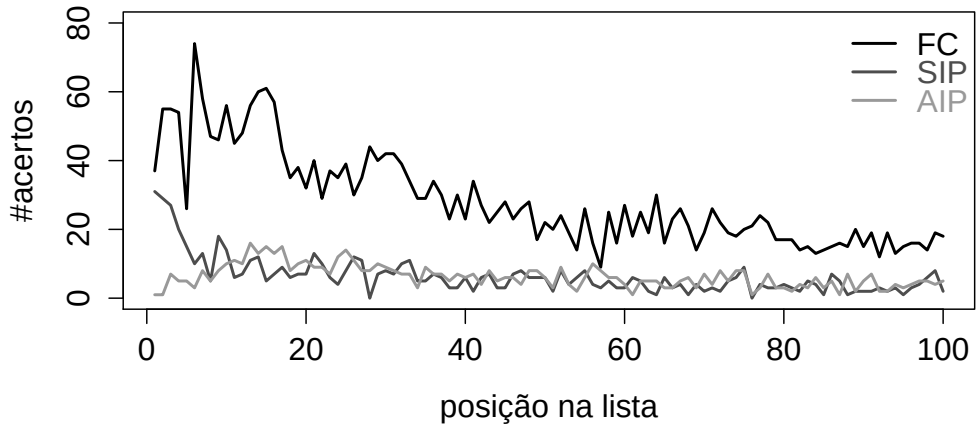

Fig. 4.4: Quantidade de acertos por Posição na lista de recomendação no *Movielens* em Given 1 com atributo **Todos** para SIP e AIP.

Agora que é conhecido um pouco melhor o comportamento dos três algoritmos em termos de seus acertos das previsões vamos analisar melhor se estes algoritmos acertam as mesmas previsões. Nesta análise é dispensada a comparação para Given 1, pois ao esconder somente 1 item do perfil do usuário os recomendadores só possuem uma única possibilidade de acerto o que transforma essa análise desnecessária. Neste dimensão é procurado identificar se quando os algoritmos acertam as previsões,

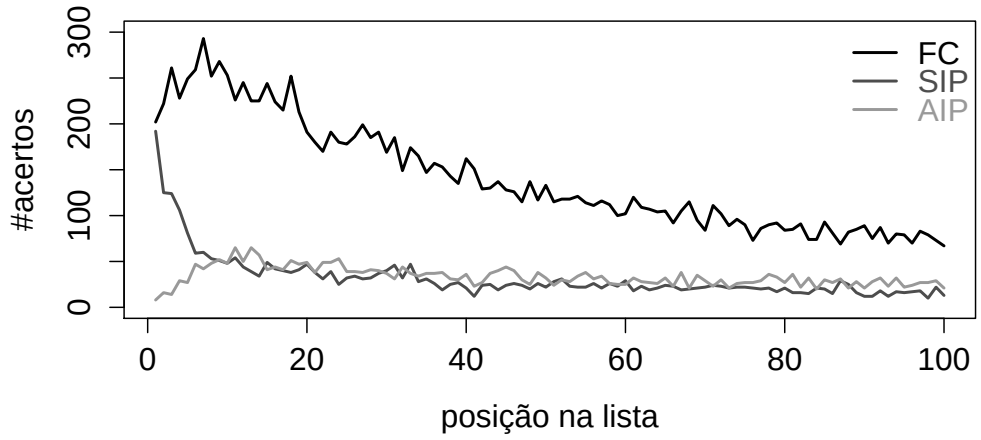

Fig. 4.5: Quantidade de acertos por Posição na lista de recomendação no *Movielens* em Given 5 com atributo **Todos** para SIP e AIP.

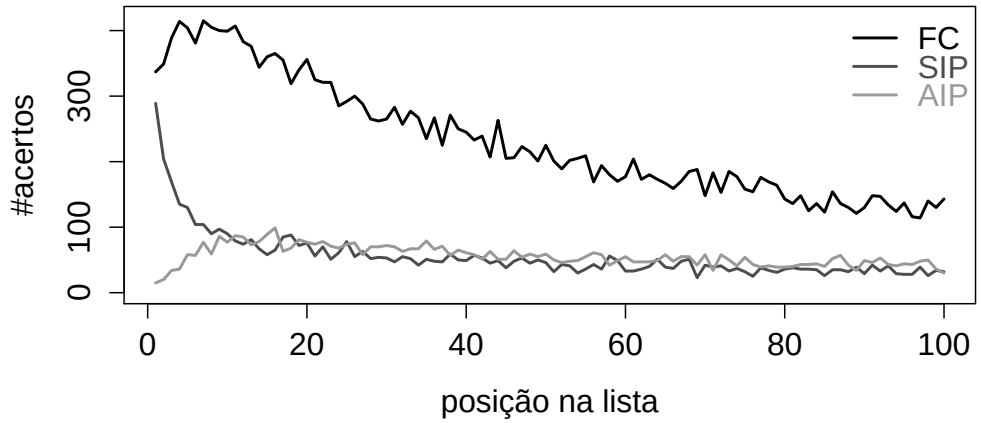

Fig. 4.6: Quantidade de acertos por Posição na lista de recomendação no *Movielens* em Given 10 com atributo **Todos** para SIP e AIP.

estes acertam para os mesmos itens. Neste caso buscamos quantificar três subconjuntos em relação ao conjunto de acertos de dois recomendadores: quantos acertos em comum (interseção), quantos acertos exclusivos do 1º algoritmo e quantos acertos exclusivos do 2º algoritmo. Para apresentar estes dados foi utilizado gráficos de linhas com as seguintes legendas: \ *Algoritmo* (e.g. \ *SIP*) indica os acertos exclusivos de um dos dois algoritmos avaliados e *Interseção* a quantidade de acertos em comum dos dois algoritmo. Todos este gráficos de interseção de acertos consideram as recomendações geradas para todos os usuários de teste sem utilização de *cross-validation* (*10-fold*).

Na Figura 4.7 apresentamos a interseção de acertos entre os algoritmos SIP e AIP em Given 5. A comparação se limita ao atributo *Todos* que apresentou superioridade para a medida MAP apreciada no início deste capítulo. É perceptível, para qualquer tamanho de lista, que a interseção dos acertos entre esses dois algoritmos é inferior que os acertos exclusivos de cada um dos algoritmos. Isto significa que mesmo utilizando algoritmos diferentes da abordagem baseada em conteúdo é possível realizar recomendações diferentes ou, de forma mais específica, é possível acertar previsões diferentes mostrando que técnicas diferentes de RI podem realizar diferentes contribuições em um Sistema de Recomendação.

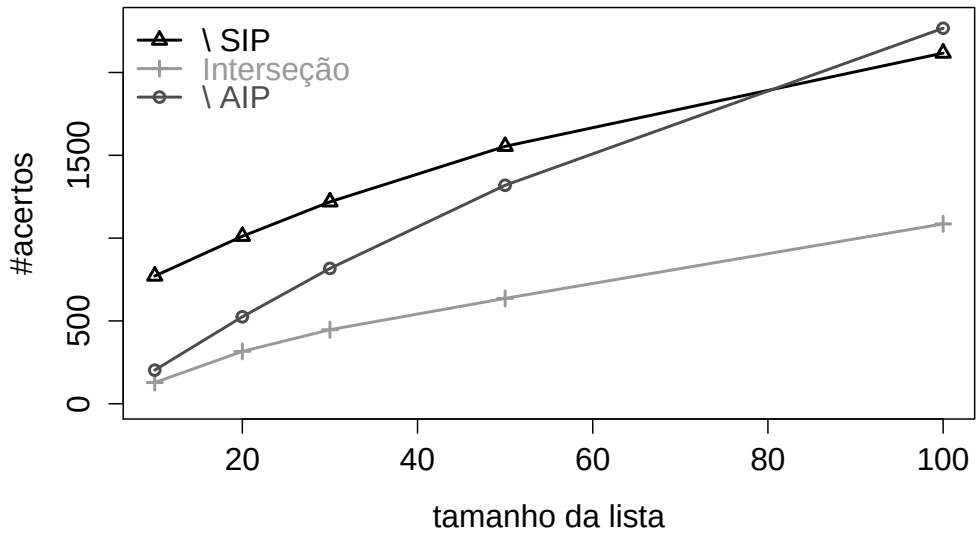

Fig. 4.7: Interseção dos acertos entre os algoritmos SIP e AIP com atributo **Todos** do conjunto de dados *MovieLens* em Given 5.

Comparando o algoritmo SIP de melhor MAP para o conjunto de dados *MovieLens* e o algoritmo de FC temos um comportamento diferente do anterior. Na Figura 4.8 apresentamos esse resultado e podemos perceber que em tamanho de listas pequenas a interseção é inferior ao subconjunto de acertos exclusivos do algoritmo SIP. A medida que aumenta o tamanho das listas de recomendação esse comportamento vai invertendo, mas somente em listas de recomendação com 100 itens que a parcela de acertos exclusivos de SIP é superada pela interseção. Ou seja, enquanto as listas de recomendação são de tamanhos como 10, 20, 30 ou mesmo 50 os itens recuperados pelo algoritmo SIP apresentam interesse ao usuário, mas com itens diferentes que FC não é capaz de recuperar.

Assim, é possível afirmar que apesar do algoritmo de Filtragem Colaborativa apresentar maior quantidade de acertos das previsões, as previsões realizadas pelo algoritmo SIP, em listas pequenas, são diferentes da abordagem social em sua maioria.

Realizamos também a comparação entre os algoritmos AIP e FC como apresentada na Figura 4.9. Considerando o conjunto de dados *MovieLens* em Given 5 os subconjuntos de interseção de acertos e os acertos exclusivos de AIP são muito pequenos em relação ao total de acertos. Ainda existe

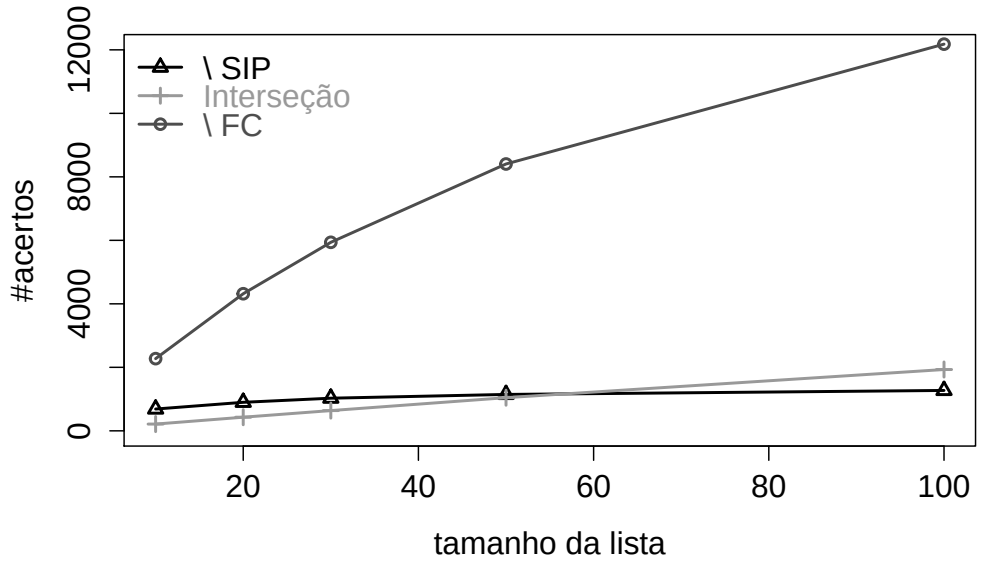

Fig. 4.8: Interseção dos acertos entre os algoritmos SIP com atributo **Todos** e FC do conjunto de dados *MovieLens* em Given 5.

uma quantidade de acertos exclusivos para a abordagem BC Textual, mas observamos um percentual menor dos acertos exclusivos do algoritmo AIP e também um interseção menor se compararmos com os resultados entre SIP e FC. A medida que cresce o tamanho da lista de recomendação a interseção sempre cresce. Porém o comportamento dos acertos exclusivos não é o mesmo: no início há um crescimento deste valor e após TOP30 uma pequena queda. Independentemente desse resultado, o comportamento geral é semelhante ao da comparação SIP e FC: o subconjunto de interseção cresce a medida que aumenta o tamanho da lista e ao final supera o subconjunto de acertos exclusivos da abordagem por conteúdo textual.

Nas Figuras 4.10, 4.11 e 4.12 apresentamos os resultados de comparação dos acertos das previsões para o conjunto de dados *MovieLens* em Given 10. O comportamento tem algumas características semelhantes ao apresentado nas três comparações anteriores sem indicar algum tipo de mudança importante em termos de quais previsões são realizadas e acertadas pelos diferentes algoritmos.

Os resultados apresentados para a similaridade entre os algoritmos estudados neste trabalho ajudam a um melhor entendimento sobre quais são as diferenças e semelhanças entre as duas principais abordagens de recomendação. Enquanto um algoritmo de recomendação baseado em conteúdo, AIP, apresenta suas listas de recomendação mais semelhantes às listas da FC, seus resultados de precisão das previsões são inferiores aos apresentados pelo algoritmo SIP quando verificado listas de recomendação menores (e.g. TOP10 e TOP20). Além disso, podemos afirmar que o algoritmo SIP possui uma característica diferenciada quando utilizado sobre os dados do *MovieLens*: seus acertos são na maioria diferentes aos realizados pela Filtragem Colaborativa também quando verificado listas de recomendação menores, pois a medida que aumenta o tamanho das listas os resultados de AIP e SIP se aproximam e FC passa a acertar grande parte das previsões feitas pelos algoritmos de recomendação

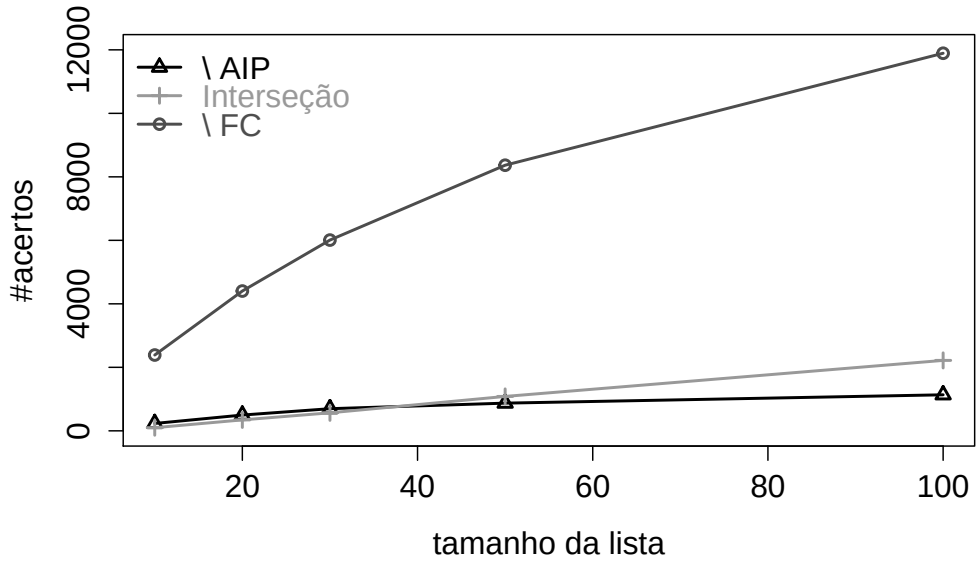

Fig. 4.9: Interseção dos acertos entre os algoritmos AIP com atributo **Todos** e FC do conjunto de dados *MovieLens* em Given 5.

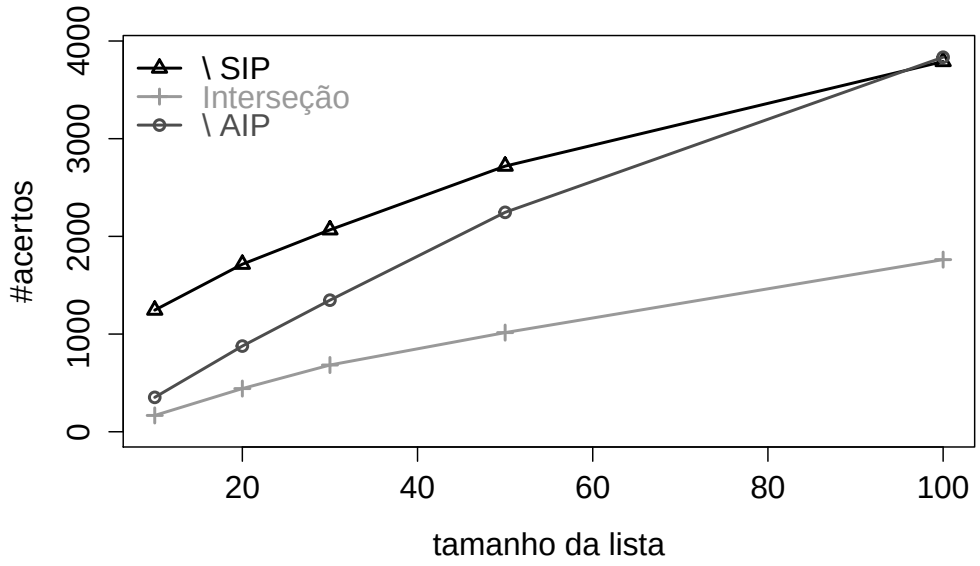

Fig. 4.10: Interseção dos acertos entre os algoritmos SIP e AIP com atributo **Todos** do conjunto de dados *MovieLens* em Given 10.

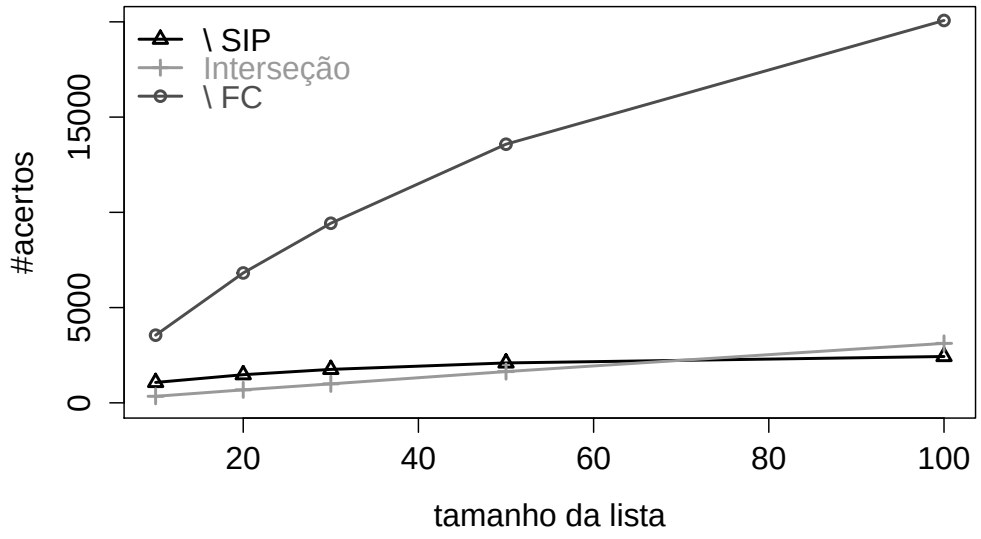

Fig. 4.11: Interseção dos acertos entre os algoritmos SIP com atributo **Todos** e FC do conjunto de dados *MovieLens* em Given 10.

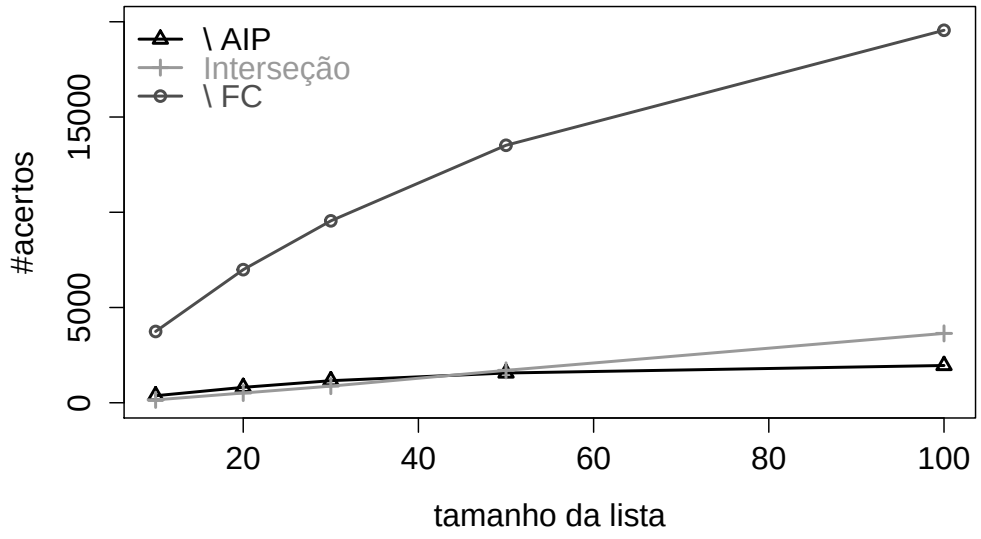

Fig. 4.12: Interseção dos acertos entre os algoritmos AIP com atributo **Todos** e FC do conjunto de dados *MovieLens* em Given 10.

BC Textual. Além disso, nossos resultados mostram uma nova ótica sobre a proposição levantada no trabalho de Burke (2002) no qual são discutidos os aspectos complementares das técnicas de recomendação. Testando Sistemas de Recomendação para a tarefa de "recomendar bons itens", além de verificarmos a cobertura do catálogo nas recomendações com resultados superiores para os algoritmos BC Textual, também verificamos que as previsões realizadas pelo algoritmo SIP são diferentes das de um recomendador social clássico.

4.4.2 Similaridade para CiteULike

Os recomendadores foram submetidos para três configurações diferentes no conjunto de dados *CiteULike* denominados Given 1, Given 5 e Given 10. Considerando os dois algoritmos SIP e AIP houve maior similaridade nos resultados em Given 1 para *Todos* os atributos concatenados seguido por Given 5 e Given 10. Na Figura 4.13 apresentamos os resultados de similaridade entre as listas de recomendação no conjunto de dados *CiteULike*.

De modo geral os dois recomendadores que filtram utilizando o conteúdo textual dos itens apresentam grande grau de similaridade entre as listas de recomendação geradas para os usuários ativos tanto no *CiteULike*, quanto no conjunto de dados *MovieLens*. Além disso, os resultados de MAP observados na Seção 4.1.2 demonstram que também possuem bom desempenho em realizar previsões. Embora a medida MAP demonstre essa capacidade, esta também pode esconder alguns detalhes no comportamento dos recomendadores por se tratar de uma "média". Na Figura 4.14 foram utilizados todos os atributos concatenados e observadas as posições do ranking nas quais cada um dos algoritmos acumulam mais acertos em Given 1. É possível perceber uma diferença entre os dois algoritmos BC Textual em relação aos acertos de suas previsões, em que SIP tem uma grande concentração nas primeiras posições da lista seguida de uma grande queda a medida que cresce a lista. Este comportamento do algoritmo SIP explica os resultados de MAP encontrados para este algoritmo quando cresce o tamanho das listas de recomendação. Nota-se que o aumento do tamanho não implica em um ganho expressivo dos resultados como, por exemplo, ocorre nos resultados de MAP de FC no conjunto de dados *MovieLens*. Outra observação é que o algoritmo AIP também apresenta um pico de acertos no início da lista, seguida da mesma queda apresentada pelo recomendador SIP. Entretanto, este pico é iniciado com valores baixos com um grande crescimento, ou seja, as primeiras posições (#1, #2, #3) apresentam baixo grau de acerto para AIP, diferente do algoritmo SIP que já inicia as primeiras posições como grande grau de acerto das previsões.

Também foram testadas as configurações do conjunto de dados *CiteULike* Given 5 e Given 10 como apresentado nas Figuras 4.15 e 4.16. Para o algoritmo AIP há uma maior suavidade no pico de acertos. Este comportamento demonstra que a quantidade de acertos demora mais algumas posições para chegar ao seu máximo (diferente do visto em Given 1) e também demora um pouco mais para finalizar a queda. Este comportamento explica o melhor desempenho deste algoritmo quando há um número maior de previsões a serem realizadas no conjunto de dados *CiteULike*. Ainda com relação a este comportamento a queda apresentada na curva inicial do algoritmo SIP também é menor, porém a altura do pico alcançado por AIP é maior indicando, de fato, sua maior capacidade de acertos mesmo perdendo nas primeiras posições da lista. Um comportamento similar em qualquer das configurações é que a quantidade de acertos ao final das listas é pequena. Por exemplo, em Given 1 próximo a posição 20 das listas de recomendação a quantidade de acertos é muito baixa e se mantém assim com valores próximos a zero até a posição 100.

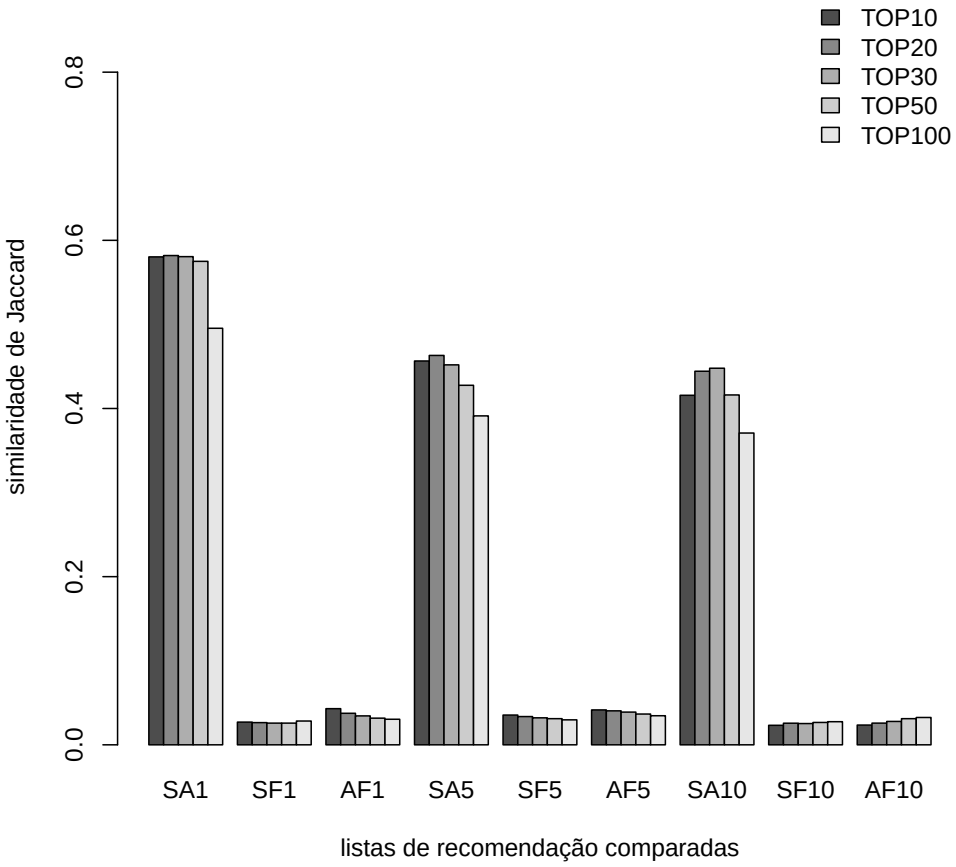

Fig. 4.13: Similaridade entre as recomendações geradas pelos algoritmos de recomendação com o conjunto de dados *CiteULike* para os atributos Todos.

Para o conjunto de dados *CiteULike* foram observadas também as semelhanças e diferenças de acertos realizados pelos algoritmos de recomendação propostos neste trabalho. A primeira verificação é dada no protocolo Given 5 entre os algoritmos SIP e AIP como é possível verificar na Figura 4.17. O resultado sugere que o tamanho da lista de recomendação desejada na aplicação do Sistema de Recomendação pode influenciar na escolha do algoritmo. Observado o comportamento dos dois algoritmos, se a tarefa for limitada a listas de tamanho TOP10 o algoritmo SIP apresenta uma maior taxa de acertos exclusivos das previsões (além da grande taxa de acertos da interseção). Entretanto, a partir de TOP20 o algoritmo AIP apresenta maior taxa de acertos exclusivos com o também crescimento da taxa dos acertos da interseção entre eles. Este comportamento sugere que o algoritmo SIP apresenta melhor desempenho em suas previsões ainda nas primeiras posições da lista, mas que ao crescer o tamanho das listas de recomendação o algoritmo AIP apresenta melhores resultados.

Foi testado também a interseção dos acertos para o algoritmo SIP e o algoritmo de FC no protocolo Given 5, mesmo considerando um cenário difícil de ser comparado por conta dos valores de UCOV

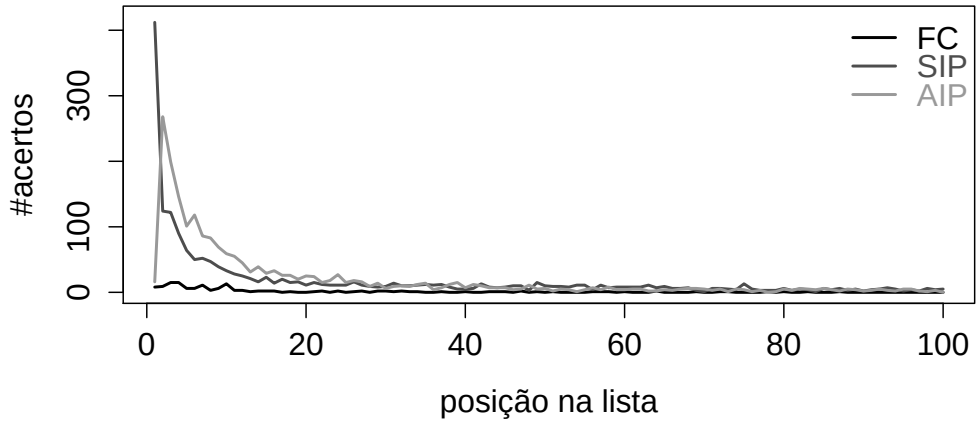

Fig. 4.14: Quantidade de acertos por Posição na lista de recomendação no *CiteULike* em Given 1 com atributo **Todos** para SIP e AIP.

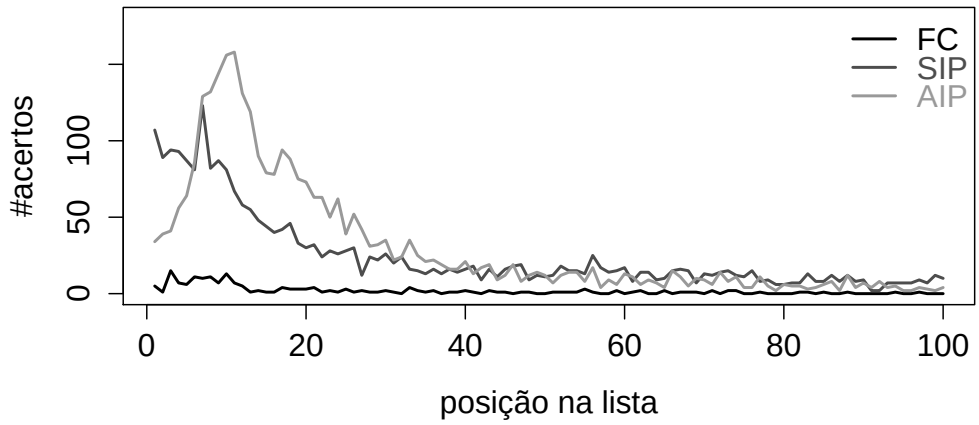

Fig. 4.15: Quantidade de acertos por Posição na lista de recomendação no *CiteULike* em Given 5 com atributo **Todos** para SIP e AIP.

muito baixos gerados pelo recomendador social. Fica mais uma vez demonstrado que as previsões realizadas pelas duas abordagens, representadas pelos seus respectivos algoritmos, são diferentes. Em nenhuma dos tamanhos de listas os algoritmos possuem interseção superior aos acertos realizados em FC.

A verificação das interseções de acertos entre AIP e o algoritmo de FC no protocolo Given 5 apresenta novamente um aspecto favorável ao algoritmo AIP no conjunto de dados *CiteULike*: com o crescimento da lista o recomendador BC Textual consegue sobrepor os acertos realizados pelo

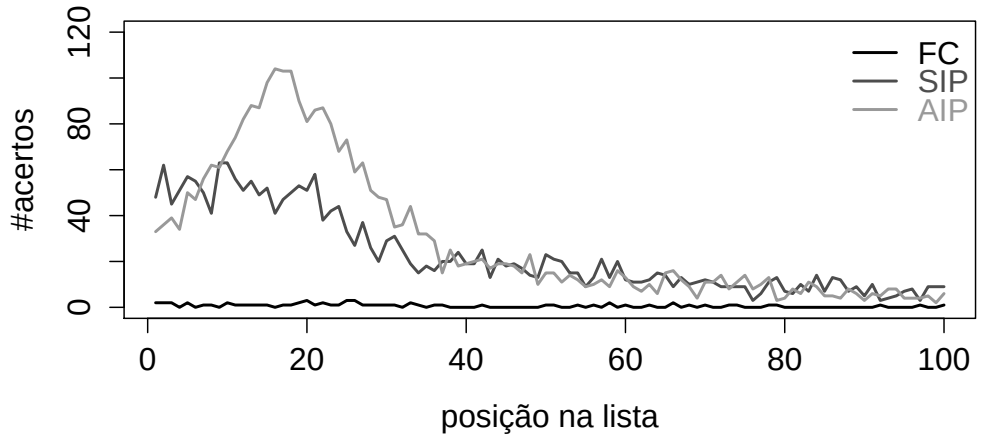

Fig. 4.16: Quantidade de acertos por Posição na lista de recomendação no *CiteULike* em Given 10 com atributo **Todos** para SIP e AIP.

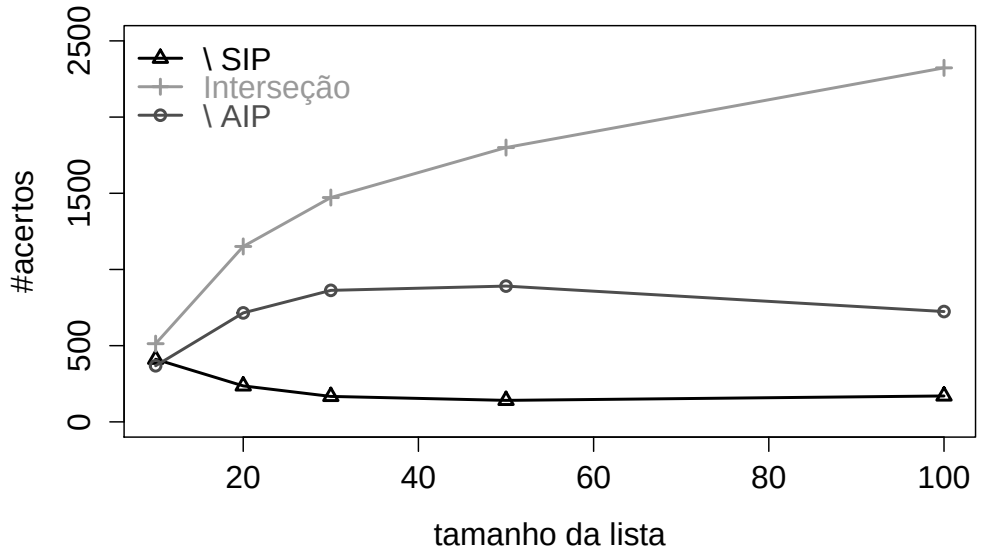

Fig. 4.17: Interseção dos acertos entre os algoritmos SIP e AIP com atributo **Todos** do conjunto de dados *CiteULike* em Given 5.

recomendador social. Como é verificado na Figura 4.19 já em TOP30 ocorre uma maior incidência de interseções de acertos entre os dois algoritmos em relação aos acertos exclusivos do algoritmo de FC. Entretanto, considerando listas de tamanhos menores (TOP10 e TOP20) novamente fica demonstrado que as diferentes técnicas geram previsões diferentes, mesmo em um cenário extremamente inóspito

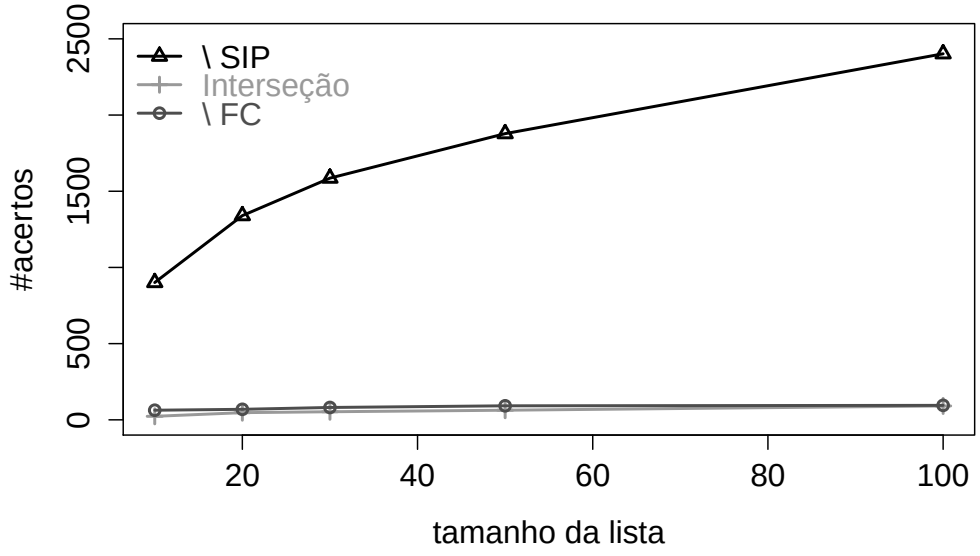

Fig. 4.18: Interseção dos acertos entre os algoritmos SIP com atributo **Todos** e FC do conjunto de dados *CiteULike* em Given 5.

para a abordagem social devida a grande esparsidade da relação $usuário \times item$.

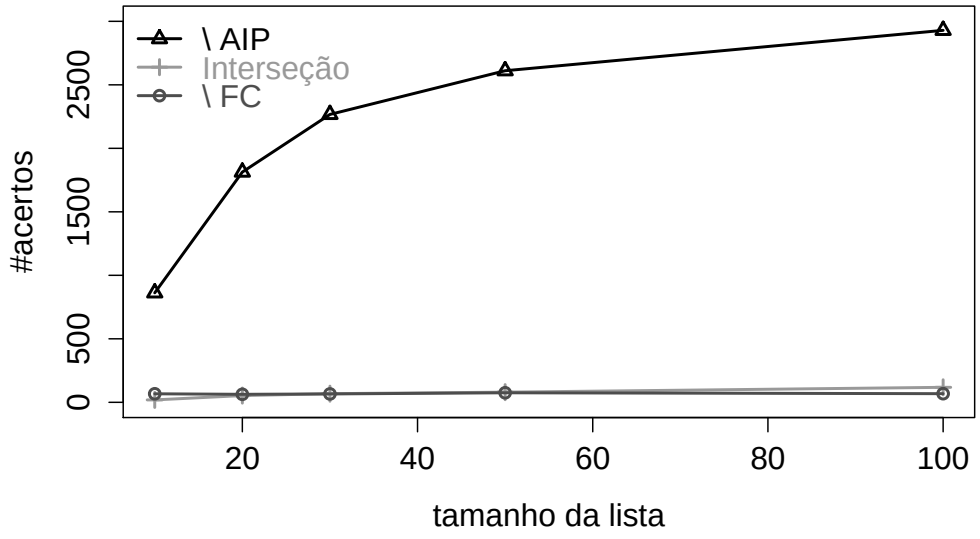

Fig. 4.19: Interseção dos acertos entre os algoritmos AIP com atributo **Todos** e FC do conjunto de dados *CiteULike* em Given 5.

Os resultados da verificação de interseção de acertos para o protocolo Given 10 apresenta o mesmo cenário apresentado em Given 5. Novamente, o algoritmo AIP apresenta comportamento superior considerando listas de tamanhos superiores ao TOP10. Além disso, todos os dois algoritmos de filtragem BC Textual não conseguem sobrepor na totalidade os acertos realizados pelo algoritmo de filtragem social. Nas Figuras 4.20, 4.21 e 4.22 apresentamos estes resultados similares ao padrão de comportamento descritos na análise em Given 5.

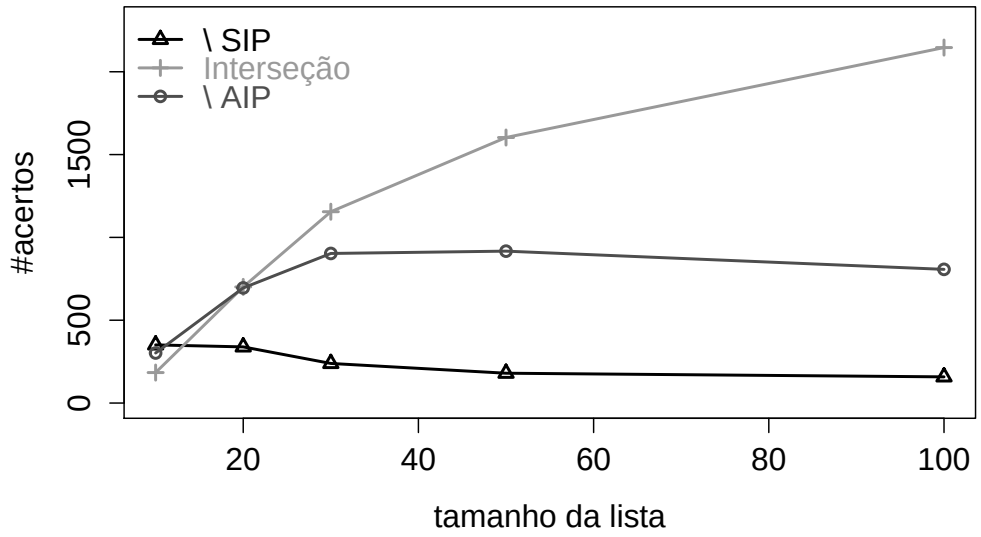

Fig. 4.20: Interseção dos acertos entre os algoritmos SIP e AIP com atributo **Todos** do conjunto de dados *CiteULike* em Given 10.

Quando analisado os resultados de similaridade encontrados no conjunto de dados *CiteULike* para os diferentes algoritmos propostos no Capítulo 3 podemos incluir mais um aspecto na análise dos algoritmos: o tamanho da lista de recomendação desejada pode influenciar na escolha do algoritmo quando a tarefa executada for "recomendar bons itens". Se observado o comportamento dos dois algoritmos BC Textual SIP e AIP conclui-se que em uma tarefa para gerar listas pequenas (de poucos itens evitando paginação pelo usuário na interface de uso do sistema) o algoritmo SIP demonstra melhor resultado. Esta conclusão é em decorrência da maior taxa de acertos verificados para esse algoritmo logo nas primeiras posições das listas de recomendação. Entretanto, a medida que cresce o tamanho da lista de recomendação o algoritmo AIP começa a apresentar comportamento superior. Se analisados somente os resultados de MAP que foram apresentados na Seção 4.1.2 é difícil realizar a escolha entre os dois algoritmos, já que estes apresentam resultados muito próximos e até invertem de posição entre quem é o melhor em termos de precisão quando ocorre mudanças de protocolo (de Given 5 para Given 10 o algoritmo AIP apresenta melhores resultados). Outro aspecto dos resultados de similaridade é que em qualquer das configurações do conjuntos de dados explorado, mesmo considerando o cenário bastante desfavorável para o algoritmo FC em *CiteULike*, os diversos algoritmos apresentam sempre um conjunto exclusivo de acertos em suas previsões. Isto significa que

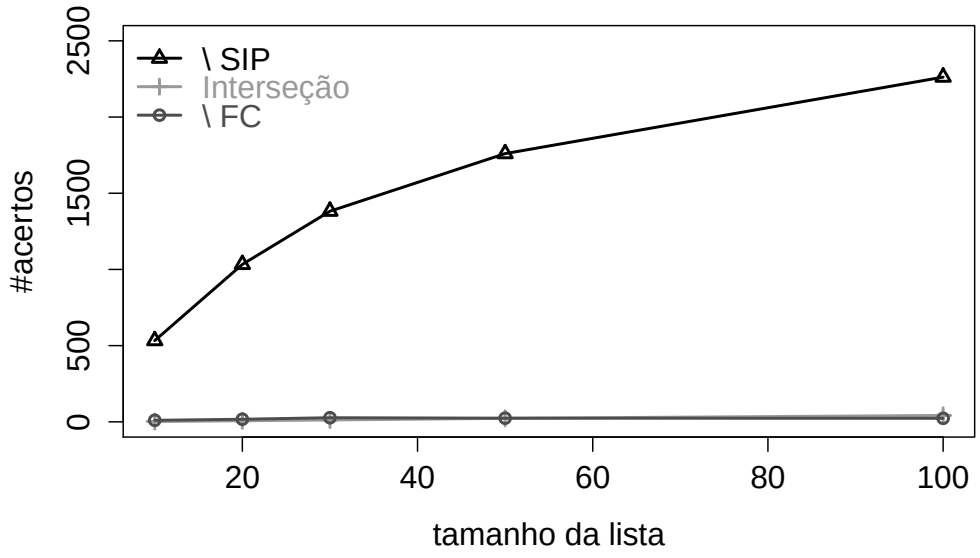

Fig. 4.21: Interseção dos acertos entre os algoritmos SIP com atributo **Todos** e FC do conjunto de dados *CiteULike* em Given 10.

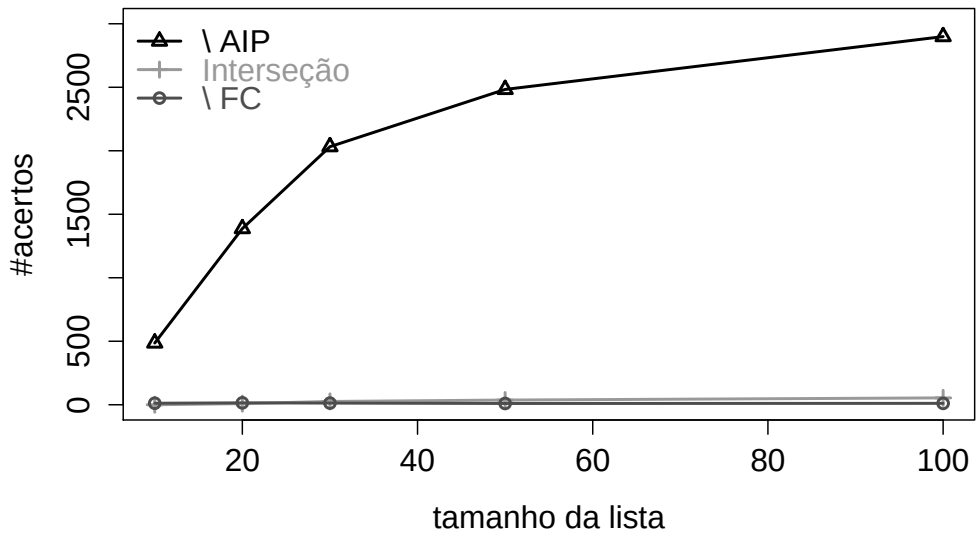

Fig. 4.22: Interseção dos acertos entre os algoritmos AIP com atributo **Todos** e FC do conjunto de dados *CiteULike* em Given 10.

sempre existe uma porção de acerto das previsões que um algoritmo é capaz de realizar e o outro não, indicando uma oportunidade de busca de soluções melhores para o problema proposto.

4.5 Resultados de Precisão das Previsões com Jaccard

Na construção de nossos algoritmos de recomendação Baseados em Conteúdo Textual criamos um modelo de representação dos itens (documentos) utilizando vetores de palavras (BOW) no qual cada posição armazena o resultado de $TF \times IDF$. Esta é uma representação dos documentos que adiciona ao sistema o custo do calculo de TF e IDF. No trabalho de Dumais et al. (1998) diversos experimentos sugerem que um modelo com pesos binários pode desempenhar tarefas de *Extração de Características* ou *Classificação de Documentos* de modo tão eficiente quanto outros modelos vetoriais com pesos diferentes (como no trabalho de Joachims (1998) que utiliza o tradicional $TF \times IDF$). Para analisar se em nossa proposta de experimento ocorre comportamento semelhante modificamos o modelo vetorial para pesos binários e o calculo de similaridade para a equação de *Similaridade Jaccard*. Além da diminuição do custo computacional na criação do modelo, a similaridade calculada utilizando a fórmula de *Jaccard* tem custo computacional de tempo inferior ao calculo de *Cosseno* o que proporciona um ganho de tempo considerável para o sistema no momento de gerar as recomendações por conteúdo.

4.5.1 Precisão com Similaridade Jaccard em MovieLens

Apresentamos nas Tabelas 4.25, 4.26 e 4.27 os valores de MAP obtidos no conjunto de dados *MovieLens* em Given 1, 5 e 10. Comparando os resultados obtidos aqui em Given 1 em relação aos apresentados na Tabela 4.1, do início deste capítulo, podemos notar que a mudança de representação fez o atributo **Todos** diminuir em torno de 21% seu desempenho quando analisada a precisão. Em contrapartida, analisando os resultados a cada tamanho de lista (TOP10,20,30,50,100) a concatenação dos atributos *Atores+Escritores+Diretores* formam a melhor solução na tarefa de "recomendar bons itens" com o algoritmo SIP. Desta forma este atributo manteve seu bom desempenho anteriormente apresentado seus novos resultados ficaram muito similares ao atributo **Todos** da Tabela 4.1.

Em Given 5 o panorama é muito similar ao encontrado em Given 1 como é possível perceber verificando os dados da Tabela 4.26. Novamente a concatenação dos atributos *1+2+3* tem o melhor resultado exceto em TOP10 se confrontarmos com o resultado de *Todos* com *Similaridade Cosseno*. Nos demais tamanhos de listas os valores de MAP para *1+2+3* são superiores, mas já não tão próximos aos apresentados pelo atributo *Todos* na Tabela 4.2 do início do capítulo.

Os resultados de MAP para o conjunto de dados *MovieLens* em Given 10 utilizando *Similaridade Jaccard* são apresentados na Tabela 4.27. Com esta configuração do experimento o comportamento do algoritmo SIP tem como principais características o atributo *1+2+3* apresentando resultado similar aos resultados da concatenação de *Todos* os atributos quando analisamos a Tabela 4.3 apresentada na Seção 4.1. E *Todos* os atributos concatenados novamente tendo um queda em seu desempenho de MAP.

Em nosso experimento é possível perceber que mudanças nas técnicas de Recuperação de Informação utilizadas tem impacto nos resultados da tarefa de recomendação empregada. Com as mudanças realizadas nos algoritmos de recomendação a concatenação dos atributos *Atores+Escritores+Diretores*

	Given 1									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0.0046	0.0008	0.0051	0.0014	0.0054	0.0016	0.0057	0.0020	0.0060	0.0022
Sinopse	0.0075	0.0018	0.0078	0.0023	0.0081	0.0025	0.0084	0.0028	0.0088	0.0031
Palavras	0.0044	0.0006	0.0047	0.0009	0.0049	0.0011	0.0052	0.0014	0.0056	0.0017
Gênero	0.0010	0.0010	0.0013	0.0011	0.0015	0.0012	0.0017	0.0014	0.0021	0.0016
Atores(1)	0.0102	0.0023	0.0109	0.0031	0.0111	0.0035	0.0114	0.0040	0.0118	0.0045
Escritores(2)	0,0060	**0,0024**	0,0072	**0,0033**	0,0076	0,0037	0,0081	0,0042	0,0086	0,0047
Diretores(3)	0.0042	0.0023	0.0050	0.0030	0.0053	0.0033	0.0059	0.0038	0.0066	0.0045
Revisão	0.0013	0.0012	0.0016	0.0016	0.0017	0.0018	0.0019	0.0021	0.0022	0.0024
1+2+3	**0,0112**	0,0023	**0,0125**	0,0033	**0,0129**	**0,0039**	**0,0134**	**0,0045**	**0,0139**	**0,0050**
Todos	0.0091	0.0018	0.0096	0.0025	0.0098	0.0028	0.0100	0.0031	0.0104	0.0036

Tab. 4.25: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 1 com Similaridade Jaccard.

	Given 5									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0.0042	0.0012	0.0049	0.0019	0.0052	0.0022	0.0055	0.0025	0.0060	0.0029
Sinopse	0.0092	0.0018	0.0096	0.0024	0.0099	0.0026	0.0103	0.0030	0.0108	0.0035
Palavras	0.0044	0.0003	0.0048	0.0006	0.0051	0.0009	0.0054	0.0012	0.0058	0.0016
Gênero	0.0023	0.0011	0.0027	0.0013	0.0029	0.0014	0.0032	0.0016	0.0036	0.0019
Atores(1)	**0,0121**	0,0024	0,0128	0,0031	0,0131	0,0036	0,0135	0,0042	0,0141	0,0049
Escritores(2)	0,0068	**0,0029**	0,0081	**0,0037**	0,0087	0,0041	0,0093	0,0046	0,0100	0,0052
Diretores(3)	0.0039	0.0027	0.0046	0.0034	0.0052	0.0039	0.0058	0.0045	0.0068	0.0053
Revisão	0.0014	0.0010	0.0017	0.0014	0.0019	0.0016	0.0021	0.0019	0.0025	0.0023
1+2+3	0,0114	0,0025	**0,0129**	0,0037	**0,0135**	**0,0043**	**0,0141**	**0,0050**	**0,0147**	**0,0058**
Todos	0.0116	0.0024	0.0120	0.0032	0.0123	0.0036	0.0126	0.0041	0.0131	0.0048

Tab. 4.26: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 5 com Similaridade Jaccard.

passou a apresentar os melhores resultados. Anteriormente, utilizando *Similaridade Cosseno* com representação $TF \times IDF$, a concatenação de *Todos* possuia o melhor resultado, mas nesta nova configuração teve queda expressiva de resultado de precisão. Esse comportamento mostra que quando utilizada técnicas de recomendação por meio de conteúdo é necessário um estudo de quais atributos são importantes e como utilizar esses atributos. Retomando nossa questão de pesquisa **Q2** podemos afirmar que utilizar técnicas de RI sem uma experimentação adequada, considerando somente as soluções tradicionais e sem levar em consideração que a tarefa executada é influenciada por um aspecto base: as preferências do usuário, pode levar o Sistema de Recomendação a resultados inferiores aos desejados. Ou mesmo elevar o custo computacional da solução tornando o sistema pouco escalá-

	Given 10									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0.0038	0.0009	0.0044	0.0016	0.0047	0.0019	0.0051	0.0024	0.0057	0.0028
Sinopse	0.0084	0.0019	0.0088	0.0025	0.0091	0.0028	0.0095	0.0032	0.0101	0.0037
Palavras	0.0046	0.0003	0.0050	0.0006	0.0053	0.0009	0.0056	0.0012	0.0060	0.0017
Gênero	0.0015	0.0010	0.0019	0.0012	0.0021	0.0013	0.0024	0.0014	0.0029	0.0018
Atores(1)	0.0118	0.0027	0.0127	0.0034	0.0130	0.0039	0.0135	0.0046	0.0141	0.0055
Escritores(2)	0,0065	**0,0028**	0,0078	0,0037	0,0086	0,0042	0,0094	0,0047	0,0103	0,0055
Diretores(3)	0.0029	0.0028	0.0038	0.0035	0.0044	0.0039	0.0052	0.0047	0.0065	0.0058
Revisão	0.0011	0.0011	0.0014	0.0015	0.0015	0.0017	0.0018	0.0020	0.0021	0.0024
1+2+3	**0,0118**	0,0024	**0,0135**	**0,0037**	**0,0144**	**0,0045**	**0,0152**	**0,0053**	**0,0160**	**0,0063**
Todos	0.0101	0.0026	0.0106	0.0034	0.0109	0.0039	0.0113	0.0045	0.0119	0.0053

Tab. 4.27: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *MovieLens* em Given 10 com Similaridade Jaccard.

vel em detrimento de escolhas não adequadas para um problema que é diferente de Recuperação de Informação.

4.5.2 Precisão com Similaridade Jaccard em CiteULike

Os algoritmos de recomendação SIP e AIP também geraram recomendações no conjunto de dados *CiteULike* utilizando um modelo de representação dos documentos (itens) com pesos binários e calculando a similaridade entre eles com *Jaccard*. Apresentamos esses resultados nas Tabelas 4.28, 4.29 e 4.30. Verificando primeiro Given 1 há uma grande queda de aproximadamente 31% no melhor resultado obtido pela *Similaridade Cosseno* com o atributo **Todos**. Utilizando *Jaccard* este atributo tem grande redução dos valores de MAP, enquanto o atributo *Tags* tem pequena variação com relação ao primeiro resultado: para o algoritmo SIP obteve uma pequena melhora por volta de 1% e para o algoritmo AIP uma piora de precisão em torno de 2%. Assim Tags passa a apresentar melhor desempenho nas previsões nesta situação e valores de MAP como 0,2020 (SIP/TOP100) são perdidos. Ainda sobre o atributo *Tags* este apresenta um pequeno aumento no valor de MAP somente até TOP20 comparando as Tabelas 4.5 e 4.28. Mais uma semelhança com aos primeiros resultados deste capítulo, na Seção 4.1.2, no protocolo Given 1 o melhor algoritmo é o SIP e novamente esta padrão é detectado para os resultados com *Jaccard*.

Prosseguindo a comparação entre os resultados de MAP dos algoritmos BC Textual em Given 5 continuamos com o algoritmo SIP apresentando os melhores resultados, um pouco diferente do comportamento resultante com *Similaridade Cosseno*. Antes havia uma ligeira vantagem para o algoritmo AIP que para esta configuração do experimento foi perdida passando o algoritmo SIP a obter o melhor resultado com uma pequena superioridade. O atributo *Tags* permanece apresentando os melhores resultados e o atributo *Todos*, novamente, sofre grande queda de 26% em seus resultados desde as primeiras partes das listas. Ainda sobre o atributo *Tags* nesta configuração do experimento há uma melhora com relação aos valores encontrados com *Cosseno* até com o maior tamanho de listas de

	Given 1									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0.0905	0.0442	0.0931	0.0478	0.0939	0.0490	0.0947	0.0499	0.0955	0.0505
Resumo	0.0800	0.0435	0.0824	0.0482	0.0833	0.0494	0.0840	0.0503	0.0846	0.0511
Autores	0.1025	0.0632	0.1034	0.0647	0.1038	0.0650	0.1042	0.0654	0.1045	0.0655
Publicação	0.0447	0.0355	0.0462	0.0375	0.0469	0.0379	0.0476	0.0385	0.0480	0.0389
Editor	0.0094	0.0068	0.0100	0.0075	0.0101	0.0077	0.0103	0.0079	0.0105	0.0082
Fonte	0.0611	0.0349	0.0625	0.0368	0.0631	0.0377	0.0638	0.0383	0.0644	0.0388
Tags	**0,1706**	**0,1334**	**0,1742**	**0,1369**	**0,1753**	**0,1381**	**0,1759**	**0,1387**	**0,1763**	**0,1391**
Todos	0.1334	0.0958	0.1367	0.1036	0.1380	0.1056	0.1393	0.1068	0.1407	0.1077

Tab. 4.28: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 1 com Similaridade Jaccard.

recomendação (TOP100) mesmo sendo uma melhora muito pequena ao comparar todos os tamanhos testados.

	Given 5									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0.0496	0.0162	0.0540	0.0244	0.0560	0.0283	0.0579	0.0314	0.0597	0.0334
Resumo	0.0449	0.0266	0.0499	0.0449	0.0518	0.0516	0.0536	0.0566	0.0557	0.0597
Autores	0.0506	0.0317	0.0550	0.0405	0.0560	0.0429	0.0570	0.0445	0.0584	0.0455
Publicação	0.0256	0.0241	0.0306	0.0277	0.0327	0.0295	0.0345	0.0314	0.0364	0.0326
Editor	0.0023	0.0040	0.0035	0.0046	0.0039	0.0049	0.0042	0.0054	0.0046	0.0058
Fonte	0.0384	0.0126	0.0430	0.0181	0.0446	0.0202	0.0462	0.0225	0.0480	0.0244
Tags	**0,1248**	**0,1215**	**0,1537**	**0,1510**	**0,1613**	**0,1585**	**0,1655**	**0,1626**	**0,1674**	**0,1653**
Todos	0.0786	0.0764	0.0962	0.1209	0.1021	0.1328	0.1066	0.1398	0.1117	0.1435

Tab. 4.29: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 5 com Similaridade Jaccard.

Os valores obtidos de MAP para os recomendadores SIP e AIP no conjunto de dados *CiteULike* em Given 10 utilizando *Similaridade Jaccard* são apresentados na Tabela 4.30. Observamos novamente a queda nos resultados de MAP para o atributo **Todos** com o algoritmo SIP em torno de 13%, embora nesta configuração o algoritmo AIP tenha obtido uma melhora próxima à 18%. Novamente o atributo **Tags** apresenta o melhor resultado e somente em Given 10 o algoritmo AIP passa a apresentar os melhores resultados. Diferente do que ocorreu nos resultados com *Similaridade Cosseno*, nos quais desde Given 5 ocorreu este fato. Neste contexto o algoritmo AIP tem uma ligeira piora nos resultados, mas é possível considera-los muito próximos diante da dificuldade da tarefa realizada.

O comportamento geral encontrado no contexto do *CiteULike* é similar ao *MovieLens* quando analisamos as mudanças realizadas nos algoritmos de recomendação BC Textual (SIP e AIP). A mu-

	Given 10									
	MAP@10		MAP@20		MAP@30		MAP@50		MAP@100	
	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP	SIP	AIP
Título	0.0412	0.0076	0.0481	0.0179	0.0511	0.0257	0.0544	0.0325	0.0580	0.0373
Resumo	0.0437	0.0173	0.0516	0.0504	0.0555	0.0668	0.0589	0.0787	0.0627	0.0858
Autores	0.0415	0.0177	0.0481	0.0289	0.0506	0.0359	0.0525	0.0396	0.0548	0.0421
Publicação	0.0212	0.0186	0.0287	0.0256	0.0332	0.0294	0.0371	0.0327	0.0408	0.0350
Editor	0.0021	0.0032	0.0036	0.0039	0.0041	0.0046	0.0047	0.0053	0.0054	0.0058
Fonte	0.0280	0.0073	0.0354	0.0143	0.0397	0.0198	0.0433	0.0244	0.0467	0.0283
Tags	**0,0890**	**0,0959**	**0,1423**	**0,1532**	**0,1675**	**0,1769**	**0,1785**	**0,1903**	**0,1828**	**0,1967**
Todos	0.0582	0.0574	0.0914	0.1254	0.1082	0.1623	0.1191	0.1807	0.1290	0.1905

Tab. 4.30: Valores de MAP obtidos dos recomendadores por Conteúdo Textual para o Conjunto de Dados *CiteULike* em Given 10 com Similaridade Jaccard.

dança de atributo com melhor resultado passa da concatenação de *Todos* para o atributo *Tags*. Quando consideramos um sistema no qual a quantidade de itens cadastrados é proporcionalmente muito superior a quantidade de usuários e, ainda que essa quantidade de itens em 2014, é na ordem de 7 milhões de registros uma solução baseada em conteúdo deve levar em consideração o custo computacional do calculo de similaridade entre os itens (documentos). Neste caso, com os resultados apresentados é possível uma solução de recomendação com custo de tempo inferior, mas com resultado de precisão próximo ao encontrado na solução *Similaridade Cosseno* e representação $TF \times IDF$. Este aspecto é levantado por Konstan & Riedl (2012) e não deve ser ignorado ao construir um SR que utilize conteúdo textual para realizar sua tarefa.

4.6 Respostas para as Questões de Pesquisa

Neste trabalho foi feita uma comparação entre dois algoritmos de recomendação Baseado em Conteúdo Textual, SIP e AIP, e um algoritmo de Filtragem Colaborativa. Como apresentado no Capítulo 1 os métodos aplicados na busca dos resultados apresentados neste capítulo visam responder algumas questões de pesquisas levantadas no início dessa dissertação. Abordaremos cada uma delas discutindo sobre os resultados obtidos neste trabalho.

4.6.1 Q1. O conteúdo textual dos itens pode ser uma fonte de dados relevante para Sistemas de Recomendação?

A resposta direta dessa pergunta é "sim". Observamos o comportamento de dois algoritmos de recomendação que utilizam conteúdo textual sobre os itens para realizar a tarefa de "recomendar bons itens". E, para este tipo de tarefa empregada em nosso experimento o uso do conteúdo textual demonstrou ser relevante para a geração de recomendações. Mesmo quando analisamos em termos de precisão das previsões no conjunto de dados *MovieLens*, no qual este tipo de algoritmo teve desempenho muito inferior ao algoritmo de Filtragem Colaborativa, foi possível perceber que os acertos das

previsões realizadas por cada um destes algoritmos foram diferentes. Observamos também que os diferentes dados extraídos apresentam resultados diferentes em avaliações como precisão e cobertura (catálogo) indicando que cada tipo de conteúdo possui uma relevância diferente neste tipo de sistema. Por exemplo, se compararmos os resultados de precisão entre os atributos *Atores principais* e *Título do filme* percebemos que o primeiro é muito mais relevante que o segundo em termos de precisão, mas é muito menos eficiente em cobrir o catálogo. Embora tenhamos obtidos resultados expressivos com os atributos *Atores+Diretores+Escritores* no conjunto de dados *MovieLens* em linhas gerais a concatenação de *Todos* os atributos apresentam os melhores resultados na construção deste tipo de recomendador.

Concluímos também que em contexto diferente, como o *CiteULike*, no qual existe baixa colaboração entre os usuários em termos de compartilhamento de suas bibliotecas de apontadores podemos afirmar que o conteúdo textual dos itens não só é uma fonte relevante, mas essencial para a construção de uma solução de recomendação do tipo *TopK*. Observa-se que pelos resultados de precisão e cobertura apresentados pelo algoritmo clássico de Filtragem Colaborativa, que este é incapaz de realizar a tarefa proposta. Enquanto que os algoritmos de recomendação BC Textual tiveram desempenho considerado relevante em todas as dimensões avaliadas em nosso teste de sistema. E, assim como ocorrido nos testes com o *MovieLens*, verificamos que cada tipo de dado apresenta resultados diferentes para as medidas testadas, mas com o atributo *Tags* apresentando os melhores resultados. O destaque para este conteúdo é o modo no qual ele é criado: de maneira colaborativa. Este resultado é também encontrado na pesquisa de Mooney & Roy (2000), em que o mesmo denomina este tipo de conteúdo como "conteúdo colaborativo" e, como visto na Seção 2.7, este tipo de conteúdo pode melhorar a precisão de um recomendador BC Textual.

Nossos resultados apontam para uma questão que pode ser a principal dificuldade na construção deste tipo de recomendador: obter o maior volume de dados possíveis sobre os itens a serem recomendados. Se observarmos nos resultados de atributos como *Fonte*, no conjunto de dados *CiteULike*, no qual ocorre uma baixa cobertura desse atributo (cerca de 38% dos itens possuem alguma informação sobre ele) nota-se que isso gerou resultados ruins para todas as avaliações realizadas. Neste sentido, mesmo considerando alguns resultados interessantes para atributos isolados (*Atores* em *MovieLens* e *Tags* em *CiteULike*) o uso do maior volume de dados disponível mostrou uma tendência que, quanto mais dados que possam caracterizar os itens estejam disponíveis, melhor as recomendações por conteúdo textual.

4.6.2 Q2. Como utilizar conteúdo textual para recomendação?

Diferente da primeira questão de pesquisa, a resposta para essa questão nos parece distante haja vista a quantidade de técnicas existentes para Mineração de Textos e que podem ser utilizadas na construção de SR BC Textual. Nos limitamos a explorar dois tipos de recomendadores que utilizam conteúdo textual dos itens de modo diferente, o primeiro (AIP) um filtro simples de conteúdo que visa extrair termos/palavras mais relevantes entre as preferências dos usuários. Por meio desse novo conteúdo é feita a busca por novos itens que possuam maior similaridade com este modelo de preferência do usuário. O segundo algoritmo (SIP) é um filtro que calcula a similaridade entre cada item do perfil do usuário com os demais itens do sistema. Os itens que apresentam maior similaridade entre ele e os itens do perfil do usuário são recuperados para formar as recomendações. Nossos resultados sugerem que a definição de qual algoritmo será utilizado é uma decisão que deve envolver diversas variáveis.

Foi possível notar que utilizando somente duas opções de algoritmos estes tiveram comportamentos diferentes para diferentes configurações do experimento. Para o conjunto de dados *MovieLens* o algoritmo SIP apresentou consistência em seu comportamento sendo sempre superior em termos de precisão com relação ao algoritmo AIP. Além disso, manteve-se com similaridade inferior às listas de recomendação apresentadas pelo algoritmo AIP em relação a Filtragem Colaborativa. Este aspecto nos parece mais salutar quando se pensa em mesclar soluções de recomendação em um sistema como fazem *Netflix* e *Amazon.com*, pois a diversidade entre as duas listas geradas por cada abordagem possivelmente será superior. Porém, na contramão do que ocorreu no *MovieLens* o comportamento apresentado pelo algoritmo AIP, no *CiteULike*, sugere a adição de novos aspectos na escolha de qual técnica por conteúdo selecionar: quantidade de previsões a serem realizadas e tamanho da lista de recomendação. Quando verificamos a precisão nos protocolos Given 1, Given 5 e Given 10 ocorre a inversão de qual algoritmo apresenta melhor desempenho, pois a partir de Given 5 AIP apresenta os melhores resultados. Além disso, observamos os resultados de similaridade entre SIP e AIP, quanto a quantidade de acertos de suas previsões, o primeiro tem melhor desempenho até TOP10. Entretanto, considerando listas de maior tamanho o algoritmo AIP torna-se uma solução mais interessante.

Outro aspecto importante quanto a esta questão de pesquisa é a escolha de qual conteúdo se deve utilizar na construção de uma solução deste tipo estudada em nosso trabalho. Como dito anteriormente, no *MovieLens* observamos que o atributo *Atores Principais* tem a maior contribuição na execução da tarefa. Este resultado sugere que o conteúdo utilizado por este tipo de recomendador deve ser melhor estudado, pois tem forte impacto nos resultados de precisão e também de cobertura. Outro resultado que confirma esta hipótese é o comportamento apresentado pelo conteúdo *Tags* no conjunto de dados *CiteULike*. Em algumas das configurações do experimento este conteúdo foi superior que a concatenação de todos os atributos, sugerindo que em determinadas situações alguns conteúdos devem ser ignorados em detrimento de um melhor desempenho nas previsões. Porém, quando não for possível realizar testes em cada um dos atributos ou mesmo na combinação deles, a melhor solução é utilizar todo o conteúdo disponível sobre os itens. Como observamos em nossos resultados, principalmente de precisão das previsões, esta opção está sempre entre os melhores resultados.

No trabalho de Konstan & Riedl (2012) é sugerido que explorar técnicas de recomendação que utilizam conteúdo pode ser uma solução viável, principalmente em termos de escalabilidade. Em diversos sistemas a quantidade de usuários possui crescimento exponencial e soluções sociais podem apresentar custo computacional extremamente alto. Em contrapartida, os problemas recorrentes de técnicas de RI quando aplicadas em uma tarefa de recomendação são também destacados pelos mesmos pesquisadores. Buscamos trazer nesta pesquisa uma amostra de que é possível utilizar técnicas de menor custo computacional sem afetar o desempenho no teste de sistema, em princípio quanto a sua precisão. Na Seção 4.5 modificamos nos algoritmos AIP e SIP a representação BOW utilizando pesos binários, ao invés de $TF \times IDF$ e o calculo de similaridade entre os itens para *Jaccard*. Os resultados de MAP apresentados com estas mudanças nos leva a afirmar que é possível utilizar técnicas de menor custo computacional e ao final obter resultado semelhante aos apresentados por técnicas mais tradicionais como já foi sugerido no trabalho de Dumais et al. (1998).

4.6.3 Q3. Quais as diferenças entre o recomendador Baseado em Conteúdo e Filtragem Colaborativa?

Os Sistemas de Recomendação podem ser utilizados em diferentes tipos de tarefas como descrito na pesquisa de Herlocker et al. (2004). Optamos neste trabalho por testar algoritmos de recomendação por meio da atividade de "recomendar bons itens", pois acreditamos que este tipo de tarefa é uma das mais utilizadas na web. Utilizando as configurações de experimento proposta no Capítulo 3 experimentamos três diferentes algoritmos em busca de resultados que pudessem evidenciar o comportamento dos mesmos para diferentes dimensões de avaliação e traçar algumas das diferenças entre eles. As três diferentes dimensões analisadas entre os recomendadores foram: precisão, cobertura de usuário e cobertura de catálogo e em cada uma destas foram observados comportamentos distintos para os algoritmos.

Quando os algoritmos foram comparados em precisão, utilizando a medida MAP, observamos grande diferença entre os algoritmos quando testados sobre o conjunto de dados *MovieLens*. O algoritmo de FC foi o mais eficiente, seguido do algoritmo SIP e com menor precisão o algoritmo AIP. Os resultados demonstraram que analisando precisão na recomendação de filmes a Filtragem Colaborativa foi até 4 vezes mais eficiente que o melhor resultado do algoritmo SIP e este por sua vez até 5 vezes melhor que o algoritmo AIP. Nesta dimensão de avaliação e no conjunto de dados *MovieLens* os algoritmos diferem muito em seus resultados. Até mesmo os dois algoritmos de recomendação BC Textual apresentam resultados muito diferentes. Por outro lado, verificando esta mesma dimensão para o conjunto de dados *CiteULike* observamos um comportamento oposto, onde FC apresenta resultados insatisfatórios e os algoritmos BC Textual aparecem como melhor solução para o problema proposto entre as duas abordagens testadas. Novamente, os algoritmos AIP e SIP não convergem em comportamento, pois em Given 1 o algoritmo SIP tem melhor desempenho utilizando *Todos* os atributos, mas isso não se mantém em Given 5 e Given 10 quando o algoritmo AIP apresenta melhor desempenho utilizando o atributo *Tags*. Em linhas gerais, diante dos resultados apresentados, podemos esperar melhores resultados dos algoritmos sociais quando há menor esparsidade de dados na relação $usuário \times item$ (como em *MovieLens*) e os melhores resultados para algoritmos BC Textual quando há maior quantidade de atributos que melhor caracterizam o item (como em *CiteULike*).

Além dos valores de precisão estudamos a similaridade entre as listas de recomendação de cada um dos algoritmos. Na maior parte dos resultados de similaridade calculados para *MovieLens* o algoritmo AIP apresenta listas com mais itens iguais as listas geradas por FC. Entretanto, considerando a melhor solução de precisão entre os dois algoritmos BC Textual (o algoritmo SIP) a dissimilaridade é maior. Ou seja, a melhor solução encontrada para recomendação por conteúdo textual no conjunto de dados *MovieLens*, apresenta listas de recomendação muito diferentes das listas por FC. Diante desse resultado buscamos verificar como é a relação dos conjuntos dos itens que formam os acertos das previsões realizadas pelos algoritmos. Os resultados demonstram que mesmo apresentando precisão inferior, os algoritmos BC Textual conseguem recuperar um conjunto de itens interessantes aos usuários, os quais, a Filtragem Colaborativa não é capaz de recuperar e este comportamento é mais evidente quando comparados SIP e FC. Este comportamento é semelhante quando mudamos o contexto e verificamos estes resultados no conjunto de dados *CiteULike*. Os algoritmos de recomendação que filtram por meio de conteúdo textual apresentam grande superioridade em termos de precisão se comparados com os resultados obtidos por FC. Entretanto, ainda se verificarmos os conjuntos de acertos exclusivos de cada um dos algoritmos avaliados constatamos que FC consegue

recuperar itens que os algoritmos Baseados em Conteúdo Textual não são capazes de apresentar aos usuários, mesmo sendo estes conjuntos muito pequenos em relação aos acertos dos algoritmos AIP e SIP.

Em termos de cobertura dos usuários, quando verificados os resultados em *MovieLens* todos os algoritmos apresentam o mesmo comportamento: em todas as configurações todos os algoritmos são capazes de cumprir com a tarefa de gerar listas de recomendação com diferentes tamanhos (TOP10, TOP20, TOP30, TOP50 e TOP100). De fato, as configurações desde conjunto de dados são favoráveis para qualquer das abordagens, pois apresenta esparsidade relativamente inferior a outros conjuntos de dados reais (sem filtros), além de uma cobertura alta quanto aos atributos recuperados em junção com os dados do *IMDb*. Porém, seguindo o comportamento apresentado pela dimensão de precisão das previsões a cobertura dos usuários apresentou resultado muito ruim com FC no conjunto de dados *CiteULike*. Este resultado sugere que neste tipo de contexto, no qual a há grande esparsidade na relação $usuário \times item$ uma solução social pura é inviável. Na contramão deste resultado os algoritmos BC Textual continuaram com excelente desempenho na execução da tarefa, demonstrando que FC sofre mais influência de problemas como o *user-cold-start* e *item-cold-start* do que recomendadores BC Textual.

Quando analisamos os resultados de cobertura do catálogo, buscamos não só verificar o percentual de itens que os algoritmos são capazes de recuperar dentro do catálogo do sistema, mas também, de forma indireta, sua capacidade de diversificar os itens na recomendação. Os resultados apresentados no conjunto de dados *MovieLens* demonstram que FC concentra-se em um percentual menor de itens para a execução da tarefa de "recomendar bons itens". Estes itens, consideramos mais populares, formam um conjunto no qual a abordagem social consegue realizar previsões, porém esta acaba sendo míope para o restante do catálogo. Os algoritmos de recomendação por conteúdo possuem uma cobertura até três vezes maior que aos valores apresentados em FC o que confere a esta abordagem maior diversidade de itens entre as listas geradas. Este comportamento se repete quando verificados os resultados de cobertura do catálogo no conjunto de dados *CiteULike*. Novamente, a FC concentra-se em um pequeno conjunto de itens, porém a proporção de itens alcançados pelos recomendadores de filtragem por conteúdo tem grande queda. No conjunto de dados *MovieLens* houveram resultados de quase 100% de cobertura do catálogo em algumas configurações do experimento, mas em *CiteULike* este valor não ultrapassa os 51% (considerando Given 1 que possui maior quantidade de usuários testados o que influencia nossa proposta para essa medida). Este comportamento reforça o *trade-off* entre $Coverage \times Precision$ discutidos em trabalhos de RI e também pesquisado no contexto de recomendação por Lathia et al. (2010). De modo geral é possível perceber que as duas principais abordagens diferem muito seus comportamentos quando analisadas suas capacidades de cobertura do catálogo.

Nossos resultados nos leva a reflexão que existem mais diferenças e complementaridades entre as duas principais abordagens de recomendação do que propriamente uma superposição ou domínio de uma sobre a outra. Nem mesmo considerando os contextos em que cada uma delas apresenta melhor desempenho de precisão das previsões (dimensão considerada como mais importante em testes de sistema como o proposto em nosso trabalho) nenhuma das abordagens conseguiu formar um grupo de acertos que fosse possível desconsiderar a relevância dos resultados apresentados pela outra abordagem. Desta maneira consideramos possível afirmar que algoritmos de recomendação, das duas principais abordagens, possuem comportamentos diferentes em diferentes contextos e em diferentes dimensões de avaliação em testes de sistema, mas com grande potencial de complementariedade.

Capítulo 5

Conclusão do Trabalho

Os Sistemas de Recomendação são ferramentas importantes para minimizar o problema da *sobrecarga de informação* presente em diversas aplicações da web como site de notícias, *e-commerce*, apontadores de publicações científicas, filmes e séries e muitos outros tipos de itens disponíveis que sobrecarregam e empobrecem a experiência dos usuários. Neste trabalho buscamos a comparação por meio de diferentes avaliações das duas principais abordagens utilizadas na construção deste tipo de sistema: Filtragem Baseada em Conteúdo (Textual) e Filtragem Colaborativa. Neste capítulo vamos apresentar as considerações finais deste trabalho na Seção 5.1 e na Seção 5.2 novos direcionamentos para esta pesquisa.

5.1 Considerações Finais

Este trabalho de dissertação de mestrado propôs evidenciar dados que por meio de sua análise fosse possível responder questões sobre as duas principais abordagens de recomendação. Dentro destas proposições consideradas, a ideia central passou por utilizar conteúdo textual disponível sobre os itens de diferentes sistemas para realizar a tarefa de "recomendar bons itens". O aspecto mais relevante e motivador para executar esse trabalho é que muitas pesquisas importantes sobre recomendação, já citadas no Capítulo 1, pouco evidenciam as diferenças entre as abordagens de modo empírico. Normalmente, as diferenciações são citadas atribuindo aos recomendadores as características oriundas das técnicas utilizadas. Por exemplo, se a filtragem de dados é realizada por técnicas de RI as características destes tipos de recomendadores são as mesmas de suas técnicas e a discussão acaba limitada a estas pré-deduções. O fato da tarefa em questão (recomendação) ser diferente e por isso ter problemas diferentes dos tratados em Recuperação de Informação, nos leva a acreditar que, neste caso, não valem simplesmente reproduzir algumas dessas afirmações.

Nesta busca por evidenciar as respostas de nossas questões de pesquisa conseguimos contribuir sobre as discussões que permeiam as comparações entre as duas principais abordagens de recomendação. Sobre os recomendadores Baseados em Conteúdo Textual foi possível verificar que diferentes técnicas oriundas de RI e diferentes conteúdos fornecem diferentes resultados em diferentes dimensões de avaliações. Ou seja, construindo diferentes algoritmos de recomendação BC Textual é possível obter diferentes resultados de recomendação. Comparando recomendadores BC Textual em relação a um algoritmo clássico de FC também obtivemos mais informações para estas discus-

sões: algoritmos das duas abordagens acertam diferentes previsões em uma tarefa de "recomendar bons itens". Isso indica que utilizando algoritmos considerados "puros" não é possível sobrepor as recomendações um do outro utilizando somente o resultado de uma das abordagens. Isto implica no conceito complementar que existe entre recomendadores dessas duas abordagens testadas em nosso trabalho. A busca por esta afirmação foi possível utilizando os cálculos de similaridade entre as listas de recomendação e a relação entre os conjuntos de acertos de previsão dos algoritmos testados propostos nesta dissertação.

Abordagens de recomendação definem princípios básicos nos quais devem ser seguidos pelos algoritmos de recomendação, logo em qualquer das abordagens é possível a construção de diversos algoritmos. Em um trabalho comparativo entre a abordagem Baseada em Conteúdo, sendo que a fonte de dados utilizada (conteúdo) é tratada como texto, existem diversas técnicas de RI e AM que podem ser utilizados no processo de filtragem e ordenação dos itens. Nosso trabalho concentrou-se em dois algoritmos BC Textual denominados AIP e SIP que utilizam técnicas em memória para realização de suas tarefas. Seria possível ainda testar diferentes tipos de representação dos textos, diferentes algoritmos de calculo de similaridade ou mesmo a aplicação de algoritmos de classificação. Devida a grande quantidade de técnicas de RI e AM disponíveis seria inviável uma avaliação de sistema de recomendação que combinassem todas elas e definitivamente não deixassem dúvidas sobre as comparações realizadas. Entretanto, a falta de testes com ao menos uma solução de classificação de documentos (e.g. Naïve Bayes) é considerada uma limitação em nosso trabalho. Além disso, em atributos como *Atores principais* que apresentou bom resultado em diferentes dimensões de avaliação de sistema, uma nova representação do conteúdo poderia ser utilizada. Em situações práticas é possível ocorrer a aproximação de itens pela presença de atores como *Robert De Niro* e *Robert Pattinson* por possuírem o mesmo nome próprio e, assim, acabar recomendando algum filme da série *Crepúsculo*[1] para algum amante da série *O Poderoso Chefão*[2] o que pode ser considerada (ou não) uma recomendação ruim.

5.2 Trabalhos Futuros

Dentro das limitações apontadas na Seção 5.1 destacamos como problemas de pesquisa para futuros trabalhos novas avaliações para recomendadores BC Textual que utilizem um modelo preditivo criado por algoritmos de classificação como *Naïve Bayes* ou *Rocchio's algorithm*. O uso deste tipo de algoritmo apresenta somente um inconveniente ao processo de ordenação dos resultados, principalmente quando uma quantidade maior de itens é recuperado em função do tamanho desejado para a lista. Além disso, em medidas como o MAP que consideram a posição do item na lista de recomendação o fator ordenação é ainda mais crítico. Nestes casos seria ainda necessário criar um critério de ordenação para os itens identificados como de interesse do usuário por parte desses classificadores. Ou utilizar outros tipos de algoritmos de classificação, a exemplo de *Perceptron Neural Network*, que além de classificar é capaz de informar um peso para cada classificação realizada. Esse peso poderia ser utilizado como fator de ordenação para as listas de recomendação a serem geradas pelo recomendador.

Na busca por generalização de nossos resultados testamos os algoritmos de recomendação propostos em dois diferentes conjuntos de dados *MovieLens+IMDb* e *CiteULike*. Entretanto, existem

[1] http://www.imdb.com/title/tt1099212/
[2] http://www.imdb.com/title/tt0068646/

diversos outros tipos de itens que sobrecarregam os usuários em suas experiências na web. Conteúdos como notícia, livro e música são alguns dentro da diversidade existente na web e testes de sistema para uma maior quantidade de conteúdos é importante e deve ser explorado em novos trabalhos comparativos entre as duas principais técnicas de recomendação: Filtragem Colaborativa e Baseada em Conteúdo.

Acreditamos que uma grande contribuição de nosso trabalho é apresentar por meio de uma observação mais detalhada a característica complementar que existe entre algoritmos de recomendação que utiliza técnicas das duas principais abordagens de recomendação. Foi possível perceber que há conteúdo relevante e exclusivo entre as listas de recomendação das duas abordagens, principalmente, pelo fato que os acertos das previsões realizadas pelos representantes de cada uma das abordagens são comumente diferentes. Porém, neste caso como combinar o uso dessas abordagens? No trabalho de Burke (2002) são apresentadas propostas de combinações entre abordagens de recomendação e ao final é realizado um experimento com uma proposta própria. Porém, é grande o trabalho de teste entre diversas combinações de abordagens de recomendação e muitas questões de como realizar essa combinação de técnicas ainda não foram respondidas. A busca para essas questões pode levar as pesquisas sobre Sistemas de Recomendação para novas direções. Por exemplo, podemos verificar nos resultados apresentados na Seção 4.4 que de modo geral os algoritmos de recomendação BC Textual concentram sua precisão em tamanhos pequenos de listas. Esse comportamento pode guiar algumas decisões em um processo de criação de um recomendador híbrido em caso de uma configuração *mixed* (Burke, 2002). Os testes com diferentes configurações de hibridização é um passo natural deste trabalho de mestrado na busca por descobrir como se comportam cada uma das abordagens quando combinadas.

Em projetos de algoritmos de recomendação é comum a definição de duas etapas de testes: teste de sistema e teste do usuário. Nossa pesquisa se concentrou na primeira etapa na qual não é necessária a presença de usuários e qualquer tipo de interação $homem \times máquina$. A construção de uma plataforma para que os algoritmos de recomendação propostos fossem convidados a apresentar suas recomendações para usuários reais é o próximo passo desta pesquisa. Muitos trabalhos como os de Konstan & Riedl (2012) e principalmente na pesquisa de McNee et al. (2006) é apresentada a perspectiva que nem sempre a busca de maior precisão nas previsões corresponde a maior satisfação dos usuários. Neste caso, considerando os Sistemas de Recomendação como uma ferramenta de apoio a uma melhor experiência do usuário na web, minimizando o problema da *sobrecarga de informação*, uma avaliação capaz de medir seu grau de satisfação é de grande importância em um projeto de algoritmo de recomendação.

Referências Bibliográficas

Aas, K. & Eikvil, L. (1999). Text categorisation: A survey. *Technical report, Norwegian Computing Center*.

Adomavicius, G., Sankaranarayanan, R., Sen, S., & Tuzhilin, A. (2005). Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems (TOIS)*, 23(1), 103–145.

Adomavicius, G. & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6), 734–749.

Adomavicius, G. & Tuzhilin, A. (2011). Context-aware recommender systems. In *Recommender Systems Handbook* (pp. 217–253). Springer.

Agichtein, E., Brill, E., & Dumais, S. (2006a). Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 19–26).: ACM.

Agichtein, E., Brill, E., Dumais, S., & Ragno, R. (2006b). Learning user interaction models for predicting web search result preferences. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 3–10).: ACM.

Agirre, E. & Edmonds, P. (2006). *Word sense disambiguation: Algorithms and applications*, volume 33. Springer Science+ Business Media.

Alias-i (2011). Lingpipe 4.1.0.

Anand, S. & Mobasher, B. (2007). Contextual recommendation. In *From web to social web: Discovering and deploying user and content profiles* (pp. 142–160). Springer.

Armstrong, R., Freitag, D., Joachims, T., & Mitchell, T. (1995). Webwatcher: A learning apprentice for the world wide web. In *AAAI Spring symposium on Information gathering from Heterogeneous, distributed environments* (pp. 6–12).

Baeza-Yates, R. & Ribeiro-Neto, B. (1999). *Modern information retrieval*, volume 463. ACM press New York.

Baharudin, B., Lee, L., & Khan, K. (2010). A review of machine learning algorithms for text-documents classification. *Journal of Advances in Information Technology*, 1(1), 4–20.

Balabanović, M. & Shoham, Y. (1997). Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3), 66–72.

Bellogín, A., Cantador, I., & Castells, P. (2010). A study of heterogeneity in recommendations for a social music service. In *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems* (pp. 1–8).: ACM.

Berry, M. (2003). *Survey of Text Mining I: Clustering, Classification, and Retrieval*, volume 1. Springer.

Berry, M., Drmac, Z., & Jessup, E. (1999). Matrices, vector spaces, and information retrieval. *SIAM Rev.*, 41(2), 335–362.

Berry, M., Dumais, S., & O'Brien, G. (1995). Using linear algebra for intelligent information retrieval. *SIAM review*, 37(4), 573–595.

Bogers, T. (2009). *Recommender Systems for Social Bookmarking*. PhD thesis, Tilburg University.

Breese, J., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence* (pp. 43–52).: Morgan Kaufmann Publishers Inc.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12, 331–370.

Burke, R. (2007). Hybrid web recommender systems. *The adaptive web*, (pp. 377–408).

Chirita, P.-A., Nejdl, W., & Zamfir, C. (2005). Preventing shilling attacks in online recommender systems. In *Proceedings of the 7th annual ACM international workshop on Web information and data management* (pp. 67–74).: ACM.

Claypool, M., Brown, D., Le, P., & Waseda, M. (2001). Inferring user interest. *Internet Computing, IEEE*, 5(6), 32–39.

Cremonesi, P., Koren, Y., & Turrin, R. (2010). Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10 (pp. 39–46). New York, NY, USA: ACM.

Cremonesi, P. & Turrin, R. (2009). Analysis of cold-start recommendations in iptv systems. In *Proceedings of the third ACM conference on Recommender systems* (pp. 233–236).: ACM.

Das, A., Datar, M., Garg, A., & Rajaram, S. (2007). Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07 (pp. 271–280). New York, NY, USA: ACM.

Deterding, S., Sicart, M., Nacke, L., O'Hara, K., & Dixon, D. (2011). Gamification. using game-design elements in non-gaming contexts. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '11 (pp. 2425–2428). New York, NY, USA: ACM.

Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management* (pp. 148–155).: ACM.

Gantner, Z., Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2011). Mymedialite: A free recommender system library. In *Proceedings of the fifth ACM conference on Recommender systems* (pp. 305–308).: ACM.

Ge, M., Delgado-Battenfeld, C., & Jannach, D. (2010). Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems* (pp. 257–260).: ACM.

Goldberg, D., Nichols, D., Oki, B., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12), 61–70.

Hacker, S. & von Ahn, L. (2009). Matchin: eliciting user preferences with an online game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09 (pp. 1207–1216). New York, NY, USA: ACM.

Harish, B., Guru, D., & Manjunath, S. (2010). Representation and classification of text documents: A brief review. *International Journal of Computer Applications IJCA*, (pp. 110–119).

Herlocker, J., Konstan, J., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 230–237).: ACM.

Herlocker, J., Konstan, J., & Riedl, J. (2000). Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work* (pp. 241–250).: ACM.

Herlocker, J., Konstan, J., Terveen, L., & Riedl, J. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1), 5–53.

Jin, W. & Srihari, R. (2007). Graph-based text representation and knowledge discovery. In *Proceedings of the 2007 ACM symposium on Applied computing*, SAC '07 (pp. 807–811). New York, NY, USA: ACM.

Jing, L.-P., Huang, H.-K., & Shi, H.-B. (2002). Improved feature selection approach tfidf in text mining. In *Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on*, volume 2 (pp. 944–946).: IEEE.

Joachims, T. (1996). *A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization.* Technical report, DTIC Document.

Joachims, T. (1998). *Text categorization with support vector machines: Learning with many relevant features*. Springer.

Joachims, T., Granka, L., Pan, B., Hembrooke, H., Radlinski, F., & Gay, G. (2007). Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)*, 25(2), 7.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, volume 14 (pp. 1137–1145).

Kohavi, R., Longbotham, R., Sommerfield, D., & Henne, R. (2009). Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1), 140–181.

Konstan, J. & Riedl, J. (2012). Recommender systems: from algorithms to user experience. *User Modeling and User-Adapted Interaction*, 22, 101–123.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08 (pp. 426–434). New York, NY, USA: ACM.

Kosala, R. & Blockeel, H. (2000). Web mining research: a survey. *SIGKDD Explor. Newsl.*, 2(1), 1–15.

Lam, S. & Riedl, J. (2004). Shilling recommender systems for fun and profit. In *Proceedings of the 13th international conference on World Wide Web* (pp. 393–402).: ACM.

Lathia, N., Hailes, S., Capra, L., & Amatriain, X. (2010). Temporal diversity in recommender systems. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval* (pp. 210–217).: ACM.

Lee, M., Choi, P., & Woo, Y. (2002). A hybrid recommender system combining collaborative filtering with neural network. In *Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, AH '02 (pp. 531–534). London, UK, UK: Springer-Verlag.

Lerman, K., Blair-Goldensohn, S., & McDonald, R. (2009). Sentiment summarization: Evaluating and learning user preferences. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 514–522).: Association for Computational Linguistics.

Liu, J., Dolan, P., & Pedersen, E. (2010). Personalized news recommendation based on click behavior. In *Proceedings of the 15th international conference on Intelligent user interfaces* (pp. 31–40).: ACM.

Liu, T., Liu, S., Chen, Z., & Ma, W.-Y. (2003). An evaluation on feature selection for text clustering. In *ICML*, volume 3 (pp. 488–495).

Lops, P., Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. *Recommender Systems Handbook*, (pp. 73–105).

Mak, H., Koprinska, I., & Poon, J. (2003). Intimate: a web-based movie recommender using text categorization. In *Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC International Conference on* (pp. 602–605).

Malone, T., Grant, K., Turbak, F., Brobst, S., & Cohen, M. (1987). Intelligent information-sharing systems. *Commun. ACM*, 30(5), 390–402.

Manning, C., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge.

Massa, P. & Bhattacharjee, B. (2004). Using trust in recommender systems: an experimental analysis. In *Trust Management* (pp. 221–235). Springer.

McNee, S. (2006). *Meeting user information needs in recommender systems*. PhD thesis.

McNee, S., Riedl, J., & Konstan, J. (2006). Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI'06 extended abstracts on Human factors in computing systems* (pp. 1097–1101).: ACM.

Mobasher, B., Burke, R., Bhaumik, R., & Williams, C. (2007). Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technology (TOIT)*, 7(4), 23.

Mooney, R. & Roy, L. (2000). Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, DL '00 (pp. 195–204). New York, NY, USA: ACM.

Nichols, D. (1997). Implicit rating and filtering. In *IN PROCEEDINGS OF THE FIFTH DELOS WORKSHOP ON FILTERING AND COLLABORATIVE FILTERING* (pp. 31–36).

Oard, D. & Kim, J. (1998). Implicit feedback for recommender systems. In *Proceedings of the AAAI workshop on recommender systems* (pp. 81–83).: Wollongong.

Pazzani, M. (1999). A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5-6), 393–408.

Pazzani, M., Muramatsu, J., & Billsus, D. (1996). Syskill & webert: Identifying interesting web sites. In *Proceedings of the national conference on artificial intelligence* (pp. 54–61).

Phelan, O., McCarthy, K., Bennett, M., & Smyth, B. (2011). Terms of a feather: Content-based news recommendation and discovery using twitter. *Advances in Information Retrieval*, (pp. 448–459).

Rajaraman, A. & Ullman, J. (2012). *Mining of massive datasets*. Cambridge University Press.

Ravikumar, P., Tewari, A., & Yang, E. (2011). On ndcg consistency of listwise ranking methods. In *International Conference on Artificial Intelligence and Statistics* (pp. 618–626).

Real, R. & Vargas, J. (1996). The probabilistic basis of jaccard's index of similarity. *Systematic biology*, (pp. 380–385).

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, CSCW '94 (pp. 175–186). New York, NY, USA: ACM.

Resnick, P. & Varian, H. (1997). Recommender systems. *Commun. ACM*, 40(3), 56–58.

Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to recommender systems handbook. In F. Ricci, L. Rokach, B. Shapira, P. B. Kantor, F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook* chapter 1, (pp. 1–35). Boston, MA: Springer.

Rodgers, J. & Nicewander, A. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1), 59–66.

Rucker, J. & Polanco, M. (1997). Siteseer: personalized navigation for the web. *Communications of the ACM*, 40(3), 73–76.

Salter, J. & Antonopoulos, N. (2006). Cinemascreen recommender agent: combining collaborative and content-based filtering. *Intelligent Systems, IEEE*, 21(1), 35–41.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce* (pp. 158–167).: ACM.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (pp. 285–295).: ACM.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1), 1–47.

Shahabi, C. & Chen, Y.-S. (2003). An adaptive recommendation system without explicit acquisition of user relevance feedback. *Distributed and Parallel Databases*, 14(2), 173–192.

Shani, G. & Gunawardana, A. (2011). Evaluating recommendation systems. In *Recommender systems handbook* (pp. 257–297). Springer.

Shardanand, U. & Maes, P. (1995). Social information filtering: algorithms for automating "word of mouth". In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95 (pp. 210–217). New York, NY, USA: ACM Press/Addison-Wesley Publishing Co.

Shuyo, N. (2010). Language detection library for java.

Silveira, G., Borenstein, D., & Fogliatto, F. (2001). Mass customization: Literature review and research directions. *International Journal of Production Economics*, 72(1), 1–13.

Sorensen, H., O'Riordan, A., & O'Riordan, C. (1997). Profiling with the informer text filtering agent. *Journal of Universal Computer Science*, 3(8), 988–1006.

Su, X. & Khoshgoftaar, T. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 4.

Swearingen, K. & Sinha, R. (2001). Beyond algorithms: An hci perspective on recommender systems. In *ACM SIGIR 2001 Workshop on Recommender Systems*, volume 13 (pp. 393–408).: Citeseer.

Thompson, C. (2008). If you liked this, you're sure to love that. *The New York Times*, 21.

Van Meteren, R. & Van Someren, M. (2000). Using content-based filtering for recommendation. In *Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop*.

Vanetti, M., Binaghi, E., Carminati, B., Carullo, M., & Ferrari, E. (2011). Content-based filtering in on-line social networks. In *Privacy and Security Issues in Data Mining and Machine Learning* (pp. 127–140). Springer.

Voorhees, E. (2002). The philosophy of information retrieval evaluation. In *Evaluation of cross-language information retrieval systems* (pp. 355–370).: Springer.

Vucetic, S. & Obradovic, Z. (2005). Collaborative filtering using a regression-based approach. *Knowledge and Information Systems*, 7(1), 1–22.

Wang, Y. & Wang, X.-J. (2005). A new approach to feature selection in text classification. In *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, volume 6 (pp. 3814–3819).

Whittaker, S. & Sidner, C. (1996). Email overload: exploring personal information management of email. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '96 (pp. 276–283). New York, NY, USA: ACM.

Yang, Y. & Pedersen, J. (1997). A comparative study on feature selection in text categorization. In *ICML*, volume 97 (pp. 412–420).

Dissertação escrita com a versão gratuita do ShareLaTeX(http://www.sharelatex.com/).