UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Ranqueamento Personalizado Baseado em Dados Enriquecidos: Uma Abordagem de Co-Treinamento

Marcelo Miky Mine

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP SÃO CARLOS

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Marcelo Miky Mine

Ranqueamento Personalizado Baseado em Dados Enriquecidos: Uma Abordagem de Co-Treinamento

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Marcelo Garcia Manzato

USP – São Carlos
Junho de 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Mine, Marcelo Miky
M664r Ranqueamento Personalizado Baseado em Dados Enriquecidos: Uma Abordagem de Co-Treinamento / Marcelo Miky Mine; orientadora Marcelo Garcia Manzato. -- São Carlos, 2021.
93 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2021.

1. Sistemas de Recomendação. 2. Aprendizado Semissupervisionado. 3. Co-treinamento. 4. Esparsidade. 5. Ranqueamento. I. Manzato, Marcelo Garcia, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Marcelo Miky Mine

Personalized Ranking Based On Enriched Data: A Co-Training Approach

Master dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Master Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Marcelo Garcia Manzato

USP – São Carlos
June 2021

AGRADECIMENTOS

Aos meus pais e irmão que direcionaram meu caminho de forma livre, uma liberdade compreendida tardiamente.

À minha esposa Flávia, pelo amor, carinho e paciência durante todo o tempo de nossa convivência.

À Universidade de São Paulo e ao Instituto de Ciências Matemáticas e de Computação pela oportunidade e aprendizados.

Ao professor Marcelo Manzato pela dedicação, disponibilidade e orientações prestadas na pesquisa.

Aos professores Solange Rezende, Moacir Ponti, Rafael Izbicki e Osvaldo Anacleto pela amizade e conversas.

Aos amigos Fernando, Fernanda, Edvard, Fausto, Rayner, Faimison e Adam pelos encontros, amizade e experiências culinárias.

Aos amigos e colegas das comunidades de tecnologia do Sancahub: grupy-sanca, Open-sanca, sancaLUG e Data Science Sanca; pelos conhecimentos transmitidos, amadurecimento e companheirismo.

Ao Brian Eno e Peter Schmidt pelo *Oblique Strategies* e reflexões.

À Angeles Arrien pelo Caminho Quádruplo e orientações para uma vida mais equilibrada.

À Ayurveda e ao Yoga pelos ensinamentos, auto-conhecimento e equilíbrio proporcionados.

Por fim, à todes que de forma direta ou indireta contribuíram para este trabalho.

“Repetition is a form of change” (Brian Eno and Peter Schmidt)

RESUMO

MINE, M. M. **Ranqueamento Personalizado Baseado em Dados Enriquecidos: Uma Abordagem de Co-Treinamento**. 2021. 93 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

A crescente oferta de produtos, serviços e informações na Web tem levado diversas aplicações a utilizar e desenvolver sistemas de recomendação para sugerir conteúdos de acordo com as preferências de cada usuário específico. Nesses sistemas, uma grande quantidade de dados rotulados deve estar disponível para obter boas previsões. No entanto, os dados rotulados são frequentemente limitados e caros de se obter, uma vez que a rotulagem geralmente requer tempo e conhecimento humano. Além disso, geralmente, os usuários estão interessados apenas nas recomendações com melhores posições, enquanto as sugestões com posições inferiores são ignoradas. Como faltam métodos que promovam o enriquecimento (rotulagem) com classificação, este trabalho propõe uma técnica que recomenda os *top-n* itens com base em dados enriquecidos. O enriquecimento é baseado em um método de co-treinamento, cuja aprendizagem usa múltiplas visualizações - dados divididos em dois subconjuntos separados - para diminuir a esparsidade dos dados. A escolha dos itens no enriquecimento é baseada em uma métrica de confiança, que busca itens mais relevantes para usuário. Então, esses dois subconjuntos enriquecidos são combinados com um método de *ensemble*. Na avaliação, utilizamos conjuntos de dados reais de domínios distintos e os experimentos mostram que o método proposto atinge melhores resultados, em comparação com o *baseline*, quando um algoritmo de recomendação é usado nestes dados enriquecidos combinados.

Palavras-chave: Sistemas de Recomendação, Aprendizado Semissupervisionado, Co-treinamento, Esparsidade, Ranqueamento.

ABSTRACT

MINE, M. M. **Personalized Ranking Based On Enriched Data: A Co-Training Approach**. 2021. 93 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

The growing supply of products, services and information on Web has led several applications to use and develop recommendation systems to suggest content according to each specific user's preference. In these systems, a large amount of labeled data must be available to obtain good predictions. However, labeled data are often limited and expensive to obtain, since labelling usually requires time and human knowledge. Besides, usually, users are only interested in the best ranked recommendations, while lower-ranked suggestions are ignored. Since there is a lack of methods that promote enrichment (labelling) with ranking, this work proposes a technique that recommends top-n items based on enriched data. The enrichment is based on a co-training method, whose learning uses multiple views, data divided into two disjoint subsets, to decrease the data sparsity. The choice of items to enrich is based on a confidence metric, which seeks items that are more relevant to the user. Then, these two enriched subsets are combined with an ensemble method. In our evaluation, we used real data sets from distinct domains and experiments show that the proposed method achieves better results, compared to baseline, when a recommender algorithm is used in this enriched ensemble set.

Keywords: Recommender Systems, Semi-supervised learning, Co-training, Sparsity, Ranking.

LISTA DE ILUSTRAÇÕES

Figura 1 – Esquema visual do Sistema 54
Figura 2 – Diagrama de Venn dos Conjuntos de Dados. 59
Figura 3 – Esquema Visual - Caso 1. 59
Figura 4 – Esquema Visual - Caso 1 para o cenário de partida-fria. 60
Figura 5 – Esquema Visual - Caso 2. 61
Figura 6 – Esquema Visual - Caso 3. 61
Figura 7 – Gráfico - Média Avaliações versus Esparsidade (%) - Filmtrust. 73
Figura 8 – Gráfico - Média Avaliações versus Esparsidade (%) - Jester. 73
Figura 9 – Gráfico - Média Avaliações versus Esparsidade (%) - Movielens. 74

LISTA DE ALGORITMOS

Algoritmo 1 – CoRec 56

LISTA DE TABELAS

Tabela 1 – Classificações de possíveis resultados da recomendação de um item para um usuário 41
Tabela 2 – Diferenças entre os trabalhos. 52
Tabela 3 – Esparsidade dos conjuntos originais e enriquecidos em porcentagem. 65
Tabela 4 – Resultados do BPRMF para o Caso 1 - Base *Filmtrust*. 66
Tabela 5 – Resultados do BPRMF para o Caso 1 - Base *Jester*. 67
Tabela 6 – Resultados do BPRMF o Caso 1 - Base *Movielens*. 67
Tabela 7 – Resultados do BPRMF para o Caso 2 - Base *Filmtrust*. 67
Tabela 8 – Resultados do BPRMF para o Caso 2 - Base *Jester*. 68
Tabela 9 – Resultados do BPRMF para o Caso 2 - Base *Movielens*. 68
Tabela 10 – Resultados para o Caso 3 - Base *Filmtrust*. 69
Tabela 11 – Resultados para o Caso 3 - Base *Jester*. 69
Tabela 12 – Resultados para o Caso 3 - Base *Movielens*. 69
Tabela 13 – Resultados do BPRMF para o cenário de partida-fria - Base *Filmtrust*. . . . 70
Tabela 14 – Resultados do BPRMF para o cenário de partida-fria com $X = 5$ - Base *Jester*. 71
Tabela 15 – Resultados do BPRMF para o cenário de partida-fria com $X = 10$ - Base *Jester*. 71
Tabela 16 – Resultados do BPRMF para o cenário de partida-fria com $X = 20$ - Base *Jester*. 71
Tabela 17 – Resultados do BPRMF para o cenário de partida-fria com - Base *Movielens*. 72
Tabela 18 – Comparação dos resultados do BPRMF para o Caso 1, discutido na Seção 4.5.1. 88
Tabela 19 – Comparação dos resultados do BPRMF para o Caso 2, discutido na Seção 4.5.2. 89
Tabela 20 – Comparação dos resultados do BPRMF para o Caso 3, discutido na Seção 4.5.3. 90
Tabela 21 – Comparação dos resultados do BPRMF para o cenário de partida-fria - Dataset Filmtrust 91
Tabela 22 – Comparação dos resultados do BPRMF para o cenário de partida-fria - Dataset Jester 92
Tabela 23 – Comparação dos resultados do BPRMF para o cenário de partida-fria - Dataset Movielens 93

LISTA DE ABREVIATURAS E SIGLAS

AP	*Average Precision*
AS	Aprendizado Semissupervisionado
BPRMF	Bayesian Personalized Ranking Matrix Factorization
CoRec	*Co-Training Approach for Recommender Systems*
DCG	*Discounted Cumulative Gain*
MAE	*Mean Absolute Error*
MAP	*Mean Average Precision*
MSE	*Mean squared error*
NDCG	*Normalized Discounted Cumulative Gain*
RMSE	*Root Mean Square Error*

SUMÁRIO

1 INTRODUÇÃO 25
1.1 Motivação 26
1.2 Objetivos 28
1.3 Estrutura do Trabalho 28

2 SISTEMAS DE RECOMENDAÇÃO 29
2.1 Conceitos Gerais 30
2.2 Abordagens Principais 31
2.2.1 Filtragem Colaborativa *32*
2.2.2 Filtragem Baseada em Conteúdo *32*
2.2.3 Abordagens Híbridas *33*
2.3 Principais Conceitos Para a Proposta 34
2.3.1 Esparsidade *34*
2.3.2 Partida-Fria *34*
2.3.3 Aprendizado Semissupervisionado *35*
2.3.4 User k-Nearest Neighbors (User-KNN) *36*
2.3.5 Item k-Nearest Neighbors (Item-KNN) *38*
2.3.6 Avaliação em Sistemas de Recomendação *38*
2.3.6.1 RMSE e MAE *40*
2.3.6.2 Precision e Recall *41*
2.3.6.3 Mean Average Precision (MAP) *42*
2.3.6.4 Normalized Discounted Cumulative Gain (NDCG) *42*
2.4 Considerações Finais 43

3 ENRIQUECIMENTO DE CONJUNTOS DE DADOS 45
3.1 Imputação 45
3.2 Side Information 47
3.3 Co-Treinamento 47
3.4 Ensembles 49
3.5 Trabalhos Relacionados 50
3.6 Considerações Finais 52

4 PROPOSTA DE SOLUÇÃO 53
4.1 Arquitetura Geral 53

4.1.1 CoRec *53*
4.1.2 Bayesian Personalized Ranking Matrix Factorization *55*
4.2 Pré-Processamento 55
4.3 Co-Treinamento e Ensemble 56
4.4 Métrica de Confiança 57
4.5 Recomendação De Itens Ranqueados 58
4.5.1 Conjunto Enriquecido Como Conjunto de Treino - Caso 1 . . . . . *59*
4.5.2 Conjunto Enriquecido Utilizado Como Treinamento do Ranqueador - Caso 2 . . . . . *60*
4.5.3 Conjunto de Itens Mais Confiáveis como Ranking de Recomendação - Caso 3 . . . . . *61*
4.6 Considerações Finais 62

5 EXPERIMENTOS E RESULTADOS 63
5.1 Metodologia de Avaliação 63
5.1.1 Ferramentas Utilizadas *63*
5.1.2 Bases de Dados *64*
5.1.3 Método de Validação e Avaliação *64*
5.2 Validação da Proposta 65
5.2.1 Avaliação dos Rankings - Caso 1 . . . . . *66*
5.2.2 Avaliação dos Rankings - Caso 2 . . . . . *67*
5.2.3 Avaliação dos Rankings - Caso 3 . . . . . *68*
5.2.4 Cenário de Coldstart *69*
5.2.5 Discussão dos Resultados *71*
5.3 Considerações Finais 75

6 CONCLUSÕES E TRABALHOS FUTUROS 77
6.1 Resumo do Trabalho 77
6.2 Contribuições da Pesquisa 78
6.2.1 Conclusões e Aplicações *78*
6.2.2 Artigo Submetido *79*
6.3 Trabalhos Futuros 79

REFERÊNCIAS 81

APÊNDICE A RESULTADOS DAS AVALIAÇÕES 87

CAPÍTULO

1

INTRODUÇÃO

A World Wide Web disponibiliza uma grande quantidade de informações aos seus usuários. No entanto, muitas das informações apresentadas são redundantes e desinteressantes. Para lidar com esse problema, os sistemas de recomendação (SR) surgiram para filtrar informações, prever classificações e sugerir prováveis itens de preferência para um determinado usuário (RICCI; ROKACH; SHAPIRA, 2015). Essas sugestões de itens estão relacionadas aos processos de tomada de decisão de quais produtos ou serviços consomem, por exemplo: quais músicas ouvir, a qual filme ou série assistir e quais produtos comprar.

O desenvolvimento dos SR partiu do fato que diversos indivíduos, muitas vezes, dependem de recomendações fornecidas por outros na tomada de decisões cotidianas. Por exemplo, é comum confiar no que os colegas recomendam ao selecionar um livro para ler; os empregadores contam com cartas de recomendação em suas decisões de recrutamento; e ao selecionar um filme para assistir, os indivíduos tendem a ler e confiar nos críticos de cinema.

Para o usuário, uma maior facilidade na procura por itens significa ganho de tempo na escolha do mesmo e a possibilidade da descoberta de novos produtos (RICCI; ROKACH; SHAPIRA, 2015). Ao implementar um SR em uma base de dados de uma plataforma com boas recomendações, é possível conseguir um aumento na venda e no acesso de itens. Isso impacta no aumento da satisfação do usuário, pois há uma maior compreensão de seus gostos e maior fidelidade ao receber sugestões de seu interesse.

Há diversas técnicas que podem ser empregadas no desenvolvimento de um SR. As principais são: a filtragem colaborativa, a baseada em conteúdo e a baseada em conhecimento. A filtragem colaborativa faz classificações de vários usuários de maneira colaborativa para prever avaliações ausentes. Na baseada em conteúdo, as notas dos usuários e as descrições de atributos dos itens são aproveitadas para fazer previsões, ou seja, os interesses dos usuários podem ser modelados com base nas propriedades dos itens que eles classificaram e/ou acessaram no passado. Por fim, na baseada em conhecimento os usuários especificam interativamente seus

interesses, e a especificação do usuário é combinada com o conhecimento do domínio para fornecer recomendações (AGGARWAL *et al.*, 2016).

A filtragem colaborativa tem sido a abordagem mais popular e eficiente usada nos SR. Essa técnica usa interações de outros usuários para prever itens relevantes. No entanto, essa abordagem tem limitações, como esparsidade (AGGARWAL *et al.*, 2016; JANNACH *et al.*, 2010), problema muito comum em aplicações diversas. Ou seja, a quantidade de *feedbacks* (notas ou avaliações) totais obtidos na base de dados geralmente é muito pequena se comparada ao número de feedbacks a serem estimados, dificultando o cálculo de boas previsões. Além disso, há chances de não gerar recomendações precisas para um indivíduo com gosto incomum, pois podem não existir usuários semelhantes a ele na base de dados (RICCI; ROKACH; SHAPIRA, 2015). A esparsidade também cria um desafio no cálculo de similaridade quando o número de notas atribuídas aos mesmos itens entre dois usuários é pequena.

Além da questão da esparsidade, outro grande problema muito comum em SR é a partida-fria (*coldstart*), quando o número de notas inicialmente disponíveis a um usuário ou item é relativamente pequeno. Quando o número de classificações inicialmente disponíveis é relativamente pequeno, torna-se mais difícil aplicar os modelos tradicionais de filtragem colaborativa (AGGARWAL *et al.*, 2016). Embora os métodos baseados em conteúdo e em conhecimento sejam mais robustos do que os colaborativos no cenário de partida-fria, informações de conteúdo ou conhecimento podem não estar sempre disponíveis.

Para tentar superar estas limitações, um tipo de aprendizado que tem ganhado destaque nos últimos anos é o aprendizado com dados de dois tipos: com e sem rótulos, chamado Aprendizado Semissupervisionado. Nesta abordagem, além dos dados não-rotulados, são fornecidas para o algoritmo algumas informações de supervisão, mas não necessariamente para todos os exemplos.

1.1 Motivação

O estudo do aprendizado semissupervisionado é motivado por dois fatores: seu valor prático na construção de melhores algoritmos computacionais e seu valor teórico em entender o aprendizado em máquinas e humanos (ZHU; GOLDBERG, 2009). Uma abordagem para este aprendizado é considerá-lo como uma forma especial de classificação, que usa grande quantidade de dados não rotulados, juntamente com os dados rotulados, para construir melhores classificadores (ZHU, 2005). Esse aprendizado é útil em tarefas em que há escassez de dados rotulados. Os rótulos podem ser difíceis de obter porque requerem anotadores (geralmente humanos), dispositivos especiais ou os experimentos são lentos. Além disso, neste tipo de aprendizagem é necessário um grande processamento computacional, requer a construção de muitos modelos que consomem muito tempo (SHEIKHPOUR *et al.*, 2017) e é bastante suscetível a rótulos errados.

Dentre as abordagens de aprendizado semissupervisionado, a técnica de co-treinamento é interessante pois possibilita o classificador ser treinado com uma pequena amostra de dados rotulados, e em seguida, ele é usado para classificar os dados não rotulados (BLUM; MITCHELL, 1998). Tal abordagem assume que os dados podem ser divididos em dois ou mais conjuntos, sendo que cada conjunto é suficiente para treinar um classificador.

Por meio desta técnica, é possível diminuir a esparsidade da matriz de *feedbacks* (usuários $\times$ item) e melhorar a eficácia do recomendador. Embora existam métodos que usam o co-treinamento para diminuir a esparsidade da matriz de notas explícitas (ZHANG *et al.*, 2014; HAO *et al.*, 2015), faltam trabalhos que explorem o co-treinamento no cenário de classificação personalizada para recomendação.

Argumentamos que o co-treinamento pode ser usado para enriquecer conjuntos de dados e auxiliar a aliviar problemas tradicionais de recomendação, como, por exemplo, esparsidade e partida-fria, com a recomendação partindo de um conjunto de dados menos esparso. Este enriquecimento é feito baseado em diferentes e independentes visões para descrever usuários e itens. Por exemplo, informações sobre as avaliações dos usuários podem ser tratadas como uma visão e as descrições dos itens podem ser tratadas como outra visão. Para isso, utilizamos o cenário de previsão de classificação, em que algoritmos de recomendação tentam prever as classificações dos usuários para todos os itens não vistos no conjunto de dados.

Nossa abordagem é capaz de construir diferentes recomendações incorporando diferentes visualizações dos dados, o que ajuda a reduzir a esparsidade e partida-fria usando os abundantes dados não-rotulados, além dos dados rotulados originais disponíveis para treinamento.

No cenário de recomendação com predição de notas, alguns trabalhos se basearam no co-treinamento Zhang *et al.* (2014), Costa, Manzato e Campello (2018), Costa, Manzato e Campello (2019) se basearam no co-treinamento, buscando reduzir o problema de partida-fria e esparsidade. Estes trabalhos não adotam abordagens de ranqueamento, são apenas para o cenário de predição de notas. A proposta desta pesquisa é avaliar quão relevante são itens recomendados em formato de *ranking*, baseado em um conjunto de dados enquecidos pelo co-treinamento. Ao recomendar este *ranking*, o SR tenta prever quais são os produtos ou serviços mais adequados, baseados nas preferências e restrições do usuário.

Para esta tarefa de ranqueamento, na qual são selecionados todos os itens ranqueados que podem satisfazer algumas necessidades do usuário, encontrar apenas alguns itens bons pode ser insuficiente, principalmente quando o número de itens disponíveis com notas é relativamente pequeno. Nesta situação, além do benefício de examinar cuidadosamente todas as possibilidades, o usuário também pode se beneficiar do *ranking* desses itens ou de explicações adicionais que o SR gera.

Dado o exposto, a principal motivação para essa pesquisa é que a abordagem de co-treinamento possa ser usada em conjunto com um SR para obter melhores *rankings*, isto compa-

rado ao aplicar o mesmo SR em um conjunto de dados sem o enriquecimento.

Considerando os problemas previamente apresentados, surgem as seguintes hipóteses:

H1: Realizar uma recomendação a partir de um conjunto de dados enriquecido, por meio de uma técnica de co-treinamento, pode gerar recomendações com boas acurácias, quando comparadas a conjuntos de dados sem enriquecimento;

H2: Este conjunto enriquecido pode ser utilizado para reduzir o problema de partida-fria.

1.2 Objetivos

A fim de validar as hipóteses apresentadas, este trabalho tem como objetivo principal produzir recomendações *top-n* baseadas em uma matriz menos esparsa, obtida por meio de um sistema de co-treinamento. E como objetivo específico produzir recomendações, também baseado neste conjunto enriquecido (menos esparso), com foco no cenário de partida-fria.

Diminuir a esparsidade do conjunto de dados permite que o algoritmo de recomendação faça sua busca em um espaço reduzido nos itens ainda sem avaliação. Dado o alto valor da esparsidade dos conjuntos de dados, este enriquecimento foi orientado a não preencher a matriz de forma significativa. Além disso, no co-treinamento é utilizado uma métrica de confiança que atua em conjunto com a rotulação dos itens não-vistos (sem notas). A recomendação final parte deste conjunto enriquecido, obtendo maior precisão quando comparado ao *baseline*.

1.3 Estrutura do Trabalho

Além deste capítulo introdutório, este trabalho está organizado da seguinte forma. No Capítulo 2 são apresentados conceitos básicos sobre sistemas de recomendação e relacionados com o trabalho proposto. Em seguida, no Capítulo 3, são apresentadas as técnicas de aprendizado semissupervisionado, o co-treinamento, e outras formas de enriquecimento da matriz de notas. No Capítulo 4 apresentamos a arquitetura geral da solução, o pré-processamento realizado, a métrica de confiança usada, o algoritmo de ranqueamento aplicado nos dados enriquecidos e as formas que foram consideradas este conjunto enriquecido. No Capítulo 5 apresentamos de que forma conduzimos os experimentos, as bases de dados utilizadas, validação e geração dos *rankings* para avaliação, além da discussão dos resultados. Por fim, no Capítulo 6 são apresentadas as contribuições da pesquisa, conclusões, submissões de artigos e trabalhos futuros.

CAPÍTULO

2

SISTEMAS DE RECOMENDAÇÃO

Como apresentado no Capítulo 1, o objetivo deste trabalho é gerar recomendações ranqueadas baseadas em uma matriz menos esparsa por meio de um sistema de co-treinamento. Em uma matriz de notas (*ratings*), com os usuários distribuídos, por exemplo, ao longo das linhas e os itens distribuídos ao longo das colunas, cada elemento desta matriz representa a nota que certo usuário atribuiu (ou irá atribuir) para aquele item. Desse modo, será apresentada uma visão geral sobre SR e suas abordagens, os principais conceitos para a proposta e, por fim, as métricas de avaliação usadas em SR.

Os modelos básicos para SR trabalham com dois tipos de dados: as interações de usuário, como as notas ou comportamento de compra; e as informações de atributo sobre os usuários e itens, como perfis textuais ou palavras-chave relevantes. Métodos que usam o primeiro tipo são referidos como métodos de filtragem colaborativa, enquanto os métodos que usam o segundo são referidos como métodos de recomendação baseados em conteúdo. Os sistemas baseados em conteúdo também utilizam as matrizes de avaliações na maioria dos casos, embora o modelo geralmente seja focado nas classificações de um único usuário e não de todos.

Há também os sistemas baseados em conhecimento, que são úteis em contextos nos quais os itens não são adquiridos com muita frequência como imóveis, automóveis e turismo (AGGARWAL *et al.*, 2016). As recomendações são baseadas em requisitos do usuário explicitamente especificados ao invés de usar avaliação histórica ou dados de compra; bases de conhecimento externas e restrições são usadas para criar a recomendação.

Alguns sistemas de recomendação ainda combinam esses diferentes aspectos para criar sistemas híbridos, que podem combinar os pontos fortes de vários tipos de SR para criar técnicas que podem executar de forma mais robusta em uma ampla variedade de configurações.

2.1 Conceitos Gerais

Os SR são um conjunto de ferramentas e técnicas que fornecem sugestões para itens que podem ser usados por um usuário (RICCI; ROKACH; SHAPIRA, 2015). As sugestões visam apoiar seus usuários em vários processos de tomada de decisão como: quais itens comprar, que música ouvir ou quais notícias ler. Estes sistemas emergiram como uma área de pesquisa, em meados dos anos 1990, quando os pesquisadores começaram a se concentrar em problemas de recomendação que dependiam explicitamente da estrutura das avaliações. Os SR provaram ser um meio valioso para os usuários lidarem com a sobrecarga de informações e se tornaram uma das ferramentas mais poderosas e populares no comércio eletrônico.

A ideia básica dos SR é utilizar várias fontes de dados para inferir os interesses dos clientes. A entidade à qual a recomendação é fornecida é chamada de **usuário** e o produto recomendado é chamado de **item**. Portanto, a análise de recomendação é frequentemente baseada na interação anterior entre usuários e itens, pois os interesses do passado são frequentemente bons indicadores de escolhas futuras.

As preferências do usuário podem, por exemplo, ser adquiridas implicitamente pelo monitoramento do comportamento do usuário, como também o SR pode questionar explicitamente ao visitante sobre suas preferências. As interações registradas dos usuários no SR é denominada de transação. Estas transações são registros de informações geradas durante a interação humano-computador e que são úteis para o algoritmo de geração de recomendação que o sistema está usando (AGGARWAL *et al.*, 2016). As avaliações, também chamadas de notas ou *feedbacks*, são a forma mais popular de dados de transação que um SR registra. Estas avaliações podem ser coletadas de maneira explícita ou implícita. Na explícita, o usuário é solicitado a fornecer uma opinião sobre um item em uma escala de avaliação. Nas implícitas são inferidas a partir das ações do usuário. Por exemplo, um usuário que visita uma página de um produto talvez tenha algum interesse naquele produto, enquanto um usuário que de fato compre este produto possui mais interesse.

As notas (ou avaliações) podem assumir várias formas (SCHAFER *et al.*, 2007):

- Avaliações escalares, como de 1 a 5;
- Avaliações ordinais, tais como “concordo totalmente, concordo, neutro, discordo, discordo totalmente” onde o usuário é solicitado a selecionar o termo que melhor indica sua opinião em relação a um item, geralmente por meio de um questionário;
- Avaliações binárias, que modelam as escolhas nas quais o usuário é simplesmente solicitado a decidir se um determinado item é bom ou ruim para ele;
- Avaliações unárias, que podem indicar que um usuário observou ou comprou um item, ou avaliou um item positivamente. Nestes casos, a ausência de uma avaliação indica que não há informações relacionadas ao usuário para o item, ou que, por exemplo, talvez o usuário tenha comprado o item em outro lugar.

Podem ser usadas também outras fontes de informações para os SR atingirem seus objetivos, como informações pessoais do usuário, informações demográficas, espaço-temporais, etc.

Com estas notas disponíveis para os SR, a tarefa é tentar prever as notas que os usuários atribuiriam para itens não vistos e/ou ranquear uma lista de itens inéditos para a recomendação.

No cenário de predição de notas, os trabalhos disponíveis na literatura focam principalmente na predição dos valores de avaliações para os itens que um usuário escolheu. Esse tipo de dado pode ser coletado facilmente, possibilitando o treinamento e teste *offline*, para, respectivamente, ajustar o modelo de recomendação e avaliá-lo. Quando consegue-se estimar estas notas para itens ainda não avaliados, pode-se recomendar a este usuário os itens com a maior avaliação estimada.

A maioria dos modelos voltados a predição de notas tratam o problema de recomendação como um problema de predição, em que o erro quadrado da nota real e a nota prevista é otimizado. Entretanto, os SR raramente apresentam todas as notas ao usuário. Além disso, é mais provável que o usuário preste atenção aos itens no topo da lista do que aos itens de classificação inferior (AGGARWAL *et al.*, 2016). Por isso, na prática, apenas os *top-n* itens são apresentados ao usuário como uma lista ranqueada.

Ranquear todos os documentos dos dados disponíveis para uma certa consulta, de acordo com sua relevância, é o problema central em Recuperação de Informação. É importante ranquear os itens recomendados de acordo com o interesse do usuário. Devido a natureza dos usuários de priorizar a seleção de itens no início de uma lista, tais classificações podem aumentar a confiança na aplicação de recomendação, bem como a vontade de comprar (CHEN; PU, 2005; FELFERNIG *et al.*, 2007). Esta é uma abordagem bastante útil quando a tarefa de recomendação é, para cada usuário, escolher um pequeno número N de itens dentre todos os disponíveis itens na coleção, como as N principais recomendações (*top-N*).

A diferença entre estas duas abordagens é que a predição de notas está preocupada com as notas atribuídas, enquanto o ranqueamento normalmente considera todos os itens da coleção, seja avaliado ou não pelo usuário (STECK, 2013). Além destas abordagens, há atualmente uma grande preocupação em explicar para o usuário o motivo daquelas recomendações, que compõe uma área específica em SR.

Na Subseção seguinte será apresentada uma revisão das principais abordagens de recomendadores.

2.2 Abordagens Principais

A seguir são detalhados alguns aspectos destas abordagens.

2.2.1 Filtragem Colaborativa

Nas abordagens de Filtragem Colaborativa, os algoritmos precisam relacionar duas entidades diferentes, usuários e itens. Na literatura há dois principais métodos (RICCI; ROKACH; SHAPIRA, 2015), a abordagem baseada em memória e a baseada em modelos.

A primeira, também conhecida como método baseado em vizinhança, é dividida em duas abordagens, a baseada no usuário e a baseada em item. Na primeira, o algoritmo usa as interações de usuários semelhantes de um determinado item para prever como este usuário alvo classificaria o item. Na segunda, o algoritmo usa as interações mais semelhantes de um certo item, com isso procura estimar o interesse do usuário por aquele determinado item. Quando o número de usuários é maior que o número de itens, as abordagens baseadas em itens geralmente têm melhor desempenho do que as baseadas no usuário. Além disso, os métodos baseados em itens são mais estáveis pois dois ou mais itens têm maior chance de ter mais usuários que os co-avaliaram (AGGARWAL *et al.*, 2016).

As abordagens baseadas em modelos usam as classificações das matrizes de usuários-itens para aprender um modelo de predição. As características relevantes de usuários e itens são capturadas por um conjunto de parâmetros do modelo, que são aprendidos com os dados de treinamento e, posteriormente, usados para prever novas classificações. Uma técnica muito usada nesta abordagem é um dos modelos de fatores latentes como a fatoração de matriz, conhecida também como SVD (do Inglês, *Singular Value Decomposition*). Este modelo transforma ambos os itens e usuários para o mesmo fator latente (oculto), no qual o espaço latente tenta explicar as classificações, caracterizando ambos os itens e usuários em fatores inferidos automaticamente a partir do feedback do usuário. No domínio de filmes, tais fatores identificados automaticamente podem corresponder a aspectos de um filme, por exemplo o gênero, como drama ou aventura, mas eles também podem não ser interpretáveis (RICCI; ROKACH; SHAPIRA, 2015).

2.2.2 Filtragem Baseada em Conteúdo

Algoritmos de Filtragem Baseada em Conteúdo constroem representações de itens e usuários a partir de suas descrições. Utiliza o histórico de interações do usuário com base nos atributos (metadados) dos itens classificados para criar seu perfil. Em seguida, sugere itens semelhantes àqueles que o usuário apreciou no passado. O perfil é uma representação estruturada dos interesses do usuário, utilizado para recomendar novos itens interessantes. Esta abordagem parte do princípio de que os usuários tendem a se interessar por itens similares aos que demonstraram interesse no passado, definindo então, a similaridade entre os itens (HERLOCKER; KONSTAN; RIEDL, 2000). Esta abordagem é adequada para domínios em que os itens têm muitas informações disponíveis sobre eles, como notícias, páginas da Web, produtos e informações textuais fornecidas pelos usuários que contêm dados não estruturados, como comentários.

Os SR baseados em conteúdo possuem algumas vantagens em relação aos demais paradigmas. Estes sistemas são independentes de outros usuários, pois necessitam apenas do histórico do usuário para gerar um perfil de preferências. Podem apresentar explicações de como certa recomendação foi gerada, por meio das características ou descrições utilizadas no processo. Além disso, novos itens no sistema que ainda não foram avaliados pelos usuários, podem ser recomendados, desde que se encaixem em um perfil de preferências.

Em alguns casos, pode haver maior dificuldade para estabelecer esta similaridade. Para se estabelecer a similaridade entre itens, seria necessário identificar os atributos dos itens a serem adquiridos (marca, cor, preço, peso e etc.). No caso dos itens serem documentos, este processo de comparação pode ser facilitado considerando similares ao compartilharem termos em comum. Sendo assim, esta abordagem é mais indicada para a recomendação de itens textuais, onde o conteúdo é geralmente descrito com palavras-chave.

As maiores desvantagens deste paradigma são a análise limitada de conteúdo e a sobre-especialização (RICCI; ROKACH; SHAPIRA, 2015). Na primeira, itens não podem ser recomendados se não houver informações suficientes que os descrevam. Outro problema é com relação a itens com as mesmas descrições, que serão considerados idênticos. Na sobre-especialização, os recomendadores propõem itens que são, de alguma forma, semelhantes aos que o usuário alvo já classificou positivamente, podendo ocasionar em recomendações óbvias e muito semelhantes àquelas que o usuário já conhece. Isso ocorre quando o sistema só consegue recomendar itens com alta pontuação para um determinado perfil de usuário. Por exemplo, uma pessoa sem experiência com a culinária tailandesa nunca receberia uma recomendação desta, mesmo que seja o melhor restaurante tailandês da região. Neste tipo de problema é comum introduzir alguma aleatoriedade. É sempre desejável ter uma certa quantidade de novidade e serendipidade (capacidade de descobrir coisas boas por mero acaso, sem previsão) nas recomendações (AGGARWAL *et al.*, 2016). Esta novidade se refere ao item que é diferente do que o usuário já viu no passado. De forma similar, serendipidade implica nos itens que o usuário gostaria de descobrir, que sejam extremamente relevantes, mas que não conseguiria encontrar de alguma forma.

2.2.3 Abordagens Híbridas

Os sistemas de recomendação híbridos combinam duas ou mais técnicas de recomendação para obter um melhor desempenho ao invés de cada técnica individualmente. Estes sistemas podem aliviar alguns dos problemas associados à Filtragem Colaborativa e outras técnicas de recomendação. Estas abordagens podem ser divididas em 7 categorias: ponderada, de comutação, mista, combinação de características (*features*), cascata, aumento de recursos e meta-nível (BURKE, 2002). Cada uma delas adota metodologias para utilizar os recomendadores. Por exemplo, na ponderada as pontuações de cada técnica de recomendação são combinadas para produzir uma única recomendação. Na comutação, o sistema alterna entre as técnicas de recomendação, dependendo da situação atual. Na cascata, um recomendador corrige as recomendações dadas

por outro. No caso do aumento de recursos, a saída de uma técnica é usada como entrada de outra. Em sistemas que permitem fazer um grande número de recomendações simultaneamente, pode ser possível usar a do tipo misto, onde mais de uma técnica é apresentada em conjunto. Desta forma, auxilia a evitar o problema de inicialização de um item novo, pois pode-se utilizar a descrição de um item, mesmo que não tenham sido avaliadas por ninguém.

Conteúdos híbridos colaborativos, independentemente do tipo, sempre terão o problema de *ramp-up*, que é o período de aperfeiçoamento do recomendador, desde o seu desenvolvimento inicial até a utilização máxima de sua capacidade, que envolve experimentações, melhorias no recomendador e no processo (BURKE, 2002). Isto ocorre pois as abordagens precisam de um banco de dados de avaliações. Ainda assim, os híbridos são populares porque em muitas situações as avaliações já existem ou podem ser inferidas a partir de dados.

2.3 Principais Conceitos Para a Proposta

Nesta Seção será apresentada uma breve revisão dos conceitos de esparsidade, partida-fria, aprendizado semissupervisionado com a abordagem de co-treinamento; e métricas de avaliação.

2.3.1 Esparsidade

Em aplicações do mundo real, a matriz de usuários $\times$ itens com os valores das notas tendem a ser muito esparsas, já que os usuários geralmente fornecem notas para uma pequena porção dos itens do acervo. Esta tentativa do sistema de tentar calcular boas previsões quando existem poucas avaliações disponíveis provém desta esparsidade dos dados (JANNACH *et al.*, 2010).

O desafio surge devido à falta de disponibilidade de usuários com gostos semelhantes, pois o número de notas no sistema geralmente é muito menor comparado ao número de notas a estimar. Em decorrência disso, pode não ser possível recomendar itens para alguém com gostos únicos, porque não haverá ninguém suficientemente semelhante a este usuário (RICCI; ROKACH; SHAPIRA, 2015). Mesmo em sistemas mais simples, é provável que a matriz seja esparsa, com milhares de linhas e colunas (quantidade de usuários e itens), a maioria das quais são zeros, ou seja, não foi atribuída nenhuma nota.

2.3.2 Partida-Fria

O problema de partida-fria (*coldstart*) refere-se à situação em que um novo usuário ou item acabou de entrar no sistema (SCHEIN *et al.*, 2002). A filtragem colaborativa não consegue gerar recomendações úteis para o novo usuário devido à falta de avaliações ou interações anteriores. Da mesma forma, quando um novo item entra no sistema, é improvável que os

sistemas de filtragem colaborativa o recomendem a muitos usuários, pois nenhum ou poucos usuários avaliaram este item. O problema de partida-fria pode ser visto como um caso especial do problema de esparsidade (HUANG; CHEN; ZENG, 2004), onde a maioria das avaliações em certas linhas ou colunas da matriz de interação são zero.

Uma opção para lidar com este problema é explorar informações adicionais sobre os usuários (categorias), como gênero, idade, escolaridade ou outras informações disponíveis que podem ajudar a classificar o usuário. Com isso, a recomendação é baseada nos atributos semelhantes do perfil do usuário. Esta técnica de recomendação vem da filtragem baseada em conteúdo, que apesar de não possuir o problema de *coldstart*, não há aprendizado sobre os usuários (RICCI; ROKACH; SHAPIRA, 2015) e trabalha somente com categorias, podendo haver uma super-especialização do sistema.

2.3.3 Aprendizado Semissupervisionado

Os algoritmos de aprendizado de máquina estão divididos em três tipos: supervisionado, não-supervisionado e semissupervisionado. No supervisionado, a tarefa de aprendizado envolve um conjunto de dados de treinamento, com seus atributos e a classe (ou rótulo/nota) de cada exemplo, que fará o treinamento para tentar obter uma função que se aproxime da função real, ou seja, uma função que se aproxime dos exemplos, mas que obtenha um aprendizado para ser aplicável a novos exemplos. Após o treinamento, para medir a acurácia desta função, é dado um conjunto de dados de teste, diferente do conjunto de treino. No aprendizado não-supervisionado o agente aprende os padrões na entrada, mesmo que não seja fornecida explicitamente a classe (RUSSELL; NORVIG, 2002). O Aprendizado Semissupervisionado (AS) lida com métodos para explorar automaticamente dados não-rotulados, com adição dos dados rotulados, para melhorar o desempenho de aprendizagem, onde nenhuma intervenção humana é assumida (CHAPELLE; SCHÖLKOPF; ZIEN, 2006; ZHU, 2006). Este aprendizado está entre o aprendizado não-supervisionado e supervisionado, e, por isso, a maioria das estratégias deste aprendizado são baseadas na extensão tanto de um aprendizado supervisionado como de um não-supervisionado, de forma a incluir informações adicionais para o aprendizado. Ou seja, pode utilizar dados rotulados e não-rotulados para obter melhores desempenhos. Esta aprendizagem pode atingir o mesmo nível de desempenho que a aprendizagem supervisionada, mas com menos instâncias rotuladas. Isso reduz o esforço de anotação, consequentemente reduzindo custo.

Um importante campo de pesquisa do AS é o co-treinamento, uma abordagem de aprendizado multivisão proposta originalmente por Blum e Mitchell (1998), onde os dados podem ser particionados em subconjuntos disjuntos, que seriam suficientes para aprender o conceito de destino se dados rotulados estiverem disponíveis (XU; TAO; XU, 2013).

Este aprendizado lida com visões de dados que podem ser descritas em vários espaços de representação, sendo que cada espaço pode ser usado para construir um preditor. Por exemplo, documentos multilíngues podem ser vistos como dados multivisão: cada idioma em que um docu-

mento é traduzido corresponde a uma visão. O objetivo geral desta aprendizagem é combinar os preditores sobre cada visão para melhorar o desempenho geral além dos preditores treinados em cada visão separadamente ou em combinações triviais de visão (USUNIER; AMINI; GOUTTE, 2011).

Em diversas aplicações práticas, rótulos são custosos e demanda-se tempo para obter, mas exemplos não rotulados são baratos e de fácil coleta. Neste cenário, é útil combinar os dados rotulados com os dados não rotulados para o aprendizado da função. A abordagem de aprendizado semissupervisionado pode auxiliar a resolver este problema, aprendendo com poucos dados rotulados. O aprendizado multivisão é uma direção emergente na aprendizagem de máquina que considera a aprendizagem com múltiplas visões para melhorar o desempenho de generalização (USUNIER; AMINI; GOUTTE, 2011).

O co-treinamento considera uma grande amostra não rotulada para poder aumentar o desempenho de um algoritmo de aprendizado quando um pequeno conjunto de exemplos rotulados está disponível. Para isso, os dados são divididos em dois conjuntos distintos (visões dos dados), ou seja, os conjuntos são disjuntos em seus atributos, de forma que seja suficiente aprender o conceito desejado caso estejam disponíveis uma quantidade suficiente de dados rotulados. Por exemplo, nas páginas *Web*, as visões podem ser o texto que estão na página e o texto no *hyperlink* associado; em imagens da *web* as visões podem ser as legendas associadas e o próprio conteúdo visual, conteúdos multimídias podem ser descritos de forma simultânea pelos sinais de vídeo e áudio.

O co-treinamento é utilizado geralmente neste cenário, quando existem apenas pequenas quantidades de dados rotulados, mas grandes quantidades de dados não rotulados, descritos por duas ou mais visões. A única suposição é que espera-se que as diferentes visões sejam independentes e não correlacionadas para garantir o sucesso deste método. Ou seja, as visões são independentes quando é possível realizar a indução de um classificador sem depender da outra visão (SUN *et al.*, 2013; XU; TAO; XU, 2013). Apesar desta forte suposição, há um estudo que mostra um relaxamento desta suposição. No trabalho de Balcan, Blum e Yang (2005) é mostrado um relaxamento da suposição quanto a independência condicional, de forma que é possível expandir esta suposição em um nível mais fraco e é justificado no processo iterativo do co-treinamento.

O surgimento deste aprendizado é amplamente motivado pelo fato dos dados, em várias aplicações, poderem ser descritos por diferentes conjuntos de recursos ou diferentes visões do mesmo conjunto de recursos.

2.3.4 User k-Nearest Neighbors (User-KNN)

Um dos algoritmos de recomendação utilizados no co-treinamento é o User-KNN (AGGARWAL *et al.*, 2016; RICCI; ROKACH; SHAPIRA, 2015), cuja recomendação é baseada em

semelhança de usuários. Dado um usuário alvo, o algoritmo prevê sua nota baseando-se nas notas dos usuários com preferências similares. Este método parte de dois pressupostos (JANNACH *et al.*, 2010), a de que usuários que possuíam gostos similares no passado terão gostos similares no futuro; e que a preferência do usuário se mantém de certa forma estável e consistente ao longo do tempo.

Dado um usuário u e um item i, a nota $\hat{r}_{ui}$ é predita para um par (u,i) desconhecido, considerando as notas dos k usuários mais similares a u que avaliaram o item i. Para obter este conjunto de usuários mais semelhantes ($S^k(i;u)$), uma medida de proximidade $p_{u\upsilon}$ deve ser adotada, como o coeficiente de correlação de Pearson, a similaridade por cosseno, dentre outras. A medida final de similaridade é definida por:

$$s_{u\upsilon} = \frac{n_{u\upsilon}}{n_{u\upsilon} + \lambda_1} p_{u\upsilon}, \tag{2.1}$$

onde $n_{u\upsilon}$ é o número de itens que os usuários u e υ possuem em comum. O λ_1 é uma constante de regularização e foi adotado o valor 100 (KOREN, 2008).

A nota final é predita pela equação:

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{\upsilon \in S^k(i;u)} s_{u\upsilon}(r_{\upsilon i} - b_{\upsilon i})}{\sum_{\upsilon \in S^k(i;u)} s_{u\upsilon}}, \tag{2.2}$$

onde $r_{\upsilon i}$ é a nota que cada usuário υ, k usuários mais similares a u, atribuiu ao item i.

Os termos b_{ui} e $b_{\upsilon i}$ são *baselines* estimados baseados nas avaliações dos usuários para modelos de vizinhança (KOREN, 2010), definido por:

$$b_{ui} = \mu + b_u + b_i, \tag{2.3}$$

onde μ é a nota média global, b_u e b_i são os desvios observados do usuário u e para o item i, respectivamente, em relação à média. Estas estimativas podem ser obtidas ao iterar m vezes as seguintes equações:

$$b_i = \frac{\sum_{i:(u,i)\in D}(r_{ui} - \mu - b_u)}{\lambda_2 + |\{u : (u,i) \in D\}|} \tag{2.4}$$

$$b_u = \frac{\sum_{u:(u,i)\in D}(r_{ui} - \mu - b_i)}{\lambda_3 + |\{i : (u,i) \in D\}|} \tag{2.5}$$

onde D é o conjunto com todas as notas conhecidas, ou seja, $D = \{(u,i) : r_{ui}$ é conhecido$\}$. As constantes λ_2 e λ_3 foram estabelecidas em 10 e 15, respectivamente (COSTA; MANZATO; CAMPELLO, 2018).

2.3.5 Item k-Nearest Neighbors (Item-KNN)

No método baseado em vizinhança de itens são identificados, para um item alvo, as avaliações dos usuários sobre esses itens mais semelhantes. Por exemplo, itens semelhantes a um filme de ação (item alvo) podem ser um conjunto de outros filmes de ação. Diferentemente do User-KNN, no qual podem haver entre os usuários alguma intersecção de interesses, mas não em sua totalidade. Por isso, os métodos baseados em vizinhança de itens apresentam recomendações mais relevantes (AGGARWAL *et al.*, 2016). Este método também pode fornecer uma razão concreta para a recomendação. Por exemplo, a Netflix[1] costuma fornecer uma lista de recomendação com explicações do tipo: *Porque você assistiu a "Breaking Bad"*.

Entretanto, este método pode recomendar itens pouco diversos. Recomendações com maior diversidade incentiva o acaso, podendo ser descobertos itens interessantes e surpreendentes. Os métodos baseados em itens às vezes podem recomendar itens óbvios ou itens que não são tão novos em relação a experiências anteriores do usuário.

A formulação e o cálculo é similar ao User-KNN. A principal diferença entre as duas técnicas de vizinhança é que o Item-KNN prediz uma nota desconhecida r_{ui} de um item i por um usuário u baseado nas notas dos k itens mais semelhantes:

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in S^k(i;u)} s_{ij}(r_{uj} - b_{uj})}{\sum_{j \in S^k(i;u)} s_{ij}} \tag{2.6}$$

onde também foi adotado uma medida de similaridade, s_{ij}, definida por:

$$s_{ij} = \frac{n_{ij}}{n_{ij} + \lambda_1} p_{ij} \tag{2.7}$$

A medida de proximidade p_{ij} aplicada entre os vetores de *ratings* e n_{ij} o número de usuários que avaliaram ambos os itens i e j. O valor de λ_1 é o mesmo adotado na Equação 2.1.

2.3.6 Avaliação em Sistemas de Recomendação

Dado um conjunto de algoritmos de recomendação, é necessário de alguma forma mensurar seu desempenho e eficácia. Os SR compartilham várias semelhanças conceituais com o problema de modelagem em classificação e regressão. Nestas modelagens, no cenário de predição de notas, a variável de classe ausente precisa ser prevista a partir dos exemplos rotuladas. Qualquer uma das entradas da matriz pode estar faltando e precisa ser prevista de forma orientada aos dados, a partir das entradas observadas na matriz remanescente. Desta forma, o problema de recomendação pode ser visto como uma generalização do problema de classificação (JANNACH *et al.*, 2010).

[1] https://www.netflix.com

Os sistemas de recomendação podem ser avaliados usando métodos *online* ou *offline*. Em um sistema *online*, as reações do usuário são medidas em relação às recomendações e, portanto, a participação do usuário é essencial nestes sistemas. Por exemplo, em uma avaliação *online* de um SR de vídeos, pode-se medir a taxa de conversão de usuários que clicam nos vídeos recomendados. Esses métodos de teste são referidos como teste A/B (AGGARWAL *et al.*, 2016). Este teste mede o impacto direto do SR sobre o usuário final para comparar dois algoritmos segmentando os usuários em, pelo menos, dois grupos A e B. É usado um algoritmo para o grupo A e outro algoritmo para o grupo B por um período de tempo, no final do processo é comparado a taxa de conversão (ou outra métrica) dos dois grupos. Testes como este buscam aumentar a taxa de conversão em itens lucrativos, que é o objetivo mais importante de um SR e pode fornecer uma medida real da eficácia do sistema. Porém, as avaliações *online* exigem participação ativa do usuário, o que muitas vezes não é viável seu uso em *benchmarking* e pesquisa.

O teste em vários conjuntos de dados é importante para assegurar maior poder de aprendizado do SR. Com este aprendizado, aumenta-se a confiança de que o algoritmo funciona em outras configurações. Nesses casos, avaliações *offline* com conjuntos de dados históricos são usadas. Os métodos *offline* são os mais comuns para avaliação de SR a partir de uma perspectiva de pesquisa e prática.

Há diversos critérios que normalmente são consideradas para decidir qual abordagem de recomendação será usada, como preferência do usuário, precisão na predição, cobertura da proporção de itens que o SR consegue recomendar, serendipidade, diversidade, risco, privacidade e outros (RICCI; ROKACH; SHAPIRA, 2015). Como a precisão de previsão de um sistema de recomendação, especialmente em sistemas de filtragem, em muitos casos, cresce com a quantidade de dados, alguns algoritmos podem fornecer recomendações com alta qualidade, mas apenas para uma pequena porção do itens onde eles têm grandes quantidades de dados. Isso é muitas vezes referido como a cauda longa ou problema de cauda pesada, onde a grande maioria dos itens foram selecionados ou classificados apenas por um punhado de usuários, mas a quantidade total de evidências sobre esses itens pouco populares é muito maior do que a evidência sobre os poucos itens populares.

Cada aplicação demanda uma necessidade diferente, então o projetista do sistema deve decidir quais propriedades usar. Ao sugerir um método que melhore alguma propriedade, deve-se também avaliar como as mudanças nesta propriedade afetam a experiência do usuário, seja por meio de um estudo do usuário ou por meio de experimentações. Nos experimentos geralmente é utilizado um único método de recomendação com um parâmetro ajustável, que afeta a propriedade sendo considerada. Por exemplo, pode-se visualizar um parâmetro que controla a diversidade da lista de recomendações. Então, os assuntos devem ser apresentados com recomendações baseadas em uma variedade de valores para este parâmetro e medir o efeito do parâmetro na experiência do usuário. Com isso, deve-se medir se a mudança na propriedade afetou sua interação com o sistema. Uma vez que se compreende de que forma os efeitos das propriedades específicas do

sistema afetam a experiência do usuário em questão, pode-se alterar estas propriedades para selecionar um recomendador.

A acurácia é a propriedade mais discutida na literatura em SR. Na grande maioria dos sistemas de recomendação, há um preditor que pode prever opiniões de usuários sobre itens, como avaliações de músicas, ou a probabilidade de uso, como a compra de um item. Esta propriedade é a preocupação de diversos pesquisadores, pois fornecer previsões mais precisas implica na predileção do usuário naqueles sistemas.

Há três grandes classes de medidas de precisão de previsão: a acurácia das previsões de avaliações, acurácia das classificações e acurácia dos ranqueamentos de itens (JANNACH *et al.*, 2010). Para o primeiro, em algumas aplicações, deseja-se saber qual avaliação um usuário daria para um item, como notas de 1 a 5 em um filme. Para este caso, procura-se avaliar a precisão das avaliações previstas do sistema. As métricas mais populares são a raiz do erro médio quadrático (do Inglês, *Root Mean Square Error* (RMSE)) e o erro absoluto médio (do Inglês, *Mean Absolute Error* (MAE)). Para o segundo, utiliza-se *Precision* e *Recall* para identificar os itens mais relevantes para um usuário. Por fim, para a acurácia dos ranqueamentos é comum utilizar o MAP (*Mean Average Precision*) e o NDCG (*Normalized Discounted Cumulative Gain*).

2.3.6.1 RMSE e MAE

Suponha que em uma matriz de avaliações $\mathbf{R}$, r_{uj} é a nota conhecida do usuário u para o item j, e o algoritmo de recomendação estimou a nota $\hat{r}_{uj}$. Considerando o erro da estimativa dado por $e_{uj} = \hat{r}_{uj} - r_{uj}$, o erro global no conjunto E de entradas na matriz de avaliações na qual a avaliação é realizada, medido pelo erro quadrático médio (do Inglês, *Mean squared error* (MSE)) é:

$$MSE = \frac{\sum_{(u,j)\in E} e_{uj}^2}{|E|}$$

Valores menores do MSE indicam performances superiores. A raiz quadrada desta medida é denominada RMSE:

$$RMSE = \sqrt{\frac{\sum_{(u,j)\in E} e_{uj}^2}{|E|}}$$

O RMSE está em unidades de avaliações, em vez de unidades de avaliações quadradas como o MSE. O RMSE tende a penalizar, de forma desproporcional, grandes erros por causa do erro ao quadrado dentro do somatório.

Já o MAE não penaliza de forma desproporcional grandes erros, por não conter o erro da estimativa elevado ao quadrado:

Tabela 1 – Classificações de possíveis resultados da recomendação de um item para um usuário

	Recomendado	Não Recomendado
Consumido	Verdadeiro Positivo (TP)	Falso Negativo (FN)
Não Consumido	Falso Positivo (FP)	Verdadeiro Negativo (TN)

$$MAE = \frac{\sum_{(u,j)\in E} |e_{uj}|}{|E|}$$

Sobre a qualidade das medidas de avaliação RMSE e MAE, o RMSE soma os erros ao quadrado, sendo afetado de forma mais significativa por grandes valores de erro ou *outliers*. Ou seja, caso haja no conjunto de dados algumas previsões mal feitas, isso poderá arruinar a medida RMSE. Em aplicações onde a robustez da predição é importante, o RMSE pode ser uma medida mais apropriada. Já o MAE é um reflexo da precisão quando a importância de *outliers* na avaliação é limitada. O RMSE não reflete de forma real o erro médio, que pode direcionar a resultados equivocados (AGGARWAL *et al.*, 2016). Por isso, a escolha depende da aplicação em questão.

2.3.6.2 Precision e Recall

Na acurácia das previsões, o SR não busca prever as notas dos itens que o usuário daria, mas procura recomendar aos usuários itens que eles podem se interessar (JANNACH *et al.*, 2010). Por exemplo, quando um vídeo é selecionado para visualização no Youtube[2], a plataforma sugere um conjunto de vídeos que também podem ser interessantes. Neste caso, a preocupação é se o sistema prevê corretamente que o usuário assistirá a esses vídeos recomendados. Em uma avaliação *offline*, há normalmente um conjunto de dados de itens que cada usuário consumiu. Em seguida, seleciona-se um usuário de teste, ocultam-se algumas de suas seleções e é solicitado ao revisor para prever um conjunto de itens que o usuário usará. Depois, há quatro resultados possíveis para os itens recomendados e ocultos, conforme mostrado na Tabela 1.

No caso *offline*, pode haver a suposição de que itens não consumidos não seriam usados, mesmo se eles tivessem sido recomendados, sendo desinteressantes para o usuário. Esta suposição pode ser falsa, da mesma forma para itens interessantes que o usuário não selecionou. Por exemplo, um usuário pode não ter visto um item pois não o conhecia, mas após o recomendador apresentar este item, o usuário pode decidir ver. Pode-se contar o número de exemplos que se enquadram em cada célula da Tabela 1 e calcular as seguintes quantidades:

$$Precision = \frac{TP}{TP+FP}$$

[2] https://www.youtube.com/

$$Recall = \frac{TP}{TP+FN}$$

Listas de recomendação mais longas geralmente melhoram o *Recall* e reduzem a Precisão. Em sistemas nos quais a quantidade de recomendações é pré-determinada, a medida mais útil é a Precisão em N, normalmente descrita por $Precision@N$ ou somente $P@n$.

Nas aplicações onde o número de recomendações apresentadas para o usuário não são determinados previamente, procura-se utilizar algoritmos que avaliam em um intervalo de comprimentos de listas de recomendação, em vez de usar um comprimento fixo. Desta forma, pode-se calcular curvas que comparam Precisão ao *Recall* (ou a taxa de verdadeiros positivos à taxa de falsos positivos). Curvas do primeiro tipo são conhecidas como curvas *Precision-Recall*. As do segundo são conhecidas como curva ROC (*Receiver Operating Characteristic*), relação entre os Verdadeiros Positivos e os Falsos Positivos. Ambas as curvas medem a proporção de itens preferidos que são recomendados, porém, a primeira enfatiza a proporção de itens que são preferidos, as curvas ROC enfatizam a proporção de itens que não são preferidos que acabam sendo recomendados. A seleção da curva a ser usada, *Precision-Recall* ou ROC, deve ser baseada nas propriedades do domínio e o objetivo da aplicação.

2.3.6.3 Mean Average Precision (MAP)

Uma métrica para avaliar uma lista ranqueada é a Média de Precisão (*Mean Average Precision* (MAP)). Esta medida procura sumarizar todos os valores $P@n$. O *Average Precision* (AP) é definida como uma média para todos os valores de $P@n$ para todos os documentos relevantes. A Precisão é calculada para cada item relevante encontrado na lista, considerando o item encontrado e os anteriores. Dessa maneira, o MAP é inicialmente calculado após percorrer os k primeiros itens da lista, sendo computado a partir da média aritmética dos valores de Precisão obtidos. Finalmente, o MAP é obtido pela média aritmética das Médias de Precisão de cada lista de recomendação.

$$MAP(Q) = \frac{1}{|Q|}\sum_{j=1}^{|Q|}\frac{1}{m_j}\sum_{k=1}^{m_j} Precision(R_{jk})$$

onde Q é o conjunto de recomendações que o sistema computou, m representa os itens relavantes da lista de recomendações j, e R_{jk} é cada item relevante do conjunto de itens relevantes de recomendações j.

2.3.6.4 Normalized Discounted Cumulative Gain (NDCG)

Geralmente os usuários dão mais importância para os primeiros itens da lista de recomendações. Os itens não relevantes que ocupam as primeiras posições da lista são considerados como erros mais graves que aqueles que estão no final da lista. Uma medida de classificação que está

entre as mais utilizadas é o Ganho Acumulado Descontado (do Inglês, *Discounted Cumulative Gain* (DCG)) (JÄRVELIN; KEKÄLÄINEN, 2000). Essa métrica assume uma decomposição logarítmica no interesse dos usuários à medida que se afastam do topo da lista de recomendações. Assumindo que cada usuário u tem uma relevância $g_{u,i}$ por ter sido recomendado um item i, a média do Ganho Acumulado Descontado (RICCI; ROKACH; SHAPIRA, 2015) para uma lista de J itens é definida como:

$$DCG = \frac{1}{N}\sum_{u=1}^{N}\sum_{j=1}^{J}\frac{g_{u,i_j}}{log_b(j+1)}$$

onde i_j é o item na posição j na lista. A base b do logaritmo é um parâmetro livre, normalmente entre 2 e 10. O logaritmo com base 2 é comumente usado para assegurar que todas as posições são descontadas. O Ganho Acumulado Descontado Normalizado (do Inglês, *Normalized Discounted Cumulative Gain* (NDCG)) é a versão normalizada do DCG, dada por:

$$NDCG = \frac{DCG}{DCG*}$$

onde DCG* é o DCG ideal.

Medidas como o NDCG, são mais informativas do que medidas que utilizam relevâncias binárias, como o AP, uma vez que respondem a qualquer inversão entre documentos de diferentes graus de relevância (LIU *et al.*, 2009). Além disso, dadas duas medidas binárias (ou multinível), uma medida pode ainda ser mais informativa do que a outra pelas seguintes razões: algumas medidas respondem somente para inversões entre documentos em algumas partes específicas do *ranking*, geralmente no topo; e mesmo que duas medidas dependam da mesma parte do ranqueamento, pode ser insensível a algumas destas inversões dentro desta parte devido a ignorar algumas inversões ou devido às funções de desconto usadas.

Nem sempre é a melhor escolha otimizar uma medida menos informativa, mesmo que eventualmente seja usada para avaliação. Por exemplo, otimizar diretamente o AP pode levar a um melhor desempenho do ranking em termos de $P@10$ do que otimizar diretamente $P@10$.

2.4 Considerações Finais

Conforme visto neste capítulo, os SR possuem uma ampla gama de abordagens e técnicas para gerar recomendações. Uma categoria bem conhecida em SR é a predição de notas. Quando consegue-se estimar estas notas dos itens ainda não avaliados, pode-se recomendar a este usuário os itens com a maior avaliação estimada. Os algoritmos da Filtragem Colaborativa são os mais utilizados em sistemas de recomendação, mas tais algoritmos enfrentam alguns problemas como grandes conjuntos de dados, partida fria e matrizes esparsas. Em determinados problemas, o

usuário tem a tendência de optar pelos primeiros itens apresentados em uma lista. A ordem desta lista e quais itens são apresentados utilizam conceitos e técnicas de outra grande categoria em SR, a de ranqueamento. É de grande importância ranquear os itens recomendados de acordo com a utilidade para o usuário, podendo aumentar a confiança na aplicação de recomendação.

Com o aprendizado semissupervisionado há a vantagem de se utilizar tanto dados rotulados como não rotulados nos treinamentos. O co-treinamento pode ser utilizado neste cenário, quando existem apenas pequenas quantidades de dados rotulados, mas grandes quantidades de dados não rotulados, dadas as premissas descritas na Subseção 2.2. Com isso, reduz-se o esforço de anotação, consequentemente reduzindo custo.

Para avaliar o desempenho da recomendação existem, como apresentado, técnicas para os diferentes aspectos dos sistemas de recomendação voltados tanto para predição de notas quanto ranqueamento. O primeiro está intimamente relacionado com a modelagem de classificação e regressão, enquanto o segundo está intimamente relacionado com a avaliação da eficácia de recuperação em busca e aplicativos de recuperação de informações (AGGARWAL *et al.*, 2016).

Este trabalho pretende utilizar conceitos de sistemas baseados em um aprendizado semissupervisionado, utilizando um *framework* de co-treinamento, juntamente com um *ensemble* para combinar estas listas ranqueadas, com intuito de gerar boas recomendações. No capítulo seguinte serão apresentados os conceitos referentes ao enriquecimento de dados e trabalhos relacionados.

CAPÍTULO

3

ENRIQUECIMENTO DE CONJUNTOS DE DADOS

Como argumentado anteriormente, a alta esparsidade na matriz de notas é um problema comum em sistemas de recomendação, pois dificulta a geração de relacionamentos item-para-item e usuário-para-usuário, tornando difícil a recomendação, especialmente para algoritmos baseados em filtragem colaborativa. Este problema é chamado de partida-fria. A definição de esparsidade da matriz de *feedbacks* (JANNACH *et al.*, 2010) é:

$$Esparsidade = \left(1 - \frac{\text{Quantidade de Notas}}{\text{Quantidade de Usuários} \times \text{Quantidade de Itens}}\right) 100\% \tag{3.1}$$

Neste capítulo serão apresentadas algumas técnicas de enriquecimento do conjunto de dados, estes enriquecimentos fazem a rotulação de itens não vistos e, consequentemente, diminuem a esparsidade dos dados; além dos trabalhos relacionados a esta pesquisa.

Usuários ou itens são considerados frios se eles têm poucas ou nenhuma interação, sendo um entrave para que o sistema seja capaz de recomendar um item frio ou para fornecer sugestões adequadas a um usuário frio. O problema de partida fria do item, também chamado de problema do item novo, pode ser suavizado agregando informações do item em seu cálculo, realizando assim uma solução de recomendação híbrida. Para resolver esses problemas, descrevemos neste capítulo possibilidades de enriquecer a matriz de usuários $\times$ itens com informações confiáveis, dentre elas o Co-Treinamento, uma estratégia de aprendizado semissupervisionado que é capaz de rotular relevantes pares de usuário-item para melhorar a qualidade da recomendação.

3.1 Imputação

Imputação é um método geral e flexível para lidar com o problema de falta de dados. Esta técnica requer um método para criar uma distribuição preditiva com base nos dados observados.

A filtragem colaborativa pode ser vista como um caso especial deste problema, em que a matriz de dados é muito grande e esparsa (AGGARWAL *et al.*, 2016).

Existem duas abordagens genéricas para gerar esta distribuição, a modelagem explícita e a implícita (LITTLE; RUBIN, 2019). Na primeira a distribuição preditiva é baseada em uma métrica (média) ou um modelo estatístico (por exemplo, distribuição normal) e, por isso, as suposições são explícitas. Na segunda o foco está em um algoritmo, o que pode implicar em um modelo subjacente. As suposições estão implícitas, mas ainda precisam ser avaliadas para garantir que sejam razoáveis.

Preencher valores ausentes com a Imputação é, em certo nível, plausível. Por exemplo, preencher as notas faltantes de um item ou usuário com o valor médio. Alguns trabalhos (GHAZANFAR; PRUGEL, 2013; HWANG *et al.*, 2014) aplicaram imputação em uma matriz esparsa, para, com isso, diminuir a dimensionalidade do conjunto de dados e aliviar tanto o problema de esparsidade quanto o de partida-fria.

Infelizmente essas técnicas também estão sujeitas a erros de estimativa com o aumento da dimensionalidade e incompletude. Isso ocorre porque, quando uma grande porcentagem das entradas está faltando, cada atributo pode ser estimado com um grau de precisão muito menor. Além disso, alguns atributos podem ser estimados com um grau de precisão muito menor do que outros, e não há como saber a priori quais estimativas são mais precisas. (AGGARWAL; PARTHASARATHY, 2001)

Uma limitação importante dos métodos de imputação simples é que as fórmulas de variância de amostragem padrão, aplicadas ao preenchimento dos dados, subestimam sistematicamente a verdadeira variância de amostragem das estimativas (LITTLE; RUBIN, 2019), mesmo estando correto o modelo usado para gerar as imputações. Outro problema é que as inferências, com base nos dados preenchidos, não levam em consideração a incerteza de imputação, subestimando erros padrão calculados a partir dos dados preenchidos. Estas técnicas, apesar de serem de baixo custo computacional, estão sujeitas a erros de estimativa e incompletude. Devido à grande porcentagem de notas ausentes, a estimativa pela imputação pode ter baixa precisão.

Uma abordagem para estes problemas é a imputação múltipla, que se refere ao procedimento de completar m valores para cada item ausente, criando m conjuntos de dados completos. Entre esses conjuntos de dados completos, os valores observados são iguais, mas os valores ausentes são preenchidos com diferentes imputações para refletir os níveis de incerteza (RICCI; ROKACH; SHAPIRA, 2015). A desvantagem da imputação múltipla, em relação à imputação única, é que exige-se mais trabalho para criar as imputações e análise dos resultados, somado a um acréscimo no armazenamento de dados.

Aplicar a imputação para enriquecer a matriz pode gerar dados consideravelmente distorcidos e aumentar significamente a quantidade de dados (AGGARWAL *et al.*, 2016), além de não considerar, em alguns casos, a personalização dos gostos dos usuários.

3.2 Side Information

Outra técnica que auxilia na diminuição da esparsidade de um conjunto de dados é a de informação secundária (*side information*), que explora informações auxiliares de usuários ou itens para melhorar a precisão da recomendação.

Estas informações podem ser, por exemplo, a forma de interação do usuário, como notas, cliques ou compras. Podem também estar inclusos nestes atributos do usuário informações como gênero, idade e hobbies. Quanto aos itens, seus atributos estão relacionadas com suas propriedades, como categoria ou conteúdo (SHI; LARSON; HANJALIC, 2014).

Com o surgimento das redes sociais, possibilitou aos sistemas de recomendação a obter informações de relacionamento entre usuários, relações estas de confiança e desconfiança. Estas relações são geralmente representadas em um grafo assimétrico usuário-usuário (GUHA *et al.*, 2004) que apresenta, dentro do conjunto de relações que possui, se um usuário confia ou não no outro.

As redes sociais são uma fonte valiosa de *side information* sobre os usuários para a filtragem colaborativa. Esta área, chamada de recomendação social, procura incorporar aspectos sociais, observando as conexões entre os usuários e recomenda diferentes itens baseados nas preferências dos usuários destas conexões. As redes sociais baseadas em localização, por exemplo, permitem analisar atividades online dos usuários e seus padrões de mobilidade baseados nas categorias das localizações (LIU *et al.*, 2013). Estas categorias podem ser hierarquizadas, como fazia o Foursquare[3]. No primeiro nível o usuário escolhia entre vida noturna, comida, entre outros. No segundo nível o tipo de comida ou o tipo de evento. No terceiro categorias mais específicas, como comida japonesa ou tailandesa. Esta categorização implica em um significado semântico da localização, refletindo a preferência do usuário em certas categorias, mediante o seu *check-in*.

Abordagens mais recentes são principalmente baseadas em memória, que são ineficazes devido ao custo da análise nos espaços do usuário ou item. Assim, muitos algoritmos de recomendação com *side information* são desenvolvidos com base em outros modelos, incluindo regra de associação, agrupamento, modelo de tópico, regressão, fatoração, aprendizagem de representação e rede neural profunda (GUO; SUN; THENG, 2019).

3.3 Co-Treinamento

O co-treinamento (BLUM; MITCHELL, 1998) é uma técnica de aprendizado semissupervisionado e representa os dados por múltiplas visualizações distintas. Este aprendizado explora automaticamente dados não rotulados, com a adição de dados rotulados, para melhorar a precisão do aprendizado, onde nenhuma intervenção humana é assumida (CHAPELLE; SCHOLKOPF;

[3] https://foursquare.com/

ZIEN, 2009; ZHU; GOLDBERG, 2009). Com isso, é possível se beneficiar de conjuntos de dados reais, nos quais é muito comum que eles tenham apenas uma pequena quantidade de dados rotulados e uma grande quantidade de dados não rotulados. Esta abordagem assume que os recursos podem ser divididos em dois ou mais conjuntos disjuntos (visão dos dados), sendo que cada conjunto deve ser suficiente para treinar um classificador e os conjuntos sejam condicionalmente independentes.

Apesar deste método assumir que cada exemplo é descrito usando duas visões diferentes, que fornecem informações complementares, o co-treinamento também pode ser realizado quando há apenas uma representação dos dados, se tal representação única for processada por modelos de predição independentes, como dois diferentes classificadores (XU; TAO; XU, 2013). Neste caso, o método explora duas visões da mesma representação única dos dados por meio de dois classificadores usando os exemplos de treinamento disponíveis.

A ideia deste aprendizado é que, em vez de tentar maximizar diretamente a concordância de sua hipótese h com a função alvo f, visa aprender duas hipóteses $h1$ e $h2$, uma sobre cada visão dos dados para, em seguida, maximizar a concordância das hipóteses uma com o outra. O ponto principal é que isso pode ser feito com dados não rotulados. Desta forma, qualquer par de hipóteses que concordam intimamente entre si e satisfazem outras condições simples, como ser mais ou menos equilibrado em suas previsões e concordando com uma pequena amostra rotulada, também deve concordar com a função de destino (BLUM; MANSOUR, 2017).

O co-treinamento aprende um modelo de predição, por exemplo um classificador, para cada visualização usando exemplos rotulados. As previsões mais confiáveis de cada modelo no conjunto não rotulado são usadas para obter conjuntos de treinamento rotulados adicionais, de forma iterativa, para refinamento adicional da outra visão. O algoritmo treina alternativamente para maximizar o acordo mútuo das duas visões distintas de dados não rotulados. Ao minimizar o erro dos exemplos rotulados e maximizar a concordância com os exemplos não rotulados, o algoritmo obtém um classificador preciso em cada visualização.

Nesta técnica, inicialmente dois classificadores separados são treinados com dados rotulados, nos dois subconjuntos, respectivamente. Cada classificador então classifica os dados não rotulados e "ensina"o outro classificador com os poucos exemplos não rotulados (e os rótulos previstos) em que eles julgam ser mais confiantes. Cada classificador é retreinado com os exemplos adicionais de treinamento fornecidos pelo outro classificador e o processo se repete (XU; TAO; XU, 2013).

Os passos deste método são os seguintes:

1. Dividir os dados de treino em dois conjuntos, f_1 e f_2, iguais inicialmente;
2. Treinar dois classificadores independentes M_1 e M_2, que usam os conjuntos f_1 e f_2, respectivamente;
3. Classificar as instâncias sem rótulo com M_1 e M_2 separadamente;

4. Adicionar as k previsões mais confiáveis do M_1 aos dados de treinamento de M_2 e vice-versa;
5. Remover estas previsões dos dados não rotulados;
6. Repetir a partir do passo 2.

Cada classificador somente presta atenção para o seu conjunto de treino, ignorando o outro classificador. Como os dois classificadores são independentemente construídos em diferentes conjuntos de *features*, essa abordagem evita *overfitting*, pelo menos em algum grau, devido à natureza disjunta dos dados utilizados pelos dois classificadores (AGGARWAL, 2014).

Recentemente, existem trabalhos considerando um relaxamento das suposições, como permitir uma fraca dependência entre as duas visões (BLUM; MANSOUR, 2017) ou relaxamento da condição de não correlação (WANG; ZHOU, 2013).

Neste trabalho foi utilizado o CoRec (COSTA; MANZATO; CAMPELLO, 2018), uma extensão de técnica original, que prevê notas em SR, possibilitando a diminuição a dispersão da matriz de *feedbacks*. Em vez de incluir todos os pares não observados de usuário-item no conjunto inicial não rotulado, que é fornecido para os recomendadores, que imporia uma carga computacional adicional desnecessária e desaceleraria a convergência do processo de co-treinamento, foi selecionado aleatoriamente X pares não rotulados (u,i) por usuário. O algoritmo finaliza quando todos os dados não rotulados de ambas as visualizações forem rotulado. Por fim, os dois conjuntos enriquecidos, um de cada recomendador, são combinador em um conjunto único. Este conjunto enriquecido é a união do conjunto de treino com o conjunto predito.

Embora existam métodos que usam o co-treinamento para diminuir a dispersão na matriz de classificação explícita (ZHANG *et al.*, 2014; HAO *et al.*, 2015), faltam trabalhos que explorem o co-treinamento no cenário de recomendação.

3.4 Ensembles

Construir um conjunto de classificadores (*ensemble*) permite, a partir dos dados de treinamento, agregar as predições em uma saída única e mais robusta, comparada com a saída individual de cada classificador. O *ensemble* de classificadores funciona desde que exista a independência destes classificadores (RICCI; ROKACH; SHAPIRA, 2015).

Utilizar uma combinação de algoritmos de filtragem colaborativa costuma superar qualquer recomendador, também de filtragem colaborativa, de forma isolada (JAHRER; TÖSCHER; LEGENSTEIN, 2010). Por exemplo, Bell, Koren e Volinsky (2007) usaram uma combinação de 107 métodos diferentes em sua solução, que foi a ganhadora para o desafio da Netflix[4]. A acurácia na predição das notas foi melhorada substancialmente ao combinar vários preditores. Por isso, os autores sugerem que os esforços devem ser voltados para diferentes abordagens, em

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

vez de refinar uma única técnica.

3.5 Trabalhos Relacionados

Nesta Seção serão apresentados resumos de alguns trabalhos relacionados a este projeto. Estes trabalhos apresentam diferentes abordagens, tanto no contexto de recomendação quanto de enriquecimento, com as técnicas de aprendizado multivisão, co-treinamento e *ensemble*.

No trabalho de Costa, Manzato e Campello (2018) foi proposto uma abordagem de co-treinamento (CoRec) para prever exemplos não-rotulados e aumentar a precisão dos resultados de recomendação. O trabalho visou o cenário de predição de notas (*ratings*) atribuídas aos itens pelos usuários, no qual os diferentes algoritmos de recomendação tentam prever as classificações dos usuários para todos os itens não vistos no conjunto de dados. Motivados pela capacidade da abordagem reduzir significativamente os problemas de partida fria e esparsidade, a abordagem usa apenas um único conjunto de treinamento como dados de entrada e dois algoritmos de recomendação distintos como visões diferentes, combinando a saída de cada recomendador em um *ensemble*. A intenção de combinar estas saídas é melhorar os resultados obtidos por cada modelo individualmente (AGGARWAL *et al.*, 2016).

Em outro trabalho dos autores supracitados, com intuito de combinar e melhorar os modelos individuais gerados pelo CoRec, propuseram o ECoRec (COSTA; MANZATO; CAMPELLO, 2019). O trabalho se assemelha pelo enriquecimento, também com co-treinamento, e pelo uso da combinação dos dados de saída dos recomendadores. A proposta também usa apenas um único conjunto de treinamento como dados de entrada, mas com três algoritmos de recomendação distintos como visões diferentes. Na fase de co-treinamento, cada recomendador é treinado com o objetivo de concordar com as previsões de outros recomendadores, de modo que eles aprendam não apenas com o conjunto de treinamento original de dados rotulados, mas também com as previsões mais confiantes de cada um dos dados não rotulados. A abordagem apresentou vantagens como reduzir a esparsidade e partida-fria da base de dados original, admitindo certa confiança, baseada nas múltiplas visualizações de diferentes algoritmos de recomendação. Da mesma forma que no ECoRec, nosso trabalho utilizou o co-treinamento para diminuir a esparsidade e aliviar a partida-fria, mas recomendando itens não vistos na forma de uma lista ranqueada, além de considerar partições com a quantidade de interações dos usuários para a partida-fria.

O uso de dados de clique registrado pelo motor de pesquisa nas consultas enviadas pelos usuários, como um *feedback* de relevância implícita nos resultados da pesquisa (TAN *et al.*, 2004), foi alvo de pesquisa para outro trabalho que também utiliza o co-treinamento, porém com ranqueamentos. Foi adotada esta abordagem pois os usuários geralmente não estão dispostos a dar *feedbacks* devido à preocupações com privacidade e do esforço exigido. O algoritmo proposto usa os cliques de dados como entrada e gera ranqueamentos adaptáveis como uma saída em um processo de aprendizado. Este algoritmo está em uma estrutura de co-treinamento,

para tornar o processo de aprendizagem eficiente mesmo quando a quantidade de conjuntos de dados de treinamento é relativamente pequena e esparsa. Este trabalho se assemelha pelo uso de algoritmos de ranqueamento em casa visão dos dados no co-treinamento. Porém, utiliza dados implícitos registrados pelos cliques ao longo da navegação; utiliza o mesmo algoritmo de ranqueamento nas visões dos dados, o *Ranking SVM* e obtém um *ranking* final sem utilizar um *ensemble*. No nosso trabalho, aplicamos um recomendador no *ensemble* para gerar os *rankings*.

Em HE e Ounis (2003) foi proposto um *framework* multi-visão com co-treinamento, mas com um algoritmo de aprendizado não-supervisionado nas visões dos dados. A tarefa era re-classificar uma lista de imagens obtidas por um motor de busca *Web*, a fim de mover os irrelevantes para o final da lista, e melhorar ainda mais a precisão entre as melhores imagens. Foi feita uma combinação dos aprendizes aplicados em cada visão e aplica esta combinação para re-ranquear os exemplos do conjunto ranqueados não-rotulados. Nesta proposta, o *framework* é implementado com aprendizado não-supervisionado. Nossa proposta é utilizar um aprendizado semissupervisionado. Mesmo assim, o trabalho pode inspirar eventuais mudanças na estrutura do nosso *framework* para combinar os aprendizes e os *rankings*.

Outro trabalho que também utiliza AS multivisão com *ensemble*, contendo tanto múltiplas visões quanto múltiplos aprendizados, foi proposto por Sun e Zhang (2011). Nesse *framework* um *ensemble* de aprendizes são treinados a partir de cada visão, e o consenso da predição do conjunto é usado para selecionar exemplos rotulados mais confiantes do conjunto não-rotulado para ensinar ao outro *ensemble* a partir da outra visão. Assim, combina as vantagens de aprendizado multivisão e *ensemble* para indução do classificador. Apesar dos custos computacionais aumentarem, um melhor o desempenho da classificação pode ser esperado. Este trabalho se assemelha em diversos aspectos, devido ao *framework* de aprendizado multivisão proposto ser uma adaptação do co-treinamento e utilizar um *ensemble* para combinar os exemplos mais confiantes. Porém, o treinamento utiliza uma rede neural de *ensembles* como instância dos aprendizes. Na nossa proposta utilizamos dois recomendadores clássicos da área de SR em cada visão dos dados.

O *framework* desenvolvido por Su *et al.* (2008) parte do princípio que as notas previstas com base nos dados imputados podem ser mais precisas do que aquelas baseadas na predição de dados originalmente muito esparsos. Este *framework* consta de duas etapas, a primeira utiliza 10 diferentes técnicas de imputação para enriquecer a matriz de notas esparsa, na segunda aplica um algoritmo de filtragem colaborativa nesta matriz enriquecida para gerar as recomendações finais.

Por fim, Bar *et al.* (2013) examinaram o desempenho de métodos de filtragem colaborativa ao aplicar *ensembles* homogêneos, ou seja, baseados em um único algoritmo de filtragem colaborativa. Os experimentos mostraram que este enriquecimento pode aumentar o precisão de modelos relativamente fracos ao nível dos mais avançados.

Nosso trabalho é baseado em um *ensemble* heterogêneo, pois é feita uma combinação de múltiplas previsões, estas geradas por diferentes recomendadores. Também difere dos trabalhos citados acima pelo trabalho que cada recomendador faz ao trocar informações entre eles, caracte-

Tabela 2 – Diferenças entre os trabalhos.

Trabalho	Características	Diferenças com esta pesquisa
Costa, Manzato e Campello (2018)	Predição de notas	Ranqueamentos e partida-fria
Costa, Manzato e Campello (2019)	Predição de notas	Ranqueamentos
Tan *et al.* (2004)	Notas implícitas	Notas explícitas
HE e Ounis (2003)	Enriquecimento com aprendizado não-supervisionado	Enriquecimento com aprendizado semissupervisionado
Sun e Zhang (2011)	Rede neural como visão dos dados	Item-KNN e User-KNN como visão dos dados
Su *et al.* (2008)	Enriquecimento com imputação	Enriquecimento com co-treinamento
Bar *et al.* (2013)	Uma visão no co-treinamento	Duas visões no co-treinamento

rizando o co-treinamento, buscando enriquecer o conjunto de dados, tornando-o menor esparso, e, a partir disso, obter melhores recomendações. Nossa contribuição pode ser considerada como um pré-processamento dos dados visando o enriquecimento de matrizes de usuários-itens com base em vários tipos de visão e uma confiança associada, mas que também utiliza uma técnica de *ensemble* para combinar e melhorar os resultados de visualizações individuais para obter uma matriz mais robusta.

Apresentamos aqui um resumo dos trabalhos relacionados e as diferenças do nosso trabalho na Tabela 2.

3.6 Considerações Finais

Como observado ao longo deste capítulo, foram apresentados trabalhos relacionados com a proposta deste trabalho. Como, por exemplo, o co-treinamento, mas para o cenário de predição de notas; o ranqueamento de documentos de notícias considerando como visão dos dados diferentes idiomas; o aprendizado de ranqueamento com *ensembles*, mas sem dividir os dados em diferentes visões (consequentemente, não aplicando o co-treinamento), como fizemos nesta proposta; e combinação de ranqueamento com *ensembles*, que se assemelha ao nosso trabalho, mas aplicando algoritmos clássicos de aprendizado de máquina, em vez de algoritmos de recomendação. Alguns trabalhos diferem em outros aspectos, com uso de dados implícitos e com aprendizado não-supervisionado, pois nesta proposta utilizamos dados explícitos e aprendizado semissupervisionado. Outros podem ser úteis na configuração dos experimentos e avaliações dos resultados.

Na nossa proposta, utilizamos algoritmos de recomendação nas diferentes visões dos dados do co-treinamento, combinando estes dados enriquecidos e aplicando outro algoritmo de recomendação para geração de *rankings*. Com isso, aliviamos o problema da esparsidade e partida-fria e favorece a qualidade das recomendações partindo de um conjunto de dados enriquecido.

No próximo capítulo apresentaremos a nossa proposta de solução, com a arquitetura do sistema e suas etapas, o algoritmo de recomendação utilizado e as distintas estruturas consideradas para os conjuntos de dados.

CAPÍTULO

4

PROPOSTA DE SOLUÇÃO

Este capítulo apresenta a arquitetura geral do sistema, suas etapas, o *framework* de co-treinamento, a métrica de confiança aplicada no enriquecimento e os diferentes conjuntos de dados utilizados.

4.1 Arquitetura Geral

A arquitetura central do trabalho foi baseada no *Co-Training Approach for Recommender Systems* (CoRec) (COSTA; MANZATO; CAMPELLO, 2018), uma abordagem *offline* de co-treinamento para SR. Esta abordagem prevê *ratings* de exemplos não-rotulados, mediante uma métrica de confiança, com intuito de aumentar a precisão dos resultados das recomendações.

4.1.1 CoRec

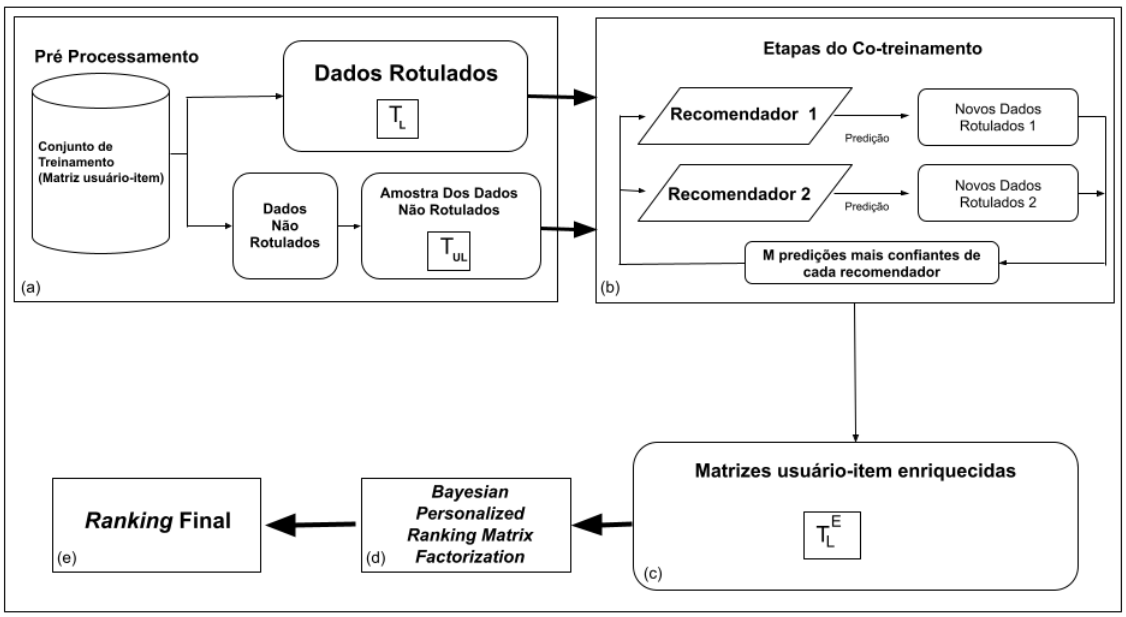
O CoRec (COSTA; MANZATO; CAMPELLO, 2018) é um *framework* que utiliza a técnica de co-treinamento para prever avaliações de exemplos não-rotulados, consequentemente diminuindo a esparsidade do conjunto de dados e aumentando a precisão dos resultados de recomendação. Originalmente, o *framework* visa o cenário de predição de notas (*ratings*) atribuídas aos itens pelos usuários, no qual os diferentes algoritmos de recomendação tentam prever as classificações dos usuários para todos os itens não vistos no conjunto de dados, como ilustrado na Figura 1. No trabalho presente, estendemos o *framework*, também utilizando o co-treinamento, para realizar o enriquecimento dos dados no cenário de ranqueamento. O *framework* apresentou a capacidade de reduzir significativamente os problemas de partida fria e esparsidade, permitindo a construção de modelos de treinamento melhores do que o uso dos recomendadores sozinhos.

A abordagem utiliza apenas um único conjunto de treinamento como dados de entrada e dois algoritmos de recomendação distintos como diferentes visões, combinando a saída de cada recomendador em um conjunto enriquecido (*ensemble*). Em cada iteração do processo de

co-treinamento, o recomendador possui uma visão que rotula os dados não rotulados de seu conjunto. As notas previstas mais confiáveis são adicionadas ao conjunto de treinamento do outro recomendador e vice-versa. Assim, as informações entre as visões são trocadas. Especificamente, cada recomendador prevê classificações para todos os pares de itens de usuário não observados (conjunto sem rótulo) de seu conjunto, as previsões mais confiáveis são usadas para preencher o conjunto de treinamento (aumentar e enriquecer o conjunto rotulado) do outro recomendador de forma iterativa. Ao término da etapa de co-treinamento, as previsões de cada recomendação são ponderadas pela sua confiança e combinadas em um conjunto enriquecido (*ensemble*).

O critério de confiança proposto se baseia no fato de que os recomendadores de filtragem colaborativa tendem a melhorar sua precisão à medida em que aumenta a quantidade de dados sobre itens ou usuários. A confiança nos SR pode ser definida como confiabilidade do sistema de suas recomendações ou previsões (RICCI; ROKACH; SHAPIRA, 2015). Quando uma medida de confiança está disponível, o usuário pode se beneficiar observando as pontuações de confiança junto com as recomendações (GUO, 2013a), além de entender que as primeiras recomendações são, em alguma medida, as mais confiantes para ele. Nesta proposta, toda previsão candidata possui um valor de confiança associado a ela.

Figura 1 – Esquema visual do Sistema



No presente trabalho, os algoritmos de recomendação utilizados no co-treinamento foram o K-Vizinhos Mais Próximos (*User-KNN*) e o K-Itens Mais Próximos (*Item-KNN*) (Figura 1b), descritos nas Seções 2.3.4 e 2.3.5, respectivamente. O primeiro produz recomendações baseadas em usuários similares, prevendo a nota de um usuário com base nas notas de usuários semelhantes (RICCI; ROKACH; SHAPIRA, 2015). O segundo também utiliza o conceito de vizinhos, mas voltado para itens. As vizinhanças são definidas ao identificar os k usuários mais semelhantes a um usuário alvo. Para determinar a vizinhança do usuário alvo, é calculada uma similaridade

com todos os outros usuários. A medida de similaridade pode ser baseada, por exemplo, no coeficiente de correlação de Pearson, semelhança de cosseno, dentre outros.

No CoRec, um recomendador é treinado com o objetivo de chegar em um acordo com as previsões do outro, de modo que ambos os recomendadores aprendam não apenas com o conjunto original de dados rotulados, mas também com as previsões mais confiantes dos dados não rotulados. A abordagem proposta demonstrou que a construção de diferentes recomendações, incorporando diferentes visões dos dados, auxilia na redução da esparsidade de notas da matriz e na partida fria, devido ao uso de dados não-rotulados e dos dados rotulados originais disponíveis para treinamento.

4.1.2 Bayesian Personalized Ranking Matrix Factorization

Após a etapa de co-treinamento, obtemos o conjunto de dados enriquecido. Nele, foi aplicado um algoritmo de recomendação (Figura 1d) para prever um *ranking* final para cada usuário (Figura 1e). Nós usamos o Bayesian Personalized Ranking Matrix Factorization (BPRMF) (RENDLE *et al.*, 2012), um modelo tradicional de ranqueamento devido à sua alta performance em tarefas de recomendação de itens *top-n* (HE *et al.*, 2017; KNIJNENBURG *et al.*, 2012).

Este método é composto de um modelo, uma função de perda e uma otimização. A função de perda, chamada de BPR-OPT, procura aproximar a métrica AUC (*Area Under the ROC Curve*), otimizando-a com uma função análoga a um gradiente descendente estocástico. A tarefa consiste em aprender os parâmetros visando otimizar, de forma aproximada, a métrica AUC. O modelo, para o BPRMF, é o BPR executado sobre um modelo de fatoração de matriz padrão.

A métrica AUC também pode ser considerada como a porcentagem de ranqueamentos corretos. Então, em um conjunto de dados contendo triplas do usuário e dois itens, onde sabemos que o usuário u preferiu o item i (visto) em relação ao item j (não visto), é calculada a porcentagem de todos os pares i e j para cada usuário que são ranqueados corretamente, e este valor é igual à métrica AUC. É gerado uma matriz de pontuação (*scores*) em cada recomendação, matriz esta obtida pelo produto da matriz de fatores latentes dos usuários pela matriz de fatores latentes dos itens, gerada para o par (u, j).

4.2 Pré-Processamento

O CoRec divide inicialmente o conjunto de dados original em rotulados e não rotulados (ou observados e não observados), T_L^η e T_{UL}^η, respectivamente, para dois algoritmos de recomendação ($\eta = 1{,}2$). Na primeira etapa, ambos os conjuntos são inicialmente os mesmos como entrada para ambos os recomendadores, constituídos de todos os pares de usuários-itens observados para todas as entradas cujas notas estão disponíveis.

A quantidade de entradas não observadas (X) são escolhidas aleatoriamente, por usuário. Em vez de inserir todas as entradas não observadas, que poderia ter um custo computacional adicional desnecessário, uma amostra de dados não rotulados por usuário é escolhida de forma aleatória (Figura 1a).

4.3 Co-Treinamento e Ensemble

O processo de co-treinamento é baseado em avaliar o mesmo conjunto de dados com dois recomendadores, direcionando-os a concordar com as predições mais confidentes de cada recomendador.

Algoritmo 1 – CoRec

Entrada: Matriz usuário-item, X e M (inteiros positivos)
Etapa 1:
$T_L \leftarrow$ entradas não-nulas da matriz usuário-item
$T_{UL} \leftarrow$ randomicamente seleciona X entradas vazias por usuários
Etapa 2: Co-treinamento
para η=1 até 2 **faça**
  $T_L^{\eta}, T_{UL}^{\eta} \leftarrow T_L, T_{UL}$
fim para
enquanto $T_{UL}^1 \neq 0$, $T_{UL}^2 \neq 0$ **faça**
  para η=1 até 2 **faça**
    Treina o Recomendador η com T_L^{η} para prever T_{UL}^{η}
    para $r_{ui} \in T_{UL}^{\eta}$ **faça**
      Obtém a confiança C_{ui}^{η} pela equação 4.2
    fim para
    $T_{\eta} \leftarrow$ Seleciona os M exemplos mais confiantes
  fim para
  $T_F \leftarrow T_1 \bigcup T_2$
  para η=1 até 2 **faça**
    $T_{UL}^{\eta} \leftarrow T_{UL}^{\eta} \setminus T_F^{\eta}$
  fim para
  $T_L^1 \leftarrow T_L^1 \bigcup T_2$
  $T_L^2 \leftarrow T_L^2 \bigcup T_1$
fim enquanto
Etapa 3: Tarefa de Recomendação
para η=1 até 2 **faça**
  Treine o recomendador η com T_L^1 e T_L^2 para prever novas notas de itens desconhecidos para cada usuário
fim para=0

Saída: Dois conjuntos com as triplas: os pares (u, i) com sua respectiva nota predita ($\hat{r}_{u,i}$).

Em cada iteração, cada recomendador η é responsável por prever as notas para seu conjunto não rotulado, T_{UL}^{η}, baseado no seu conjunto rotulado T_L^{η} (linha 10 do algoritmo 1). Então, o valor de confiança de cada nota predita é calculado, escolhendo os M mais confiantes e

removendo-os do seu conjunto não rotulado do respectivo recomendador (linha 18). Após este procedimento ser realizado para ambos os recomendadores, o algoritmo atualiza o conjunto rotulado de cada recomendador adicionando os M exemplos mais confiáveis ao conjunto rotulado do outro recomendador (linhas 20 e 21), direcionando o aprendizado do modelo com os novos exemplos rotulados da outra visão, além dos exemplos originais. Na Subseção 4.4 a formulação adotada para a métrica de confiança é detalhada. Em cada iteração, os itens recomendados são adicionados ao fluxo de treinamento do outro recomendador, e vice-versa. Na iteração seguinte, ambos os recomendadores terão um novo conjunto de dados rotulados para o treinamento. O parâmetro M é um valor inteiro positivo, responsável por controlar a taxa de aprendizado no processo de co-treinamento (Figura 1b).

Após a etapa de co-treinamento, a saída de cada recomendador é um conjunto enriquecido de cada visão, T_L^1 e T_L^2. As múltiplas predições geradas são combinadas em um conjunto de dados T_L^E (Figura 1c). Considerando a métrica de confiança C_{ui}^{η}, os resultados são agrupados pela média ponderada (COSTA; MANZATO; CAMPELLO, 2018) de todas as previsões geradas por cada recomendador em cada processo de iteração de co-treinamento:

$$\hat{r}_{ui} = \frac{\sum_{\eta} C_{ui}^{\eta} \hat{r}_{ui}^{\eta}}{\sum_{\eta} C_{ui}^{\eta}} \tag{4.1}$$

Desta forma, múltiplas predições confiantes para um mesmo par *(u, i)* são combinados em uma predição única $\hat{r}_{ui}$.

Com o término da etapa de co-treinamento, o BPRMF foi aplicado no conjunto enriquecido combinado (*ensemble*), gerando as recomendações ranqueadas.

4.4 Métrica de Confiança

A métrica de confiança em SR pode ser definida como a confiabilidade associada a uma recomendação ou predição (RICCI; ROKACH; SHAPIRA, 2015). Quando uma medida de confiança é avaliada, o usuário pode se beneficiar ao observar estes *scores* juntamente com a recomendação (GUO, 2013b). No co-treinamento, as predições mais confiantes de cada recomendador dos dados não rotulados são então usados para enriquecer iterativamente o conjunto de treino para refinamento adicional do outro recomendador.

A literatura relata algumas medidas de confiança tradicionais como *Suporte para usuário* (MAZUROWSKI, 2013), *Suporte para item* (MAZUROWSKI, 2013) e *Variabilidade para item* (ADOMAVICIUS; KAMIREDDY; KWON, 2007). *Suporte para usuário* é o número de avaliações que o usuário atribuiu anteriormente (dentro do conjunto de dados de treinamento disponível). *Suporte para item* é o número de avaliações que foram atribuídas a um item dentro do conjunto de dados de treinamento disponível. *Variabilidade para o item* é o desvio padrão de

todas as classificações que foram atribuídas a um item para todos os usuários no conjunto de dados.

Gerar um valor de confiança associado ao valor da predição da nota permite ao sistema analisar a confiança de cada predição. Por exemplo, uma base de dados com notas no intervalo de 1 a 5 estrelas. Suponha que a predição de uma nota para um item específico seja 5, mas possui um baixo valor de confiança. Ou uma nota predita baixa com um alto valor de confiança. Em sistemas de recomendação tradicionais, somente os maiores valores são considerados como os mais confiantes.

A medida de confiança que utilizamos neste trabalho foi proposta por Costa, Manzato e Campello (2018). Ela baseia-se na melhoria da precisão na filtragem colaborativa conforme a esparsidade diminui. Este critério de confiança é baseado na ideia que a filtragem colaborativa tem uma tendência de aperfeiçoar suas recomendações conforme o volume de informação sobre os usuários ou itens aumenta. Nesta proposta, cada predição possui um valor de confiança associado a ela. A métrica de confiança proposta tem maior desempenho em comparação com as três medidas tradicionais acima (COSTA; MANZATO; CAMPELLO, 2019). A medida é definida como C_{ui}^{η} para uma predição $\hat{r}_{ui}^{\eta}$ feita por um algoritmo de recomendação η para a nota do usuário u e item i:

$$C_{ui}^{\eta} = \frac{N_u N_i}{|b_{ui} - \hat{r}_{ui}^{\eta}|} \tag{4.2}$$

onde N_u e N_i são a quantidade de notas aplicados por u e o número de notas que i recebeu no conjunto rotulado do recomendador η. O b_{ui} é a estimativa do *baseline* da nota para o par *(u, i)*, apresentado na Equação 2.3.

A justificativa para o uso desta medida está relacionada com o seguinte fato: quanto mais usuários e itens são populares, as previsões calculadas com base nesses usuários e itens tendem a ser mais precisas. Outro aspecto é que uma predição muito discrepante do baseline b_{ui} implica no módulo desta diferença ser grande (denominador da Equação 4.2) e, consequentemente, o valor da confiança tende a ser prejudicado.

4.5 Recomendação De Itens Ranqueados

Neste trabalho foram considerados três conjuntos de dados distintos no treinamento com intuito de mensurar a qualidade da recomendação final (Figura 1e), detalhado abaixo. Para o primeiro caso, o objetivo foi examinar a qualidade das recomendações partindo de um conjunto de dados enriquecido. Já no segundo caso, o intuito foi utilizar o conjunto enriquecido (*ensemble*) apenas para o treino do BPRMF e gerar os *rankings* considerando os arquivos de treino, com isso possibilita a recomendação de itens que poderia estar nos itens preditos. O terceiro caso é considerar apenas os itens preditos na etapa de enriquecimento como um ranqueamento. Este

caso não é utilizado o BPRMF, a recomendação é confrontada com o conjunto de teste.

O conjunto enriquecido (*ensemble*), obtido pela etapa de co-treinamento, é a união do conjunto de treino com o conjunto dos itens preditos (Figura 2).

Figura 2 – Diagrama de Venn dos Conjuntos de Dados.

4.5.1 Conjunto Enriquecido Como Conjunto de Treino - Caso 1

Após a etapa de co-treinamento, os conjuntos enriquecidos de cada recomendador são combinados em um conjunto único enriquecido (Figura 1c). Com este conjunto enriquecido em mãos, foram feitos dois testes, aplicar o BPRMF para gerar os rankings finais, ilustrado no esquema da Figura 3, e para o cenário de partida-fria (Figura 4).

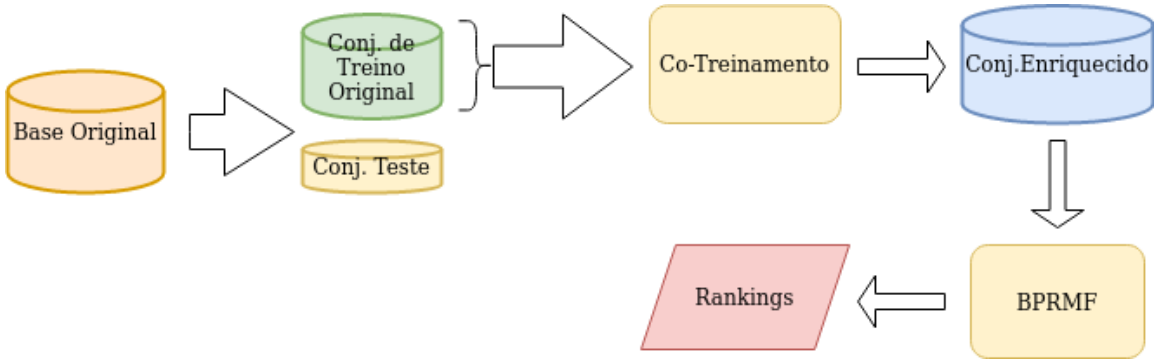


Figura 3 – Esquema Visual - Caso 1.

Fonte: Elaborada pelo autor.

Para o cenário de partida-fria, o conjunto de dados enriquecido foi particionado considerando um intervalo para a quantidade de interações (notas atribuídas) pelos usuários.

Em mais detalhes, o procedimento desta filtragem das partições para este cenário, foi:

1. Dividir a base de dados em treino e teste;

2. Aplicar o CoRec no arquivo de treino e o *output* é o conjunto enriquecido combinado (*ensemble*);
3. Gerar o *ranking*, por meio do BPRMF, com o ensemble como *input*;
4. Filtrar os usuários do conjunto de treino pela quantidade de interações e criar três novas partições;
5. Os **usuários** que estão em cada partição são mantidos no *ranking* gerado pelo BPRMF, gerando 3 partições com *rankings*. Da mesma forma são feitas filtragens no conjunto de teste, gerando 3 partições para os testes de cada;
6. Com estes *rankings* filtrados pra cada partição, é feita a avaliação com o respectivo arquivo de teste.

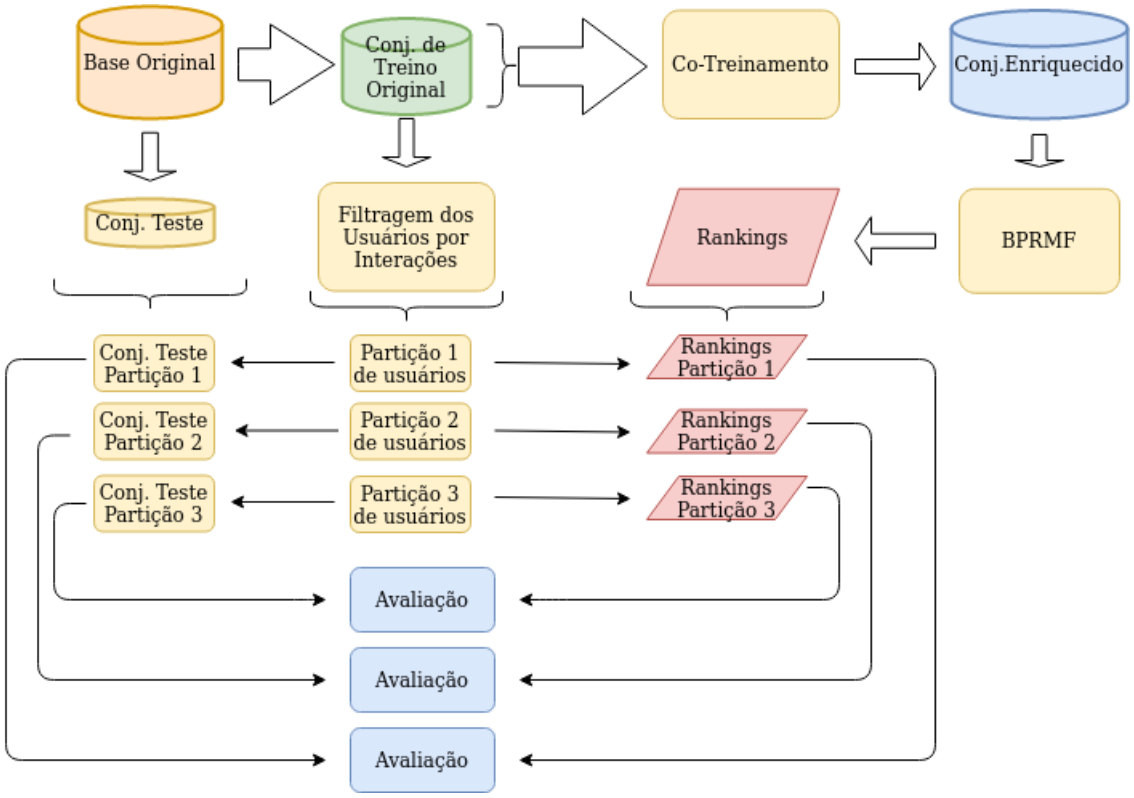


Figura 4 – Esquema Visual - Caso 1 para o cenário de partida-fria.

Fonte: Elaborada pelo autor.

Desta forma, com a filtragem por meio das partições, pode-se verificar a eficácia deste método também para o cenário de partida-fria.

4.5.2 Conjunto Enriquecido Utilizado Como Treinamento do Ranqueador - Caso 2

Outra análise que fizemos utilizou o conjunto enriquecido apenas no treinamento do BPRMF, mas considerando o arquivo de treino para a recomendação final dos *rankings*. Desta

forma, possibilitamos o eventual ranqueamento de itens que foram previstos pelo CoRec durante o enriquecimento.

Ao aplicar o BPRMF diretamente no conjunto enriquecido, sem considerar o conjunto de treino na recomendação, como é feito no Caso 1, os itens preditos são considerados como vistos pelo ranqueador e, por isso, não poderiam ser recomendados.

Fizemos esta análise para verificar a performance do recomendador neste cenário, mantendo a proposta do ranqueador ser treinado com um conjunto menos esparso. O esquema para este caso está ilustado na Figura 5.

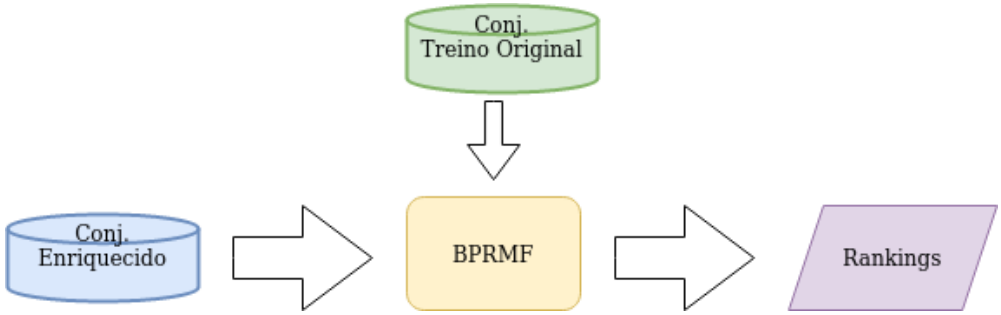


Figura 5 – Esquema Visual - Caso 2.

Fonte: Elaborada pelo autor.

4.5.3 Conjunto de Itens Mais Confiáveis como Ranking de Recomendação - Caso 3

Foram feitos testes também para analisar o desempenho específico somente dos itens preditos, ou seja, apenas para as triplas que o modelo de co-treinamento previu. Estas triplas foram avaliadas diretamente, sendo confrontadas com o conjunto de teste, sem aplicar o BPRMF nelas. Havia uma expectativa neste caso por estas triplas serem as predições mais confiantes, dado pela métrica discutida na Subseção 4.4. O esquema para este caso está ilustado na Figura 6.

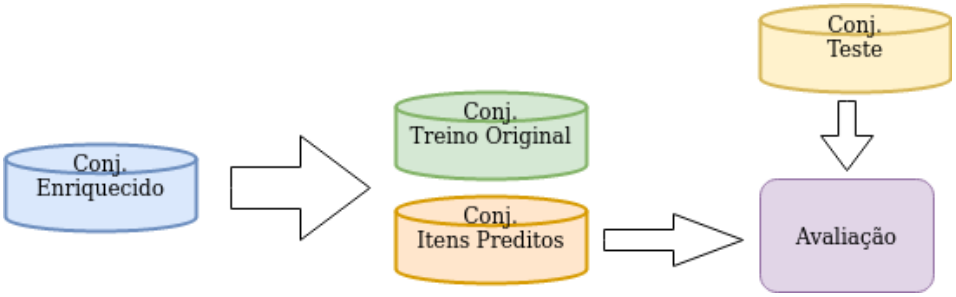


Figura 6 – Esquema Visual - Caso 3.

Fonte: Elaborada pelo autor.

4.6 Considerações Finais

Neste capítulo foi apresentada a arquitetura proposta para solução. O trabalho baseia-se na recomendação por meio da diminuição da esparsidade de um conjunto de dados com a técnica de co-treinamento. Uma métrica de confiança foi utilizada nestas previsões para enriquecer apenas com os mais confiantes. Tanto o *framework* de co-treinamento, utilizado no enriquecimento, quanto a métrica de confiança utilizada foram contribuições do trabalho de Costa, Manzato e Campello (2018). A contribuição deste trabalho foi analisar as diferentes recomendações geradas partindo dos conjuntos enriquecidos.

A recomendação em forma de *ranking* foi feita utilizando o conjunto enriquecido, com o BPRMF, algoritmo de ranqueamento personalizado. Foram considerados também três conjuntos distintos para avaliar a qualidade da recomendação final. Para o cenário de partida-fria foi proposto um ajuste nos dados, detalhado na Seção 5.2.4.

Ao apresentar a proposta de solução, alguns aspectos como a configuração e detalhes da experimentação foram poupados para o capítulo seguinte. Nele serão descritos as ferramentas e bases de dados utilizados, os critérios para validação dos parâmetros e métricas de avaliação. Apresentam-se também os resultados obtidos em cada experimento e um comparativo dos melhores resultados.

CAPÍTULO

5

EXPERIMENTOS E RESULTADOS

Neste capítulo serão descritas as ferramentas e bases de dados utilizadas para o desenvolvimento da pesquisa, os procedimentos adotados para validar e avaliar a proposta, além dos resultados e discussão dos mesmos.

5.1 Metodologia de Avaliação

Para avaliar a proposta, dividimos os conjunto de dados em pastas, para validação cruzada, e, em seguida, a divisão em um conjunto de treinamento e de teste. Utilizamos algoritmos de vizinhança no co-treinamento para obter os conjuntos enriquecidos, aplicamos o algoritmo de recomendação nestes conjuntos enriquecidos e confrontamos com o conjunto de teste. Maiores detalhes destas etapas estão descritos nas Subseções seguintes.

5.1.1 Ferramentas Utilizadas

Para o desenvolvimento do projeto foram utilizadas as bibliotecas e *frameworks* de código aberto: *Case Recommender*, Pandas, Numpy, Scipy, Scikit-Learn.

O *Case Recommender* (COSTA; MANZATO, 2016) é um *framework* implementado em Python e possui vários recomendadores das abordagens Baseada em Conteúdo e da Filtragem Colaborativa, que utilizam coleta de dados explícita e implícita. O *Case Recommender* fornece um conjunto de componentes a partir do qual os desenvolvedores podem construir um sistema personalizado. O *framework* aborda o cenário de predição de notas e recomendações de itens, como o CoRec, utilizado neste trabalho para enriquecer os *datasets*, disponilizando várias rotinas de avaliações conhecidas, como RMSE, MAE, MAP, NDCG e Precisão.

O Pandas (TEAM, 2020) é uma biblioteca Python com estruturas para lidar com conjuntos de dados estruturados comuns em estatística, finanças e diversos outros campos. A biblioteca fornece rotinas integradas para a manipulação e análise de dados. Outra biblioteca utilizada foi o

NumPy (HARRIS *et al.*, 2020) é um projeto de código 100% aberto com o objetivo de permitir a computação numérica com Python. Foi criado em 2005, é gratuito, lançado sob os termos liberais da licença BSD modificada e desenvolvido abertamente no GitHub.

O SciPy (VIRTANEN *et al.*, 2020) (*Scientific computing tools for Python*) se refere a diversas entidades distintas, mas relacionadas, como o ecossistema SciPy, uma coleção de softwares de código aberto para computação científica, e o Matplotlib, pacote de plotagem popular que fornece plotagem 2 e 3-D. Por fim, o Scikit-Learn (PEDREGOSA *et al.*, 2011) é uma biblioteca que fornece implementações de diversos algoritmos de aprendizado de máquina na linguagem Python.

5.1.2 Bases de Dados

Os conjuntos de dados utilizados para avaliar a proposta são de acesso público e de diferentes domínios, com notas explícitas atribuídas pelos usuários.

- *Filmtrust*: Este conjunto de dados foi obtido monitorando o site FilmTrust em junho de 2011, desenvolvido por Guo, Zhang e Yorke-Smith (2013), e contém 1.508 usuários, 2.071 filmes e 35.497 notas, com valores de 0.5 a 4.

- *Jester*: Esta base de dados voltado para recomendar piadas foi desenvolvido por Goldberg *et al.* (2001) e contém 4,1 milhões de notas contínuas (-10,00 a +10,00) de 100 piadas obtidas de 73.496 usuários. Para otimizar os experimentos, foi utilizado apenas um subconjunto da base original, resultando em 947.736 *feedbacks* (notas) de 10.291 usuários para 140 itens.

- *Movielens*: Conhecida base de dados desenvolvida pelo grupo *GroupLens Research* que coletou e disponibilizou as avaliações do seu site de filmes, o *dataset* utilizado (CANTADOR; BRUSILOVSKY; KUFLIK, 2011) é uma extensão do Movielens10M no qual apenas os usuários com informações de notas e marcações foram mantidos. Possui 855.598 notas (de 0.5 a 5) de 10.197 filmes obtidas de 2.113 usuários.

5.1.3 Método de Validação e Avaliação

Cada conjunto de dados foi dividido em 10 pastas para validação cruzada (LANGFORD, 2005), com 80% Treino, 10% teste e 10% validação. Avaliamos os resultados de cada *fold* e registramos sua média. Um conjunto de validação, que não continha o conjunto de teste, foi usado para obter os melhores parâmetros do BPRMF para cada *dataset*. Utilizamos neste trabalho dois algoritmos de recomendação como diferentes visões na etapa de co-treinamento, buscando a similaridade entre itens e usuários, respectivamente. A qualidade da recomendação foi mensurada pelo Mean Average Precision (MAP@N) e Normalized Discounted Cumulative Gain (NDCG@N), descritos nas Subseções 2.3.6.3 e 2.3.6.4, respectivamente, com *rankings* de tamanho N = {5, 10, 50 e 100}. Nesta Seção serão apresentados os resultados obtidos pelo

NDCG. Para averiguar a relevância estatística entre os resultados foi aplicado o teste estatístico t-student com 95% de confiança.

No co-treinamento foram utilizados dois algoritmos tradicionais de recomendação, Item-KNN e User-KNN, aplicados nas visões dos dados, com o cosseno como medida de proximidade. A utilização destes algoritmos justifica-se pelas recomendações baseadas em pontos de vista diferentes, enquanto o Item-KNN baseia-se nas inter-relações entre itens, o User-KNN se baseia nas inter-relações entre os usuários para as previsões. A quantidade de vizinhos definidos está relacionada com a quantidade de usuários da base. Foi adotado para todos o valor inteiro da raiz quadrada da quantidade total de usuários (HASSANAT *et al.*, 2014).

O enriquecimento foi realizado de forma moderada, ou seja, com poucas predições por usuário, com intuito de não diminuir demasiadamente a esparsidade, havendo espaço para gerar recomendações. Este controle foi feito com o parâmetro X, que representa a quantidade de entradas não observadas por usuário.

Os valores das esparsidades dos conjuntos originais e com enriquecimento estão na Tabela 3.

Tabela 3 – Esparsidade dos conjuntos originais e enriquecidos em porcentagem.

		X						
Conjunto de Dados	Original	1	5	10	20	30	40	50
Filmtrust	98,93	98,92	98,88	98,83	98,73	98,63	98,53	98,43
Jester	40,80	40,66	40,10	39,41	38,02	36,83	35,85	34,94
Movielens	96,372	96,37	96,36	96,35	96,33	96,31	96,29	96,27

Foram experimentados diversos valores de X (número de itens não rotulados por usuário no co-treinamento) para verificar se havia alguma relação dos enriquecimentos e a qualidade da recomendação. Os valores adotados para X foram 1, 5, 10, 20, 30, 40 e 50. O parâmetro M, que seleciona as previsões mais confiantes, foi definido como 10% do tamanho do conjunto de dados não rotulados.

O conjunto de treino, ou seja, sem enriquecimento, foi utilizado como conjunto de dados para a avaliação do *baseline*. O BPRMF foi aplicado neste conjunto sem enriquecimento e as recomendações foram confrontadas com o conjunto de teste, sendo este nosso *baseline*. No método proposto o BPRMF foi aplicado no conjunto enriquecido. Podemos nos beneficiar do uso de um conjunto de dados menos esparso porque conforme o recomendador possui mais exemplos rotulados, mais conhecimento terá sobre o comportamento dos usuários.

5.2 Validação da Proposta

O trabalho foi dividido em duas etapas, como mencionado anteriormente. A primeira utilizando o conjunto enriquecido como dados de treino do ranqueador final, a segunda voltada

para o cenário de *coldstart*. Neste capítulo são apresentados os resultados mais relevantes. As tabelas completas, com todos os resultados, estão disponíveis no Apêndice A.

O *baseline* considerado para as avaliações nos 3 casos abordados nas Subseções 4.5.1, 4.5.2 e 4.5.3, foi a aplicação do BPRMF na base original de treino (sem enriquecimento) e avaliado com o arquivo de teste.

Com o conjunto enriquecido pela etapa de co-treinamento em mãos, o BPRMF gerou 100 recomendações de itens por usuário, ordenados decrescentemente pela sua pontuação (*score*).

5.2.1 Avaliação dos Rankings - Caso 1

Avaliamos as recomendações obtidas no Caso 1, discutido na Seção 4.5.1, por meio da medição da acurácia dos *rankings* gerados, como apresentado na Figura 3. Neste Caso 1 analisamos a qualidade dos *rankings* que foram gerados com o BPRMF aplicado no conjunto enriquecido (união do conjunto de treino com o conjunto dos itens preditos). Uma particularidade deste caso é que os itens preditos pelo enriquecimento são considerados já vistos pelo recomendador, não podendo ser recomendados.

A base *Filmtrust* apresentou os melhores resultados para os valores iniciais de X: 1, 5 e 10 (Tabela 4), com melhora de $0,73\%$, $0,60\%$ e $0,66\%$ na média, respectivamente, considerando ambas as métricas (MAP e NDCG). Mesmo possuindo uma alta esparsidade, o pouco enriquecimento propiciou melhora na acurácia da recomendação. Os valores em negrito indicam ganho do *baseline*. Valores sublinhados indicam o maior valor da categoria (quando há ganho do *baseline*), ** indica significância estatística com *p-value* < 0,01 e * para *p-value* < 0,05.

Filmtrust		NDCG			
	Esparsidade (%)	@5	@10	@50	@100
Baseline	98.94	0.6006	0.6149	0.6175	0.6182
X = 1	98.92	**0.6032**	**0.6180***	**0.6200***	**0.6207***
X = 5	98.89	**0.6033**	**0.6181**	**0.6197**	**0.6202**
X = 10	98.84	**0.6038***	**0.6177**	**0.6200****	**0.6209****

Tabela 4 – Resultados do BPRMF para o Caso 1 - Base *Filmtrust*.

Para a base *Jester*, o valor de X que obteve melhores resultados, comparados com o baseline, foi de 1. Neste caso, a base não possui alta esparsidade, possui em torno de 40%. O pouco enriquecimento proporcionou o BPRMF realizar recomendações melhores que o *baseline*, porém, com menor percentual de ganho comparado com o *Filmtrust* (Tabela 5). Obteve melhora de $0,03\%$ na média, comparado com o *baseline*.

Na base Movielens, o resultado com maior destaque se deu para $X = 5$ (Tabela 6). Da mesma forma que o Filmtrust, é uma base com alta esparsidade e obteve ganho percentual de $0,34\%$ na média, comparado com o *baseline*.

Jester		NDCG			
	Esparsidade (%)	@5	@10	@50	@100
Baseline	40.80	0.9137	0.8930	0.8620	0.8609
X = 1	40.66	0.9136	**0.8936**	**0.862**	**0.8610**
X = 5	40.10	0.9121	0.8916	0.8600	0.8587
X = 10	39.41	0.9115	0.8906	0.8583	0.856775

Tabela 5 – Resultados do BPRMF para o Caso 1 - Base *Jester*.

O conjunto de dados enriquecido possui mais informação que o *baseline*, o que permite um melhor aprendizado do recomendador e, consequentemente, gera melhores recomendações.

Movielens		NDCG			
	Esparsidade (%)	@5	@10	@50	@100
Baseline	96.37	0.6629	0.6495	0.6143	0.6005
X = 1	96.37	**0.6655**	**0.6512**	**0.6148**	**0.6013**
X = 5	96.36	**0.6662**	**0.6525**	**0.6152**	**0.6017**
X = 10	96.35	0.6621	**0.6499**	0.6142	**0.6005**

Tabela 6 – Resultados do BPRMF o Caso 1 - Base *Movielens*.

5.2.2 Avaliação dos Rankings - Caso 2

Os melhores resultados para o Caso 2, discutido na Seção 4.5.2, serão apresentados nesta Seção. Neste Caso 2 o *ranking* é gerado de forma semelhante ao Caso 1, aplicando o BPRMF no conjunto enriquecido (união do conjunto de treino com o conjunto predito), porém, para gerar o *ranking*, o BPRMF considera o conjunto de treino. O que permite a possibilidade de recomendar itens do conjunto predito, que não é possível acontecer no Caso 1, pois considera o conjunto predito como visto e não entraria no *ranking*.

A base *Filmtrust* apresentou o melhor resultado para o valor de $X = 1$, com melhora de $0,45\%$ (Tabela 7). O valor de X com melhor desempenho também foi o mesmo que o Caso 1.

Filmtrust		NDCG			
	Esparsidade (%)	@5	@10	@50	@100
Baseline	98.94	0.6007	0.6149	0.6175	0.6183
X = 1	98.92	**0.6022**	**0.6163**	**0.6187**	**0.6196**
X = 5	98.89	**0.6017**	**0.6167****	**0.6188***	**0.6196***
X = 10	98.84	**0.6011**	**0.6162**	**0.6185**	**0.6192**

Tabela 7 – Resultados do BPRMF para o Caso 2 - Base *Filmtrust*.

Para a base *Jester*, o destaque foi para $X = 1$ (Tabela 8). A proposta obteve melhora de $0,02\%$ na média, comparado com o *baseline*, quando obtém valores maiores (NDCG@5 e @10).

Da mesma forma que no Caso 1 para esta base, o baixo ganho pode estar relacionado com o fato da base não ser muito esparsa. As outras bases, com esparsidades acima de 95%, obtiveram resultados melhores.

Jester		NDCG			
	Esparsidade (%)	@5	@10	@50	@100
Baseline	40.80	0.9137	0.8931	0.8620	0.8609
X = 1	40.66	**0.9138**	**0.8931**	0.8618	0.8606
X = 5	40.10	0.9118	0.8917	0.8606	0.8592
X = 10	39.41	0.9108	0.8903	0.8587	0.8570

Tabela 8 – Resultados do BPRMF para o Caso 2 - Base *Jester*.

Por fim, a base *Movielens*, o resultado com maior destaque se deu para $X = 5$ (Tabela 9), que obteve ganho percentual de $0,24\%$ na média, comparado com o *baseline*.

Possibilitar a recomendação de itens que já foram preditos, nesta configuração, não houve muita diferença comparado com os resultados do Caso 1. Acreditamos que a recomendação esteja mais relacionada com o conjunto que é utilizado para o treinamento do algoritmo do que esta permissão da possibilidade de recomendar itens preditos, ao forçar o BPRMF olhar o conjunto de treino no *ranking* final.

Movielens		NDCG			
	Esparsidade (%)	@5	@10	@50	@100
Baseline	96.37	0.6629	0.6495	0.6143	0.6005
X = 1	96.37	**0.6635**	**0.6504**	**0.6144**	**0.6010**
X = 5	96.36	**0.6650**	**0.6516**	**0.6155**	**0.6019**
X = 10	96.35	0.6623	0.6486	0.6132	0.5994

Tabela 9 – Resultados do BPRMF para o Caso 2 - Base *Movielens*.

5.2.3 Avaliação dos Rankings - Caso 3

Para este Caso 3, o intuito foi avaliar os itens escolhidos pela métrica de confiança na etapa de co-treinamento. Estes itens preditos foram considerados como a própria recomendação, sem aplicar o BPRMF, e avaliados com o conjunto de teste.

Os resultados da proposta para o Caso 3, apresentado na Seção 4.5.3, não apresentou nenhuma melhora nas três bases de dados trabalhadas. Os melhores resultados, em todas as bases, foi para o valor de X sendo 50. A diferença foi discrepante, com o valor do *baseline* sendo 24 vezes maior para o *Filmtrust*, considerando o melhor dos desempenhos (Tabela 10), $68,8\%$ maior para o *Jester* (Tabela 11) e quase 17 vezes maior para o *Movielens* (Tabela 12). Para o cálculo destas diferenças, foi considerada a média da diferença percentual entre cada métrica.

Filmtrust	NDCG			
	@5	@10	@50	@100
Baseline	0.6007	0.6149	0.6175	0.6183
X = 50	0.0215	0.0254	0.0275	0.0275

Tabela 10 – Resultados para o Caso 3 - Base *Filmtrust*.

Jester	NDCG			
	@5	@10	@50	@100
Baseline	0.9137	0.8931	0.8620	0.8609
X = 50	0.4903	0.5218	0.5422	0.5422

Tabela 11 – Resultados para o Caso 3 - Base *Jester*.

Movielens	NDCG			
	@5	@10	@50	@100
Baseline	0.6629	0.6495	0.6143	0.6005
X = 50	0.0233	0.0368	0.0486	0.0486

Tabela 12 – Resultados para o Caso 3 - Base *Movielens*.

Os desempenhos da proposta nas 3 bases mostraram que, apesar da métrica de confiança auxiliar na escolha dos itens para a redução da esparsidade, estes itens escolhidos não são boas recomendações. Uma possível justificativa está na própria formulação da métrica de confiança, a qual não é direcionada para o cenário de ranqueamento, mas está associada com a acurácia nos valores da predição, para distinguir as previsões individuais que podem ser confiáveis daquelas que não podem.

5.2.4 Cenário de Coldstart

Para o cenário de *coldstart*, o *baseline* adotado segue o mesmo procedimento na filtragem das partições, mas feito para o *ranking* gerado pelo BPRMF com o arquivo de treino.

Neste cenário, o conjunto de dados enriquecido foi particionado considerando a quantidade de interações (notas atribuídas) pelos usuários, como apresentado na Figura 4. Foram consideradas 3 partições distintas, com os seguintes intervalos de interações: 1 a 20, 21 a 40 e 41 a 60 para o Filmtrust e Movielens. Ou seja, os usuários que tiveram entre 1 e 20 interações, no conjunto de treino, foram alocados em uma partição distinta para treinamento do BPRMF, específico para análise do cenário de *coldstart*. Da mesma forma com os outros dois intervalos, gerando outras 2 partições. Para o Jester, por ser uma base menos esparsa, foi considerado um intervalo maior de interações: 30 a 50, 51 a 70, 71 a 90. Nas tabelas, o *Caso 1* representa a aplicação do BPRMF para o caso que o conjunto de treino usado é o enriquecido, descrito na Seção 4.5.1. O *Caso 2* representa a aplicação do BPRMF para o caso que o conjunto enriquecido é usado apenas como treinamento do ranqueador, descrito na Seção 4.5.2.

Em seguida serão apresentados os resultados de destaque para a base *Filmtrust*. A base *Filmtrust* apresentou os melhores resultados para X sendo 20 e 40 (Tabela 13). Mesmo possuindo uma alta esparsidade, o pouco enriquecimento propiciou melhora na acurácia da recomendação. Houve uma melhora de 1,90% e 1,74% na média, respectivamente, para o Caso 1 e 2, quando X é igual a 20, na partição com 1 até 20 interações. Para a partição seguinte, com 21 até 40 interações, o Caso 2 obteve 0,56% de melhora na média relação ao *baseline*. Para X igual a 40, o Caso 1 melhora em 1,04% e o Caso 2 melhora em 1,08% em relação ao *baseline*, na média.

Filmtrust			NDCG			
	X	Interações	@5	@10	@50	@100
Baseline	20	$1 \sim 20$	0.4315	0.4655	0.4780	0.4796
Caso 1	20	$1 \sim 20$	**0.4353**	**0.4706**	**0.4822**	**0.4842**
Caso 2	20	$1 \sim 20$	**0.4360****	**0.4712****	**0.4825****	**0.4840***
Baseline	20	$21 \sim 40$	**0.6130**	0.6030	0.5893	0.5898
Caso 1	20	$21 \sim 40$	0.6095	**0.6046**	**0.5897**	**0.5899**
Caso 2	20	$21 \sim 40$	**0.6127**	**0.6066***	**0.5901**	**0.5912**
Baseline	40	$1 \sim 20$	0.4339	0.4688	0.4814	0.4830
Caso 1	40	$1 \sim 20$	**0.4393***	**0.4717**	**0.4835**	**0.4854***
Caso 2	40	$1 \sim 20$	**0.4392***	**0.4727***	**0.4843**	**0.4860***

Tabela 13 – Resultados do BPRMF para o cenário de partida-fria - Base *Filmtrust*.

Os resultados de destaque para a base *Jester* estão nas Tabelas 14, 15 e 16, respectivamente para X sendo 5, 10 e 20. Para $X = 5$, nas 3 partições de interações, o Caso 1 obteve melhor acurácia, na média, 4,18% a mais para a partição entre 30 e 50 interações, 4,56% para 51 a 70 interações e para a partição entre 71 a 90 interações 1,66% na média acima do *baseline*. O caso de $X = 5$ foi a que obteve melhor desempenho para as 2 partições iniciais, reforçando a qualidade da recomendação para a partida-fria.

Também para $X = 10$ o Caso 1 obteve melhor desempenho nas 3 partições. Obteve 3,95% melhor desempenho, na média, comparado ao baseline para a primeira partição, 3,78% e 0,51% para as partições seguintes, respectivamente.

No caso do $X = 20$, o Caso 1 obteve ganhos apenas nas duas primeiras partições, com 1,8% acima do baseline na média para a partição entre 30 e 50 interações e 2,13% para a partição entre 51 e 70 interações.

Por fim, os melhores resultados para a base Movielens, que ficaram concentrados na partição na primeira partição, a que possui entre 1 e 20 interações (Tabela 17).

Para $X = 1$, o Caso 1 ficou 5,19% acima do *baseline* na média, o Caso 2 com 8,59%. Para $X = 5$, o ganho do Caso 1 foi de 8,06% acima do *baseline* na média e 14,21% do Caso 2. Para $X = 10$, o Caso 1 obteve 14,70% acima do baseline e 13,09% para $X = 30$.

Jester			NDCG			
	X	Interações	@5	@10	@50	@100
Baseline	5	30 ∼ 50	0.8404	0.8058	0.7494	0.7455
Caso 1	5	30 ∼ 50	**0.8673****	**0.8276****	**0.7761****	**0.7733****
Caso 2	5	30 ∼ 50	**0.8417**	0.8066	0.7482	0.7435
Baseline	5	51 ∼ 70	0.8484	0.8159	0.7630	0.7610
Caso 1	5	51 ∼ 70	**0.8739****	**0.8408****	**0.7929****	**0.7912****
Caso 2	5	51 ∼ 70	0.8462	0.8149	0.7613	0.7586
Baseline	5	71 ∼ 90	0.9042	0.8767	0.8382	0.8381
Caso 1	5	71 ∼ 90	**0.9158****	**0.8869****	**0.8500****	**0.8499****
Caso 2	5	71 ∼ 90	0.9032	0.8749	0.8368	0.8366

Tabela 14 – Resultados do BPRMF para o cenário de partida-fria com $X = 5$ - Base *Jester*.

Jester			NDCG			
	X	Interações	@5	@10	@50	@100
Baseline	10	30 ∼ 50	0.8361	0.8012	0.7466	0.7430
Caso 1	10	30 ∼ 50	**0.8621****	**0.8245****	**0.7707****	**0.7673****
Caso 2	10	30 ∼ 50	**0.8429***	**0.8050**	**0.7471**	0.7409
Baseline	10	51 ∼ 70	0.8454	0.8141	0.7622	0.7601
Caso 1	10	51 ∼ 70	**0.8697****	**0.8352****	**0.7861****	**0.7843****
Caso 2	10	51 ∼ 70	0.8454	0.8130	0.7589	0.7554
Baseline	10	71 ∼ 90	0.9064	0.8782	0.8398	0.8397
Caso 1	10	71 ∼ 90	**0.9098***	**0.8806***	**0.8435****	**0.8434****
Caso 2	10	71 ∼ 90	0.8990	0.8717	0.8331	0.8329

Tabela 15 – Resultados do BPRMF para o cenário de partida-fria com $X = 10$ - Base *Jester*.

Jester			NDCG			
	X	Interações	@5	@10	@50	@100
Baseline	20	30 ∼ 50	0.8376	0.8035	0.7469	0.7428
Caso 1	20	30 ∼ 50	**0.8508****	**0.8126****	**0.7576****	**0.7538****
Caso 2	20	30 ∼ 50	**0.8440***	**0.8088***	**0.7480**	0.7400
Baseline	20	51 ∼ 70	0.8457	0.8142	0.7620	0.7599
Caso 1	20	51 ∼ 70	**0.8605****	**0.8257****	**0.7752****	**0.7733****
Caso 2	20	51 ∼ 70	0.8431	0.8109	0.7556	0.7509

Tabela 16 – Resultados do BPRMF para o cenário de partida-fria com $X = 20$ - Base *Jester*.

5.2.5 Discussão dos Resultados

Os experimentos apresentaram a possibilidade de obter melhores resultados usando a abordagem de enriquecer o conjunto de dados comparado ao usar o mesmo algoritmo de recomendação no conjunto de treino (*baseline*). Ao diminuir a esparsidade da matriz, novos itens são rotulados, resultando em melhores recomendações ranqueadas com o BPRMF. A quantidade

Movielens			NDCG			
	X	Interações	@5	@10	@50	@100
Baseline	1	1 ∼20	0.1373	0.1626	0.2027	0.2145
Caso 1	1	1 ∼20	**0.1482**	**0.1735**	**0.2064**	**0.2155**
Caso 2	1	1 ∼20	**0.1505**	**0.1763**	**0.2096**	**0.2255**
Baseline	5	1 ∼20	0.1373	0.1626	0.2027	0.2145
Caso 1	5	1 ∼20	**0.1496**	**0.1796**	**0.2094**	**0.2170**
Caso 2	5	1 ∼20	**0.1671**	**0.1866**	**0.2142**	**0.2257**
Baseline	10	1 ∼20	0.1373	0.1626	0.2027	0.2145
Caso 1	10	1 ∼20	**0.1634**	**0.1899**	**0.2159**	**0.2196**
Caso 2	10	1 ∼20	0.1350	0.1593	0.2001	0.2062
Baseline	30	1 ∼20	0.1373	0.1626	0.2027	0.2145
Caso 1	30	1 ∼20	**0.1600**	**0.1789**	**0.2136**	**0.2192**
Caso 2	30	1 ∼20	**0.1396**	0.1592	0.1945	0.2069

Tabela 17 – Resultados do BPRMF para o cenário de partida-fria com - Base *Movielens*.

do enriquecimento se mostrou crucial para a obtenção de boas recomendações, pois enriquecer demasiadamente pode começar a interferir na recomendação de um item, disponibilizando poucos itens não vistos para o recomendador.

Pelos resultados apresentados tanto pelo Caso 1 quanto pelo Caso 2 é possível notar que há um limite de performance do algoritmo BPRMF. Dependendo do valor de X, responsável pelo quanto é enriquecido o conjunto de dados, é possível obter resultados melhores comparados com o *baseline*.

A redução da esparsidade mostra uma melhoria dos resultados, uma vez que o sistema possui mais exemplos rotulados, aumenta a habilidade do recomendador em aprender o comportamento dos usuários. Até mesmo uma pequena redução na porcentagem, como acontece com o *Filmtrust* e *Movielens*, apresentam melhores resultados comparados com o conjunto de dados de treino original. As porcentagens das esparsidades, seja do conjunto original, quanto dos conjuntos enriquecidos, estão apresentados na Tabela 3. Em especial para o *Filmtrust*, todos os casos foram melhores que o *baseline*, os melhores desempenhos se deram para os valores iniciais de X (1, 5 e 10 itens rotulados por usuário) para o Caso 1, e para $X = 40$ para o Caso 2. Para o *Movielens*, os melhores resultados foram também para os valores iniciais de X (1 e 5) tanto para o Caso 1 como para o Caso 2.

Em conjuntos de dados mais esparsos, como o *Filmtrust* e *Movielens*, houveram maiores ganhos na média com um pequeno enriquecimento das bases. Os gráficos das Figuras 7 e 9 ilustram o valor da média dos resultados das métricas em relação ao valor das esparsidades, todos para o Caso 1. Nota-se que com o Jester (Figura 8) obteve-se pouco ganho com o enriquecimento.

O conjunto de dados *Jester* não é altamente esparso, o conjunto original possui 40,80% de esparsidade (Tabela 3), o que pode não ter gerado boas recomendações ao ser enriquecido. Apesar disso, é possível aplicar este sistema para o cenário de *coldstart* e obter bons resultados,

Figura 7 – Gráfico - Média Avaliações versus Esparsidade (%) - Filmtrust.

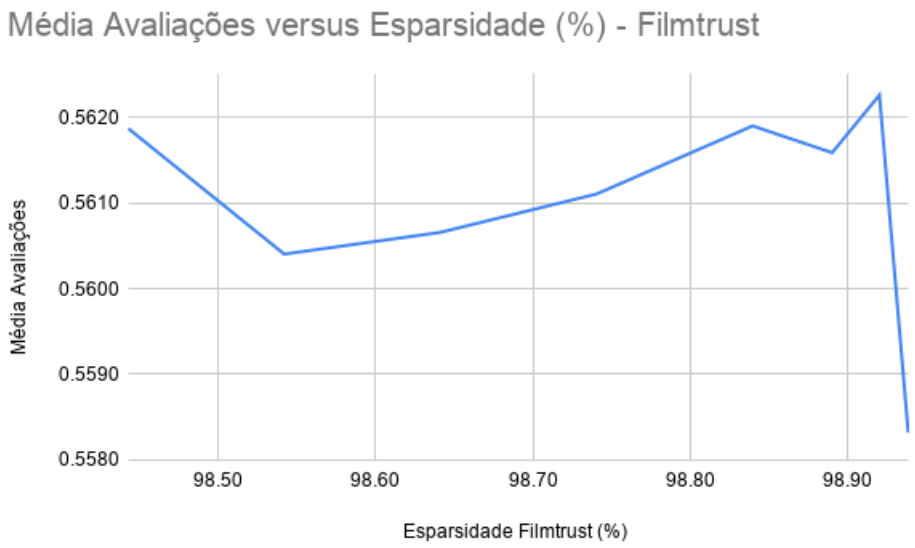
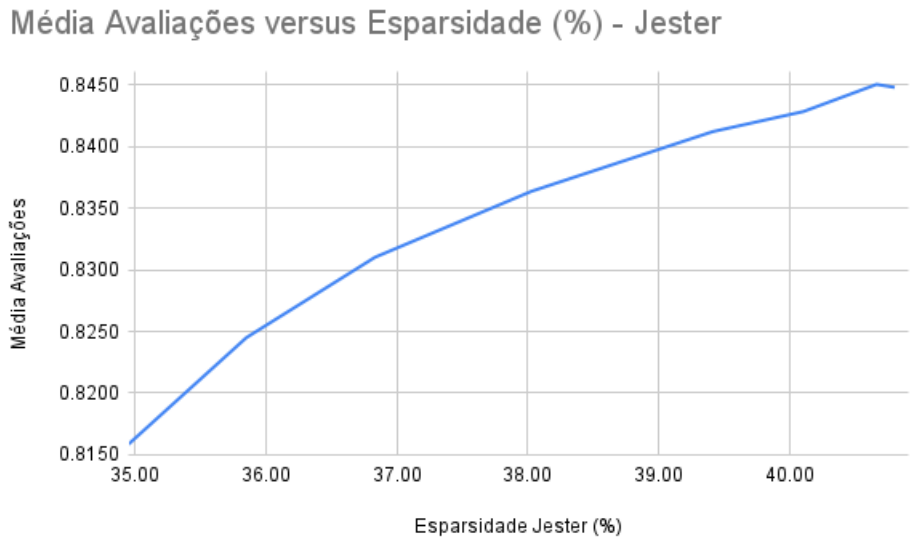


Figura 8 – Gráfico - Média Avaliações versus Esparsidade (%) - Jester.

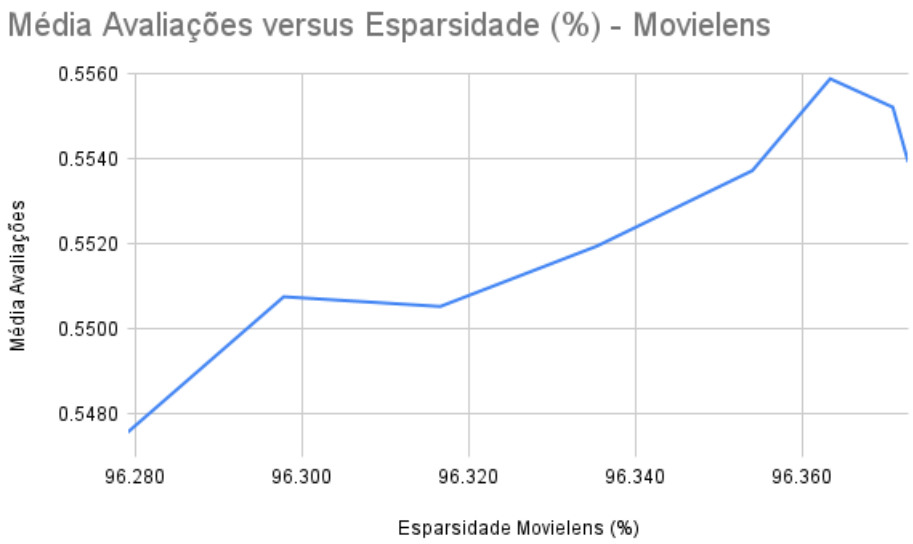


com significância estatística. Para este *dataset* foram consideradas nas partições interações de número mais elevado devido a esta questão da esparsidade.

Os experimentos modelados para a partida-fria apresentaram que a proposta de solução pode atuar neste cenário, recomendando itens para usuários com poucas interações. No caso do *Filmtrust* os melhores resultados foram para X sendo 20 e 40, considerando os casos que possuem significância estatística, cujos valores de X coincidem com os resultados dos Casos 1 e 2.

Com relação ao *Jester*, os melhores resultados foram obtidos com X igual a 5, 10 e 20. Para o *Movielens*, os melhores resultados foram para o $MAP@5$ e 10 principalmente, pois

Figura 9 – Gráfico - Média Avaliações versus Esparsidade (%) - Movielens.



ganham do *baseline* com significância estatística.

Observamos também que aumentar X tende a melhorar o desempenho, mas até certo ponto. O desempenho pode se estabilizar ou começar a degradar. A razão para este comportamento é que, à medida que X aumenta, a proporção das notas estimadas aumenta em comparação com a proporção das notas reais na matriz enriquecida. Uma vez que as notas estimadas estão sujeitas a erros de estimativa, há um ponto que os ganhos com a redução da dispersão não se sobrepõem mais a crescente incerteza nos dados. Nos experimentos o desempenho caiu na maioria das bases de dados quando $X > 50$, devido ao fato de que as amostras se tornam menos confiáveis (as notas tendem a ser mais globais em vez de refletir o comportamento do usuário) e o método se torna mais sensível a ruídos. E para o cenário de *coldstart*, para as situações que obtiveram melhores resultados comparados com o *baseline*, notam-se diversos máximos locais, considerando o gráfico no aprendizado dos gostos dos usuários.

Há muita similaridade quanto ao desempenho dos Casos 1 e 2. Percebemos que, na maioria das situações, os valores de X são similares para ambos quando há bons desempenhos. A diferença entre os resultados é bem baixa. Para o *Filmtrust*, menos de 3% de todos os resultados possuem diferença maior que 1% entre os valores do Caso 1 e 2. Para o *Jester*, menos que 8,5% de todos os resultados possuem diferença maior que 5%. E, por fim, para o Movielens, menos que 9,5% de todos os resultados possuem diferença maior que 6% entre os dois casos.

Havia uma certa expectativa com relação ao desempenho da proposta para o Caso 3, abordado na Subseção 4.5.3, no qual apenas os M itens rotulados pelo co-treinamento são considerados como a própria recomendação, por talvez serem os "próximos"a serem recomendados dado a métrica utilizada. Mas ao serem confrontados com o conjunto de teste, os resultados foram pouco satisfatórios, possivelmente pela própria formulação da métrica, voltada para a

acucácia na predição e não para o ranqueamento.

5.3 Considerações Finais

Este trabalho apresentou uma abordagem de co-treimanento, que diminui a esparsidade ao processar dados com diferentes visões, obtendo um conjunto de dados enriquecido para um recomendador gerar listas *top-n*. Para os experimentos foram utilizados três conjuntos de dados de diferentes domínios (filmes e piadas), os resultados apresentaram que esta abordagem melhora a recomendação.

A principal vantagem ao enriquecer uma matriz de notas é lidar com a questão da alta esparsidade. Notamos que reduzir a esparsidade está diretamente relacionada com a melhoria da precisão dos resultados, visto que quanto mais rotulados exemplos, melhor será a capacidade dos algoritmos de recomendação no aprendizado do comportamento dos usuários. Foi observado também que esta abordagem alivia o problema de *coldstart* em um certo nível, considerando usuários que já visualizaram alguns itens.

CAPÍTULO

6

CONCLUSÕES E TRABALHOS FUTUROS

6.1 Resumo do Trabalho

Neste trabalho foi proposto analisar recomendações partindo de um conjunto de dados enriquecido. O enriquecimento foi feito por meio do co-treinamento, um *framework* de aprendizado semissupervisionado, que diminui a esparsidade processando dados com diferentes visões. No caso, os dois algoritmos utilizados no *framework* foram o Item-KNN e User-KNN, recomendadores tradicionais da filtragem colaborativa. O principal objetivo desse estudo foi investigar as recomendações *top-n* produzidas por uma matriz menos esparsa.

O algoritmo utilizado para gerar as recomendações finais, o *Bayesian Personalized Ranking Matrix Factorization* (BPRMF), prioriza o ranqueamento dos itens, gerando listas *top-n*. Consideramos 3 situações distintas para a recomendação final: o conjunto enriquecido como conjunto de treino do BPR, o conjunto enriquecido apenas no treinamento do BPR e para a recomendação utilizar o arquivo de treino original, por fim, apenas os itens que foram enriquecidos na etapa de co-treinamento serem os itens na recomendação final. Medimos a qualidade das recomedações com o *Mean Average Precison* (MAP) e o *Normalized Discounted Cumulative Gain* (NDCG).

Analisamos estas recomendações também para o cenário de *coldstart*, típico problema em sistemas de recomendação. Foram modeladas 3 partições distinas, considerando o número de interações dos usuários. Observamos que esta abordagem alivia este problema em algum nível.

Observamos que, de modo geral, verificou-se que gerar recomendações a partir de um conjunto de dados menos esparso obteve melhor desempenho comparado com os *baselines*, mostrando que há melhora no aprendizado do algoritmo ao partir de conjuntos de dados com itens enriquecidos. O que comprova a Hipótese 1: Realizar uma recomendação a partir de um conjunto de dados enriquecido, por meio de uma técnica de co-treinamento, pode gerar recomendações com boas acurácias, quando comparadas a conjuntos de dados sem enriquecimento.

Para o cenário de partida-fria, verificou-se que a proposta ameniza este problema com precisão, obtendo melhor desempenho em todas as bases de dados avaliadas. Isso mostra a eficácia deste modelo ao abordar o problema de *coldstart* com enriquecimento dos dados. Com isso, também comprovamos a Hipótese 2: Este conjunto enriquecido pode ser utilizado para reduzir o problema de partida-fria.

Os resultados de ambas as propostas geram expectativas positivas para utilizar este *framework* em conjunto, num *pipeline* de dados, com outras abordagens de recomendação.

6.2 Contribuições da Pesquisa

As principais contribuições desta pesquisa são:

- Validação do *framework* CoRec para o cenário de ranqueamento, uma vez que o conjunto de dados enriquecido permite maior aprendizado do algoritmo de recomendação;

- Gerar recomendações para o cenário de partida-fria, também utilizando o conjunto enriquecido, considerando determinados intervalos de interações. O conjunto enriquecido alivia o problema da esparsidade e resulta em recomendações com maior impacto comparado com o conjunto sem enriquecimento.

6.2.1 Conclusões e Aplicações

Neste trabalho foram realizados experimentos para analisar a qualidade das recomendações utilizando um conjunto de dados enriquecidos no recomendador. A análise e validação dos resultados mostraram a eficácia deste método.

Observou-se, também, que esta proposta alivia o problema de partida-fria e pode ser aplicado para lidar com este problema.

Devido à versatilidade da arquitetura, é possível alterar os algoritmos utilizados nas visões do co-treimanento, como também é possível utilizar outro algoritmo para gerar as recomendações finais.

Como mencionado no Capítulo 1, dados rotulados são caros. O *trade-off* aqui está em obter precisão nos *rankings*, porém há um alto custo, em tempo, no enriquecimento e treinamento deste recomendador com este *framework*. Por este motivo, é interessante averiguar a possibilidade de um estudo da estrutura dos conjuntos de dados e o quanto enriquecer destes conjuntos, para obter um intervalo confiável de X e diminuir custo computacional.

6.2.2 Artigo Submetido

Ao longo do desenvolvimento desta pesquisa submetemos um trabalho relacionado à técnica proposta na ferramenta. O trabalho foi submetido para a revista *Information Systems*.

- **Personalized Ranking Based On Enriched Data: A Co-Training Approach**
- **Abstract**: The growing supply of products, services and information on Web has led several systems to use and develop recommendation systems to suggest contents according to each specific user's preference. In these systems, a large amount of labeled data must be available to obtain good predictions. However, labeled data are often limited and expensive to obtain, since labelling usually requires time and human knowledge. Besides, usually, users are only interested in the best ranked recommendations, while lower-ranked suggestions are ignored. Since there is a lack of methods that promote enrichment (labelling) with ranking, this paper proposes a technique that recommends top-n items based on enriched data. The enrichment is based on a co-training method, whose learning uses multiple views, data divided into two disjoint subsets, to decrease the data sparsity. The choice of items to enrich is based on a confidence metric, which seeks items that are more relevant to the user. Then, these two enriched subsets are combined with an ensemble method. In our evaluation, we used real data sets from distinct domains and experiments show that the proposed method achieves better results, compared to baseline, when a recommender algorithm is used in this ensemble set.

6.3 Trabalhos Futuros

Nos experimentos foi identificado que há uma quantidade ideal no valor de X, relacionado com a quantidade de enriquecimento, que varia para cada conjunto de dados para a recomendação final com o algoritmo proposto, o BPR. Pretendemos, como trabalho futuro, um estudo teórico para obtenção de um intervalo deste valor para evitar custo computacional.

Outro trabalho futuro será a investigação de uma métrica de confiança otimizada para ranqueamentos e averiguar o quão relevante é a recomendação especificamente destes itens enriquecidos no co-treinamento.

Outra possibilidade é comparar este *framework* com outros métodos de enriquecimentos e outros algoritmos de recomendação, analisando suas diferenças e desempenhos. Para o cenário de partida-fria, pode-se comparar com outra abordagem de recomendação, como a baseada em conteúdo.

REFERÊNCIAS

ADOMAVICIUS, G.; KAMIREDDY, S.; KWON, Y. Towards more confident recommendations: Improving recommender systems using filtering approach based on rating variance. In: **Proc. of the 17th Workshop on Information Technology and Systems**. New York, NY, USA: Association for Computing Machinery, 2007. p. 152–157. Citado na página 57.

AGGARWAL, C. C. **Data classification: algorithms and applications**. New York, NY, USA: CRC Press, 2014. Citado na página 49.

AGGARWAL, C. C. *et al.* **Recommender systems**. Berlin/Heidelberg, Germany: Springer, 2016. v. 1. Citado nas páginas 26, 29, 30, 31, 32, 33, 36, 38, 39, 41, 44, 46 e 50.

AGGARWAL, C. C.; PARTHASARATHY, S. Mining massively incomplete data sets by conceptual reconstruction. In: **Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2001. p. 227–232. Citado na página 46.

BALCAN, M.-F.; BLUM, A.; YANG, K. Co-training and expansion: Towards bridging theory and practice. **Advances in neural information processing systems**, MIT Press, v. 17, p. 89–96, 2005. Citado na página 36.

BAR, A.; ROKACH, L.; SHANI, G.; SHAPIRA, B.; SCHCLAR, A. Improving simple collaborative filtering models using ensemble methods. In: SPRINGER. **International Workshop on Multiple Classifier Systems**. Berlin/Heidelberg, Germany, 2013. p. 1–12. Citado nas páginas 51 e 52.

BELL, R. M.; KOREN, Y.; VOLINSKY, C. The bellkor solution to the netflix prize. **KorBell Team's Report to Netflix**, Citeseer, 2007. Citado na página 49.

BLUM, A.; MANSOUR, Y. Efficient co-training of linear separators under weak dependence. In: **Conference on Learning Theory (COLT)**. Amsterdam, Netherlands: PMLR, 2017. v. 65. Citado nas páginas 48 e 49.

BLUM, A.; MITCHELL, T. Combining labeled and unlabeled data with co-training. In: **Proceedings of the eleventh annual conference on Computational learning theory**. New York, NY, USA: Association for Computing Machinery, 1998. p. 92–100. Citado nas páginas 27, 35 e 47.

BURKE, R. Hybrid recommender systems: Survey and experiments. **User modeling and user-adapted interaction**, Springer, v. 12, n. 4, p. 331–370, 2002. Citado nas páginas 33 e 34.

CANTADOR, I.; BRUSILOVSKY, P.; KUFLIK, T. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In: **Proceedings of the 5th ACM conference on Recommender systems**. New York, NY, USA: ACM, 2011. (RecSys 2011). Citado na página 64.

CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. **Semi-supervised learning, ser. Adaptive computation and machine learning**. Cambridge, MA, USA: Cambridge, MA: The MIT Press, 2006. Citado na página 35.

CHAPELLE, O.; SCHOLKOPF, B.; ZIEN, A. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. **IEEE Transactions on Neural Networks**, IEEE, v. 20, n. 3, p. 542–542, 2009. Citado na página 48.

CHEN, L.; PU, P. Trust building in recommender agents. In: CITESEER. **Proceedings of the Workshop on Web Personalization, Recommender Systems and Intelligent User Interfaces at the 2nd International Conference on E-Business and Telecommunication Networks**. Lausanne, Vaud, Switzerland, 2005. p. 135–145. Citado na página 31.

COSTA, A. F. D.; MANZATO, M. G.; CAMPELLO, R. J. Corec: a co-training approach for recommender systems. In: **Proceedings of the 33rd Annual ACM Symposium on Applied Computing**. New York, NY, USA: Association for Computing Machinery, 2018. p. 696–703. Citado nas páginas 27, 37, 49, 50, 52, 53, 57, 58 e 62.

____. Boosting collaborative filtering with an ensemble of co-trained recommenders. **Expert Systems with Applications**, Elsevier, v. 115, p. 427–441, 2019. Citado nas páginas 27, 50, 52 e 58.

COSTA, A. F. da; MANZATO, M. G. Case recommender: A recommender framework. In: SBC. **Anais Estendidos do XXII Simpósio Brasileiro de Sistemas Multimídia e Web**. Gramado, RS, Brasil, 2016. p. 99–102. Citado na página 63.

FELFERNIG, A.; FRIEDRICH, G.; GULA, B.; HITZ, M.; KRUGGEL, T.; LEITNER, G.; MELCHER, R.; RIEPAN, D.; STRAUSS, S.; TEPPAN, E. *et al.* Persuasive recommendation: serial position effects in knowledge-based recommender systems. In: SPRINGER. **International Conference on Persuasive Technology**. Palo Alto, CA, USA, 2007. p. 283–294. Citado na página 31.

GHAZANFAR, M. A.; PRUGEL, A. The advantage of careful imputation sources in sparse data-environment of recommender systems: Generating improved svd-based recommendations. **Informatica**, v. 37, n. 1, 2013. Citado na página 46.

GOLDBERG, K.; ROEDER, T.; GUPTA, D.; PERKINS, C. Eigentaste: A constant time collaborative filtering algorithm. **information retrieval**, Springer, v. 4, n. 2, p. 133–151, 2001. Citado na página 64.

GUHA, R.; KUMAR, R.; RAGHAVAN, P.; TOMKINS, A. Propagation of trust and distrust. In: **Proceedings of the 13th international conference on World Wide Web**. New York, NY, USA: Association for Computing Machinery, 2004. p. 403–412. Citado na página 47.

GUO, G. Improving the performance of recommender systems by alleviating the data sparsity and cold start problems. In: **Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence**. AAAI Press, 2013. (IJCAI '13), p. 3217–3218. ISBN 978-1-57735-633-2. Disponível em: <http://dl.acm.org/citation.cfm?id=2540128.2540617>. Citado na página 54.

____. Improving the performance of recommender systems by alleviating the data sparsity and cold start problems. In: **Twenty-Third International Joint Conference on Artificial Intelligence**. Menlo Park, California: AAAI Press / International Joint Conferences on Artificial Intelligence, 2013. Citado na página 57.

GUO, G.; ZHANG, J.; YORKE-SMITH, N. A novel bayesian similarity measure for recommender systems. In: . Menlo Park, California: AAAI Press / International Joint Conferences on Artificial Intelligence, 2013. Citado na página 64.

GUO, Q.; SUN, Z.; THENG, Y.-L. Exploiting side information for recommendation. In: SPRINGER. **International Conference on Web Engineering**. Berlin/Heidelberg, Germany, 2019. p. 569–573. Citado na página 47.

HAO, Z.; CHENG, Y.; CAI, R.; WEN, W.; WANG, L. A semi-supervised solution for cold start issue on recommender systems. In: SPRINGER. **Asia-Pacific Web Conference**. Berlin/Heidelberg, Germany, 2015. p. 805–817. Citado nas páginas 27 e 49.

HARRIS, C. R.; MILLMAN, K. J.; WALT, S. J. van der; GOMMERS, R.; VIRTANEN, P.; COURNAPEAU, D.; WIESER, E.; TAYLOR, J.; BERG, S.; SMITH, N. J.; KERN, R.; PICUS, M.; HOYER, S.; KERKWIJK, M. H. van; BRETT, M.; HALDANE, A.; R'ıO, J. F. del; WIEBE, M.; PETERSON, P.; G'eRARD-MARCHANT, P.; SHEPPARD, K.; REDDY, T.; WECKESSER, W.; ABBASI, H.; GOHLKE, C.; OLIPHANT, T. E. Array programming with NumPy. **Nature**, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, set. 2020. Disponível em: <https://doi.org/10.1038/s41586-020-2649-2>. Citado na página 64.

HASSANAT, A. B.; ABBADI, M. A.; ALTARAWNEH, G. A.; ALHASANAT, A. A. Solving the problem of the k parameter in the knn classifier using an ensemble learning approach. **arXiv preprint arXiv:1409.0919**, 2014. Citado na página 65.

HE, B.; OUNIS, I. A study of parameter tuning for term frequency normalization. In: **Proceedings of the Twelfth International Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 2003. (CIKM '03), p. 10–16. ISBN 1-58113-723-0. Disponível em: <http://doi.acm.org/10.1145/956863.956867>. Citado nas páginas 51 e 52.

HE, X.; LIAO, L.; ZHANG, H.; NIE, L.; HU, X.; CHUA, T.-S. Neural collaborative filtering. In: **Proceedings of the 26th international conference on world wide web**. New York, NY, USA: Association for Computing Machinery, 2017. p. 173–182. Citado na página 55.

HERLOCKER, J. L.; KONSTAN, J. A.; RIEDL, J. Explaining collaborative filtering recommendations. In: **Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work**. New York, NY, USA: ACM, 2000. (CSCW '00), p. 241–250. ISBN 1-58113-222-0. Disponível em: <http://doi.acm.org/10.1145/358916.358995>. Citado na página 32.

HUANG, Z.; CHEN, H.; ZENG, D. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. **ACM Transactions on Information Systems (TOIS)**, ACM New York, NY, USA, v. 22, n. 1, p. 116–142, 2004. Citado na página 35.

HWANG, W.-S.; LI, S.; KIM, S.-W.; LEE, K. Data imputation using a trust network for recommendation. In: **Proceedings of the 23rd International Conference on World Wide Web**. New York, NY, USA: Association for Computing Machinery, 2014. p. 299–300. Citado na página 46.

JAHRER, M.; TÖSCHER, A.; LEGENSTEIN, R. Combining predictions for accurate recommender systems. In: **Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining**. New York, NY, USA: Association for Computing Machinery, 2010. p. 693–702. Citado na página 49.

JANNACH, D.; ZANKER, M.; FELFERNIG, A.; FRIEDRICH, G. **Recommender systems: an introduction**. Cambridge, United Kingdom: Cambridge University Press, 2010. Citado nas páginas 26, 34, 37, 38, 40, 41 e 45.

JÄRVELIN, K.; KEKÄLÄINEN, J. Ir evaluation methods for retrieving highly relevant documents. In: ACM. **Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval**. Athens, Greece, 2000. p. 41–48. Citado na página 43.

KNIJNENBURG, B. P.; WILLEMSEN, M. C.; GANTNER, Z.; SONCU, H.; NEWELL, C. Explaining the user experience of recommender systems. **User Modeling and User-Adapted Interaction**, Springer, v. 22, n. 4-5, p. 441–504, 2012. Citado na página 55.

KOREN, Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: **Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining**. New York, NY, USA: Association for Computing Machinery, 2008. p. 426–434. Citado na página 37.

____. Factor in the neighbors: Scalable and accurate collaborative filtering. **ACM Transactions on Knowledge Discovery from Data (TKDD)**, ACM New York, NY, USA, v. 4, n. 1, p. 1–24, 2010. Citado na página 37.

LANGFORD, J. The cross validation problem. In: SPRINGER. **International Conference on Computational Learning Theory**. Berlin/Heidelberg, Germany, 2005. p. 687–688. Citado na página 64.

LITTLE, R. J.; RUBIN, D. B. **Statistical analysis with missing data**. Hoboken, New Jersey: John Wiley & Sons, 2019. v. 793. Citado na página 46.

LIU, T.-Y. *et al.* Learning to rank for information retrieval. **Foundations and Trends® in Information Retrieval**, Now Publishers, Inc., v. 3, n. 3, p. 225–331, 2009. Citado na página 43.

LIU, X.; LIU, Y.; ABERER, K.; MIAO, C. Personalized point-of-interest recommendation by mining users' preference transition. In: **Proceedings of the 22nd ACM international conference on Information & Knowledge Management**. New York, NY, USA: Association for Computing Machinery, 2013. p. 733–738. Citado na página 47.

MAZUROWSKI, M. A. Estimating confidence of individual rating predictions in collaborative filtering recommender systems. **Expert Systems with Applications**, Elsevier, v. 40, n. 10, p. 3847–3857, 2013. Citado na página 57.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V. *et al.* Scikit-learn: Machine learning in python. **Journal of machine learning research**, v. 12, n. Oct, p. 2825–2830, 2011. Citado na página 64.

RENDLE, S.; FREUDENTHALER, C.; GANTNER, Z.; SCHMIDT-THIEME, L. Bpr: Bayesian personalized ranking from implicit feedback. **arXiv preprint arXiv:1205.2618**, 2012. Citado na página 55.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Recommender systems: introduction and challenges. In: **Recommender systems handbook**. Berlin/Heidelberg, Germany: Springer, 2015. p. 1–34. Citado nas páginas 25, 26, 30, 32, 33, 34, 35, 36, 39, 43, 46, 49, 54 e 57.

RUSSELL, S.; NORVIG, P. Artificial intelligence: a modern approach. 2002. Citado na página 35.

SCHAFER, J. B.; FRANKOWSKI, D.; HERLOCKER, J.; SEN, S. Collaborative filtering recommender systems. In: **The adaptive web**. Berlin/Heidelberg, Germany: Springer, 2007. p. 291–324. Citado na página 30.

SCHEIN, A. I.; POPESCUL, A.; UNGAR, L. H.; PENNOCK, D. M. Methods and metrics for cold-start recommendations. In: **Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval**. New York, NY, USA: Association for Computing Machinery, 2002. p. 253–260. Citado na página 34.

SHEIKHPOUR, R.; SARRAM, M. A.; GHARAGHANI, S.; CHAHOOKI, M. A. Z. A survey on semi-supervised feature selection methods. **Pattern Recognition**, Elsevier, v. 64, p. 141–158, 2017. Citado na página 26.

SHI, Y.; LARSON, M.; HANJALIC, A. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 47, n. 1, p. 1–45, 2014. Citado na página 47.

STECK, H. Evaluation of recommendations: rating-prediction and ranking. In: **Proceedings of the 7th ACM conference on Recommender systems**. New York, NY, USA: Association for Computing Machinery, 2013. p. 213–220. Citado na página 31.

SU, X.; KHOSHGOFTAAR, T. M.; ZHU, X.; GREINER, R. Imputation-boosted collaborative filtering using machine learning classifiers. In: **Proceedings of the 2008 ACM symposium on Applied computing**. New York, NY, USA: Association for Computing Machinery, 2008. p. 949–950. Citado nas páginas 51 e 52.

SUN, M.; LI, F.; LEE, J.; ZHOU, K.; LEBANON, G.; ZHA, H. Learning multiple-question decision trees for cold-start recommendation. In: ACM. **Proceedings of the sixth ACM international conference on Web search and data mining**. Rome, Italy, 2013. p. 445–454. Citado na página 36.

SUN, S.; ZHANG, Q. Multiple-view multiple-learner semi-supervised learning. **Neural processing letters**, Springer, v. 34, n. 3, p. 229, 2011. Citado nas páginas 51 e 52.

TAN, Q.; CHAI, X.; NG, W.; LEE, D.-L. Applying co-training to clickthrough data for search engine adaptation. In: SPRINGER. **International Conference on Database Systems for Advanced Applications**. Berlin/Heidelberg, Germany, 2004. p. 519–532. Citado nas páginas 50 e 52.

TEAM, T. pandas development. **pandas-dev/pandas: Pandas**. Zenodo, 2020. Disponível em: <https://doi.org/10.5281/zenodo.3509134>. Citado na página 63.

USUNIER, N.; AMINI, M.-R.; GOUTTE, C. Multiview semi-supervised learning for ranking multilingual documents. In: SPRINGER. **Joint European Conference on Machine Learning and Knowledge Discovery in Databases**. Athens, Greece, 2011. p. 443–458. Citado na página 36.

VIRTANEN, P.; GOMMERS, R.; OLIPHANT, T. E.; HABERLAND, M.; REDDY, T.; COURNAPEAU, D.; BUROVSKI, E.; PETERSON, P.; WECKESSER, W.; BRIGHT, J.; van der Walt, S. J.; BRETT, M.; WILSON, J.; MILLMAN, K. J.; MAYOROV, N.; NELSON, A. R. J.; JONES,

E.; KERN, R.; LARSON, E.; CAREY, C. J.; POLAT, İ.; FENG, Y.; MOORE, E. W.; VanderPlas, J.; LAXALDE, D.; PERKTOLD, J.; CIMRMAN, R.; HENRIKSEN, I.; QUINTERO, E. A.; HARRIS, C. R.; ARCHIBALD, A. M.; RIBEIRO, A. H.; PEDREGOSA, F.; van Mulbregt, P.; SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. **Nature Methods**, v. 17, p. 261–272, 2020. Citado na página 64.

WANG, W.; ZHOU, Z.-H. Co-training with insufficient views. In: **Asian conference on machine learning**. Cambridge, MA: MIT Press, 2013. p. 467–482. Citado na página 49.

XU, C.; TAO, D.; XU, C. A survey on multi-view learning. **arXiv preprint arXiv:1304.5634**, 2013. Citado nas páginas 35, 36 e 48.

ZHANG, M.; TANG, J.; ZHANG, X.; XUE, X. Addressing cold start in recommender systems: A semi-supervised co-training algorithm. In: **Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval**. New York, NY, USA: Association for Computing Machinery, 2014. p. 73–82. Citado nas páginas 27 e 49.

ZHU, X. Semi-supervised learning literature survey. **Computer Science, University of Wisconsin-Madison**, v. 2, n. 3, p. 4, 2006. Citado na página 35.

ZHU, X.; GOLDBERG, A. B. Introduction to semi-supervised learning. **Synthesis lectures on artificial intelligence and machine learning**, Morgan & Claypool Publishers, v. 3, n. 1, p. 1–130, 2009. Citado nas páginas 26 e 48.

ZHU, X. J. **Semi-supervised learning literature survey**. Madison, WI, USA, 2005. Citado na página 26.

APÊNDICE

A

RESULTADOS DAS AVALIAÇÕES

Dataset		Sparsity (%)	MAP				NDCG			
			@5	@10	@50	@100	@5	@10	@50	@100
Filmtrust	Baseline	98.938	0.5174	0.5132	0.4937	0.4907	0.6006	0.6149	0.6175	0.6182
	1	98.920	**0.5232****	**0.5190****	**0.4985****	**0.4955****	**0.6032**	**0.6180***	**0.6200***	**0.6207***
	5	98.890	**0.5219**	**0.5178**	**0.4973**	**0.4942***	**0.6033**	**0.6181**	**0.6197**	**0.6202**
	10	98.839	**0.5222****	**0.5179***	**0.4978****	**0.4948****	**0.6038***	**0.6177**	**0.6200****	**0.6209****
	20	98.741	**0.5212***	**0.5178***	**0.4975***	**0.4943***	**0.6016**	**0.6176***	**0.6189**	**0.6195**
	30	98.641	**0.5211***	**0.5170***	**0.4967***	**0.4941****	**0.6015**	**0.6164**	**0.6184**	**0.6195**
	40	98.542	**0.5205**	**0.5163**	**0.4964***	**0.4933**	**0.6016**	**0.6164**	**0.6189**	**0.6193**
	50	98.443	**0.5225****	**0.5183****	**0.4978****	**0.4948****	**0.6035***	**0.6176***	**0.6198**	**0.6203**
Jester	Baseline	40.797	0.87863	0.8391	0.7569	0.7537	0.9136	0.8930	0.8620	0.8609
	1	40.658	**0.8786**	**0.8398**	**0.7573**	**0.7541**	0.9136	**0.8936**	**0.8622**	**0.8610**
	5	40.101	0.8769	0.8372	0.7547	0.7511	0.9121	0.8916	0.8600	0.8586
	10	39.408	0.8751	0.8356	0.7528	0.7486	0.9115	0.8906	0.8583	0.8567
	20	38.026	0.8701	0.8301	0.7479	0.7430	0.9077	0.8862	0.8536	0.8517
	30	36.834	0.8651	0.8244	0.7418	0.7367	0.9038	0.8815	0.8482	0.8462
	40	35.853	0.8582	0.8178	0.7340	0.7292	0.8985	0.8764	0.8417	0.8398
	50	34.954	0.8494	0.8081	0.7237	0.7193	0.8919	0.8690	0.8335	0.8317
Movielens	Baseline	96.373	0.5897	0.5515	0.4117	0.3513	0.6628	0.6495	0.6142	0.6004
	1	96.371	**0.5919**	**0.5531**	**0.4118**	**0.3521**	**0.6655**	**0.6512**	**0.6148**	**0.6013**
	5	96.363	**0.5927**	**0.5541**	**0.4123**	**0.3525**	**0.6662**	**0.6525**	**0.6152**	**0.6017**
	10	96.354	0.5888	**0.5517**	0.4113	**0.3515**	0.6621	**0.6499**	0.6142	0.6005
	20	96.335	0.5878	0.5504	0.4094	0.3492	0.6602	0.6479	0.6122	0.5985
	30	96.317	0.5864	0.5471	0.4080	0.3485	0.6603	0.6458	0.6108	0.5973
	40	96.298	0.5857	0.5485	0.4085	0.3485	0.6598	0.6470	0.6108	0.5973
	50	96.279	0.5823	0.5441	0.4052	0.3461	0.6566	0.6429	0.6082	0.5951

Tabela 18 – Comparação dos resultados do BPRMF para o Caso 1, discutido na Seção 4.5.1.

Dataset		Sparsity (%)	MAP @5	MAP @10	MAP @50	MAP @100	NDCG @5	NDCG @10	NDCG @50	NDCG @100
Filmtrust	Baseline	98.938	0.5174	0.5132	0.4937	0.4907	0.6006	0.6149	0.6175	0.6182
	1	98.920	**0.5213****	**0.5169****	**0.4967****	**0.4938****	**0.6022**	**0.6162**	**0.6187**	**0.6196**
	5	98.890	**0.5213***	**0.5169***	**0.4969***	**0.4940****	**0.6017**	**0.6167****	**0.6188***	**0.6195***
	10	98.839	**0.5203**	**0.5165**	**0.4963**	**0.4931***	**0.6011**	**0.6162**	**0.6185**	**0.6192**
	X 20	98.741	**0.5219****	**0.5185****	**0.4980****	**0.4951****	**0.6025***	**0.6188****	**0.6200****	**0.6208****
	30	98.641	**0.5225***	**0.5183****	**0.4977****	**0.4946****	**0.6036**	**0.6176****	**0.6199***	**0.6207***
	40	98.542	**0.5240****	**0.5193****	**0.4983****	**0.4954****	**0.6048****	**0.6191****	**0.6205****	**0.6212****
	50	98.443	**0.5224***	**0.5179****	**0.4971****	**0.4942****	**0.6030**	**0.6178***	**0.6195****	**0.6202****
Jester	Baseline	40.797	0.87863	0.8391	0.7569	0.7537	0.9136	0.8930	0.8620	0.8609
	1	40.658	0.8785	0.8390	0.7563	0.7530	**0.9138**	**0.8932**	0.8618	0.8606
	5	40.101	0.8758	0.8372	0.7549	0.7511	0.9118	0.8917	0.8606	0.8592
	10	39.408	0.8746	0.8356	0.7522	0.7476	0.9108	0.8903	0.8587	0.8570
	X 20	38.026	0.8701	0.8321	0.7483	0.7422	0.9078	0.8877	0.8558	0.8535
	30	36.834	0.8635	0.8254	0.7396	0.7323	0.9033	0.8830	0.8502	0.8474
	40	35.853	0.8565	0.8183	0.7286	0.7206	0.8980	0.8773	0.8429	0.8398
	50	34.954	0.8420	0.8047	0.7106	0.7021	0.8866	0.8665	0.8303	0.8270
Movielens	Baseline	96.373	0.5897	0.5515	0.4117	0.3513	0.6628	0.6495	0.6142	0.6004
	1	96.371	**0.5911**	**0.5527**	**0.4119**	**0.3518**	**0.6635**	**0.6504**	**0.6144**	**0.601**
	5	96.363	**0.5910**	**0.5531**	**0.4122**	**0.3521**	**0.6650**	**0.6516**	**0.6155**	**0.6018**
	10	96.354	**0.5898**	0.5506	0.4102	0.3499	0.6623	0.6486	0.6132	0.5994
	X 20	96.335	0.5866	0.5492	0.4088	0.3490	0.6598	0.6478	0.6119	0.5982
	30	96.317	0.5863	0.5482	0.4074	0.3476	0.6595	0.6463	0.6102	0.5968
	40	96.298	0.5863	0.5481	0.4075	0.3479	0.6593	0.6464	0.6100	0.5968
	50	96.279	0.5825	0.5456	0.4060	0.3466	0.6566	0.6440	0.6091	0.5954

Tabela 19 – Comparação dos resultados do BPRMF para o Caso 2, discutido na Seção 4.5.2.

Dataset			Sparsity (%)	MAP				NDCG			
				@5	@10	@50	@100	@5	@10	@50	@100
Filmtrust	Baseline		98.94	0.5174	0.5132	0.4937	0.4908	0.6007	0.6149	0.6175	0.6183
	X	1	99.60	0.0010	0.0010	0.0010	0.0010	0.0010	0.0010	0.0010	0.0010
		5	99.75	0.0039	0.0039	0.0039	0.0039	0.0043	0.0043	0.0043	0.0043
		10	99.69	0.0069	0.0069	0.0069	0.0069	0.0083	0.0084	0.0084	0.0084
		20	99.56	0.0111	0.0113	0.0114	0.0114	0.0144	0.0150	0.0152	0.0152
		30	99.43	0.0135	0.0144	0.0146	0.0146	0.0178	0.0200	0.0206	0.0206
		40	99.31	0.0152	0.0163	0.0166	0.0166	0.0202	0.0232	0.0244	0.0244
		50	99.20	0.0158	0.0172	0.0178	0.0178	0.0215	0.0254	0.0275	0.0275
Jester	Baseline		40.80	0.8786	0.8392	0.7569	0.7538	0.9137	0.8931	0.8620	0.8609
	X	1	99.28	0.0699	0.0699	0.0699	0.0699	0.0699	0.0699	0.0699	0.0699
		5	98.89	0.2584	0.2584	0.2584	0.2584	0.2726	0.2726	0.2726	0.2726
		10	98.35	0.3688	0.3693	0.3693	0.3693	0.4014	0.4025	0.4025	0.4025
		20	97.12	0.4381	0.4427	0.4430	0.4430	0.4866	0.4974	0.4982	0.4982
		30	95.97	0.4458	0.4548	0.4565	0.4565	0.4963	0.5178	0.5233	0.5233
		40	95.01	0.4449	0.4571	0.4608	0.4608	0.4938	0.5228	0.5349	0.5349
		50	94.12	0.4432	0.4563	0.4625	0.4625	0.4903	0.5218	0.5422	0.5422
Movielens	Baseline		96.37	0.5900	0.5505	0.4108	0.3510	0.6646	0.6497	0.6143	0.6006
	X	1	99.74	0.0027	0.0027	0.0027	0.0027	0.0027	0.0027	0.0027	0.0027
		5	99.91	0.0097	0.0097	0.0097	0.0097	0.0126	0.0126	0.0126	0.0126
		10	99.92	0.0142	0.0143	0.0143	0.0143	0.0212	0.0215	0.0215	0.0215
		20	99.91	0.0165	0.0187	0.0187	0.0187	0.0263	0.0315	0.0316	0.0316
		30	99.89	0.0162	0.0207	0.0213	0.0213	0.0251	0.0365	0.0385	0.0385
		40	99.87	0.0157	0.0208	0.0227	0.0227	0.0237	0.0371	0.0436	0.0436
		50	99.85	0.0161	0.0212	0.0244	0.0244	0.0233	0.0368	0.0486	0.0486

Tabela 20 – Comparação dos resultados do BPRMF para o Caso 3, discutido na Seção 4.5.3.

Dataset Filmtrust			MAP				NDCG			
	X	Interactions	@5	@10	@50	@100	@5	@10	@50	@100
Baseline	1	1 ∼ 20	0.3178	0.3246	0.3052	0.3036	0.4327	0.4670	0.4796	0.4812
Caso 1	1	1 ∼ 20	**0.3200**	**0.3265**	**0.3070**	**0.3053**	**0.4340**	**0.4693**	**0.4813**	**0.4825**
Caso 2	1	1 ∼ 20	**0.3217**	**0.3278**	**0.3077**	**0.3061**	**0.4344**	**0.4680**	**0.4802**	**0.4819**
Baseline	1	21 ∼ 40	0.5154	0.4882	0.4501	0.4467	0.6136	0.6040	0.5893	0.5898
Caso 1	1	21 ∼ 40	0.5133	0.4873	**0.4504**	0.4466	0.6102	0.6032	0.5888	0.5896
Caso 2	1	21 ∼ 40	**0.5159**	0.4872	**0.4503**	**0.4468**	**0.6139**	**0.6029**	**0.5897**	**0.5907**
Baseline	1	41 ∼ 60	**0.9145**	0.9132	**0.9100**	**0.9084**	**0.9293**	**0.9316**	**0.9308**	**0.9307**
Caso 1	1	41 ∼ 60	**0.9148**	0.9132	**0.9103**	**0.9085**	0.9282	0.9306	0.9304	0.9306
Caso 2	1	41 ∼ 60	**0.9150**	**0.9134**	**0.9100**	0.9082	0.9290	**0.9310**	**0.9302**	**0.9304**
Baseline	5	1 ∼ 20	0.3213	0.3274	0.3076	0.3060	0.4329	0.4684	0.4807	0.4822
Caso 1	5	1 ∼ 20	**0.3180**	**0.3267**	**0.3062**	0.3047	**0.4311**	**0.4700**	**0.4809**	**0.4826**
Caso 2	5	1 ∼ 20	**0.3216**	**0.3280**	**0.3078**	**0.3067**	0.4325	0.4683	0.4800	0.4819
Baseline	5	21 ∼ 40	0.5136	**0.4862**	**0.4494**	0.4451	0.6125	**0.6040**	**0.5889**	0.5888
Caso 1	5	21 ∼ 40	0.5136	**0.4862**	0.4489	0.4450	0.6106	0.6028	0.5881	0.5885
Caso 2	5	21 ∼ 40	**0.5184**	**0.4885**	**0.4520**	**0.4481**	**0.6152**	0.6038	**0.5904**	**0.5907**
Baseline	5	41 ∼ 60	0.9147	0.9130	0.9096	0.9081	0.9288	0.9306	0.9303	0.9305
Caso 1	5	41 ∼ 60	**0.9139**	**0.9130**	**0.9098**	0.9076	**0.9285**	**0.9309**	**0.9304**	0.9301
Caso 2	5	41 ∼ 60	0.9140	0.9125	0.9092	0.9072	**0.9292**	**0.9314**	**0.9304**	0.9304
Baseline	10	1 ∼ 20	0.3241	0.3294	0.3098	0.3083	0.4358	0.4691	0.4815	0.4831
Caso 1	10	1 ∼ 20	0.3186	0.3253	0.3055	0.3042	0.4331	0.4663	0.4789	0.4806
Caso 2	10	1 ∼ 20	**0.3210**	**0.3273**	**0.3071**	**0.3057**	**0.4344**	**0.4683**	**0.4803**	**0.4824**
Baseline	10	21 ∼ 40	0.5135	0.4859	0.4494	0.4465	0.6108	0.6034	0.5882	0.5897
Caso 1	10	21 ∼ 40	**0.5153**	**0.4851**	**0.4497**	0.4455	**0.6160**	**0.6033**	**0.5899**	**0.5895**
Caso 2	10	21 ∼ 40	**0.5145**	**0.4872**	**0.4500**	0.4457	0.6102	0.6031	0.5888	0.5885
Baseline	10	41 ∼ 60	**0.9146**	**0.9128**	0.9095	0.9078	**0.9291**	**0.9310**	**0.9304**	**0.9305**
Caso 1	10	41 ∼ 60	0.9142	0.9127	0.9092	0.9077	0.9289	0.9307	0.9300	0.9304
Caso 2	10	41 ∼ 60	**0.9146**	**0.9131**	**0.9101**	**0.9082**	**0.9281**	0.9303	0.9302	0.9301
Baseline	20	1 ∼ 20	0.3152	0.3219	0.3024	0.3011	0.4315	0.4655	0.4780	0.4796
Caso 1	20	1 ∼ 20	**0.3248**	**0.3309**	**0.3108**	**0.3095**	**0.4353**	**0.4706**	**0.4822**	**0.4842**
Caso 2	20	1 ∼ 20	**0.3233****	**0.3295****	**0.3096****	**0.3085****	**0.4360****	**0.4712****	**0.4825****	**0.4840***
Baseline	20	21 ∼ 40	0.5134	0.4839	0.4489	0.4447	**0.6130**	0.6030	0.5893	0.5898
Caso 1	20	21 ∼ 40	0.5109	**0.4863**	0.4485	0.4447	0.6095	**0.6046**	**0.5897**	**0.5899**
Caso 2	20	21 ∼ 40	**0.5168**	**0.4906****	**0.4521**	**0.4482***	**0.6127**	**0.6066***	**0.5901**	**0.5912**
Baseline	20	41 ∼ 60	0.9139	0.9125	0.9094	0.9076	0.9288	0.9301	0.9300	0.9300
Caso 1	20	41 ∼ 60	**0.9141**	**0.9126**	0.9091	0.9072	**0.9297**	**0.9312**	**0.9304**	**0.9304**
Caso 2	20	41 ∼ 60	**0.9149**	**0.9134**	**0.9103**	**0.9082**	**0.9296**	**0.9316***	**0.9313**	**0.9311**
Baseline	30	1 ∼ 20	**0.3203**	0.3272	0.3073	0.3059	**0.4336**	0.4684	0.4810	0.4826
Caso 1	30	1 ∼ 20	**0.3205**	0.3270	0.3069	0.3058	**0.4339**	0.4683	0.4792	0.4810
Caso 2	30	1 ∼ 20	**0.3249**	**0.3312**	**0.3103**	**0.3089**	**0.4378**	**0.4708**	**0.4833**	**0.4848**
Baseline	30	21 ∼ 40	0.5180	0.4892	0.4520	0.4483	0.6158	0.6052	0.5905	0.5915
Caso 1	30	21 ∼ 40	**0.5132**	**0.4857**	**0.4479**	**0.4443**	0.6118	0.6022	0.5874	0.5884
Caso 2	30	21 ∼ 40	0.5160	0.4875	0.4505	0.4465	0.6143	0.6039	0.5897	0.5900
Baseline	30	41 ∼ 60	0.9147	0.9136	0.9101	0.9085	**0.9286**	0.9313	**0.9305**	**0.9306**
Caso 1	30	41 ∼ 60	0.9138	0.9119	0.9089	0.9076	**0.9291**	0.9312	0.9308	0.9309
Caso 2	30	41 ∼ 60	0.9141	0.9125	0.9089	0.9069	0.9291	0.9308	0.9300	0.9301
Baseline	40	1 ∼ 20	0.3224	0.3282	0.3085	0.3071	0.4339	0.4688	0.4814	0.4830
Caso 1	40	1 ∼ 20	**0.3280****	**0.3320****	**0.3124****	**0.3113****	**0.4393***	**0.4717**	**0.4835**	**0.4854***
Caso 2	40	1 ∼ 20	**0.3273***	**0.3327***	**0.3122***	**0.3109***	**0.4392***	**0.4727***	**0.4843**	**0.4860***
Baseline	40	21 ∼ 40	0.5161	0.4894	0.4523	0.4483	0.6093	0.6036	0.5896	0.5901
Caso 1	40	21 ∼ 40	0.5136	0.4882	0.4510	0.4468	**0.6115**	**0.6042**	**0.5901**	0.5899
Caso 2	40	21 ∼ 40	**0.5176**	0.4887	0.4490	0.4454	**0.6156**	**0.6060**	0.5892	0.5895
Baseline	40	41 ∼ 60	0.9147	0.9140	0.9106	0.9091	0.9286	0.9317	0.9310	0.9314
Caso 1	40	41 ∼ 60	**0.9149**	0.9126	0.9093	0.9072	**0.9302**	0.9310	0.9306	0.9304
Caso 2	40	41 ∼ 60	0.9141	0.9123	0.9091	0.9072	0.9287	0.9305	0.9302	0.9298
Baseline	50	1 ∼ 20	0.3224	0.3291	0.3086	0.3068	0.4343	0.4697	0.4815	0.4829
Caso 1	50	1 ∼ 20	**0.3253**	**0.3303**	**0.3104**	**0.3094**	**0.4382**	**0.4710**	**0.4825**	**0.4841**
Caso 2	50	1 ∼ 20	**0.3266**	**0.3315**	**0.3109**	**0.3097**	**0.4385**	**0.4715**	**0.4832**	**0.4849**
Baseline	50	21 ∼ 40	0.5164	0.4876	0.4508	0.4472	0.6147	0.6047	0.5905	0.5910
Caso 1	50	21 ∼ 40	**0.5179**	**0.4898**	**0.4520**	**0.4480**	0.6123	0.6036	0.5888	0.5900
Caso 2	50	21 ∼ 40	0.5141	**0.4876**	0.4497	0.4457	0.6105	0.6040	0.5893	0.5892
Baseline	50	41 ∼ 60	0.9145	0.9134	0.9100	0.9082	0.9287	0.9312	0.9306	0.9308
Caso 1	50	41 ∼ 60	0.9129	0.9103	0.9073	0.9053	0.9282	0.9297	0.9296	0.9295
Caso 2	50	41 ∼ 60	0.9129	0.9102	0.9072	0.9057	0.9283	0.9297	0.9292	0.9295

Tabela 21 – Comparação dos resultados do BPRMF para o cenário de partida-fria - Dataset Filmtrust

Dataset Jester			MAP				NDCG			
	X	Interactions	@5	@10	@50	@100	@5	@10	@50	@100
Baseline	1	30 ∼ 50	0.7759	0.7143	0.5796	0.5689	0.8392	0.8036	0.7472	0.7432
Caso 1	1	30 ∼ 50	0.7723	0.7130	0.5785	0.5675	0.8369	0.8027	0.7465	0.7424
Caso 2	1	30 ∼ 50	**0.7812**	**0.7166**	**0.5806**	**0.5690**	**0.8423**	**0.8052**	**0.7480**	**0.7437**
Baseline	1	51 ∼ 70	0.7913	0.7327	0.6029	0.5966	0.8495	0.8166	0.7641	0.7618
Caso 1	1	51 ∼ 70	0.7875	0.7293	0.6008	0.5945	0.8472	0.8148	0.7626	0.7603
Caso 2	1	51 ∼ 70	0.7885	0.7302	0.6000	0.5935	0.8488	0.8159	0.7626	0.7602
Baseline	1	71 ∼ 90	0.8645	0.8068	0.6896	0.6891	0.9060	0.8782	0.8393	0.8391
Caso 1	1	71 ∼ 90	0.8642	0.8065	0.6890	0.6886	0.9057	0.8774	0.8387	0.8386
Caso 2	1	71 ∼ 90	**0.8646**	**0.8069**	0.6891	0.6886	0.9059	0.8778	0.8392	0.8390
Baseline	5	30 ∼ 50	0.7802	0.7184	0.5830	0.5719	0.8404	0.8058	0.7494	0.7455
Caso 1	5	30 ∼ 50	**0.8136****	**0.7446****	**0.6169****	**0.6089****	**0.8673****	**0.8276****	**0.7761****	**0.7733****
Caso 2	5	30 ∼ 50	**0.7804**	**0.7187**	0.5808	**0.5681**	**0.8417**	0.8066	0.7482	0.7435
Baseline	5	51 ∼ 70	0.7885	0.7307	0.6011	0.5954	0.8484	0.8159	0.7630	0.7610
Caso 1	5	51 ∼ 70	**0.8243****	**0.7609****	**0.6421****	**0.6374****	**0.8739****	**0.8408****	**0.7929****	**0.7912****
Caso 2	5	51 ∼ 70	0.7867	0.7297	0.5997	0.5925	0.8462	0.8149	0.7613	0.7586
Baseline	5	71 ∼ 90	0.8631	0.8056	0.6886	0.6882	0.9042	0.8767	0.8382	0.8381
Caso 1	5	71 ∼ 90	**0.8797****	**0.8173****	**0.7046****	**0.7043****	**0.9158****	**0.8869****	**0.8500****	**0.8499****
Caso 2	5	71 ∼ 90	0.8615	0.8035	0.6874	0.6869	0.9032	0.8749	0.8368	0.8366
Baseline	10	30 ∼ 50	0.7728	0.7122	0.5796	0.5691	0.8361	0.8012	0.7466	0.7430
Caso 1	10	30 ∼ 50	**0.8081****	**0.7405****	**0.6093****	**0.6000****	**0.8621****	**0.8245****	**0.7707****	**0.7673****
Caso 2	10	30 ∼ 50	**0.7815***	**0.7171**	0.5785	0.5627	**0.8429***	**0.8050**	**0.7471**	0.7409
Baseline	10	51 ∼ 70	0.7860	0.7287	0.6005	0.5945	0.8454	0.8141	0.7622	0.7601
Caso 1	10	51 ∼ 70	**0.8184****	**0.7540****	**0.6324****	**0.6274****	**0.8697****	**0.8352****	**0.7861****	**0.7843****
Caso 2	10	51 ∼ 70	0.7852	0.7273	0.5960	0.5867	0.8454	0.8130	0.7589	0.7554
Baseline	10	71 ∼ 90	0.8652	0.8073	0.6906	0.6901	0.9064	0.8782	0.8398	0.8397
Caso 1	10	71 ∼ 90	**0.8722****	**0.8090**	**0.6959****	**0.6955****	**0.9098***	**0.8806***	**0.8435****	**0.8434****
Caso 2	10	71 ∼ 90	0.8554	0.7990	0.6811	0.6804	0.8990	0.8717	0.8331	0.8329
Baseline	20	30 ∼ 50	0.7729	0.7128	0.5792	0.5679	0.8376	0.8035	0.7469	0.7428
Caso 1	20	30 ∼ 50	**0.7923****	**0.7241****	**0.5920****	**0.5818****	**0.8508****	**0.8126****	**0.7576****	**0.7538****
Caso 2	20	30 ∼ 50	**0.7848****	**0.7228****	**0.5811**	0.5601	**0.8440***	**0.8088***	**0.7480**	0.7400
Baseline	20	51 ∼ 70	0.7859	0.7290	0.6005	0.5946	0.8457	0.8142	0.7620	0.7599
Caso 1	20	51 ∼ 70	**0.8050****	**0.7414****	**0.6188****	**0.6135****	**0.8605****	**0.8257****	**0.7752****	**0.7733****
Caso 2	20	51 ∼ 70	0.7813	0.7246	0.5916	0.5794	0.8431	0.8109	0.7556	0.7509
Baseline	20	71 ∼ 90	0.8667	0.8094	0.6911	0.6906	0.9069	0.8792	0.8402	0.8401
Caso 1	20	71 ∼ 90	0.8612	0.7970	0.6842	0.6839	0.9024	0.8716	0.8336	0.8335
Caso 2	20	71 ∼ 90	0.8396	0.7861	0.6703	0.6694	0.8891	0.8625	0.8253	0.8251
Baseline	30	30 ∼ 50	0.7752	0.7132	0.5802	0.5695	0.8387	0.8020	0.7470	0.7432
Caso 1	30	30 ∼ 50	**0.7795**	0.7125	0.5795	0.5688	**0.8429**	**0.8039**	**0.7483**	**0.7442**
Caso 2	30	30 ∼ 50	**0.7811**	**0.7186**	0.5689	0.5441	**0.8412**	**0.8053**	0.7403	0.7306
Baseline	30	51 ∼ 70	0.7888	0.7309	0.6028	0.5968	0.8468	0.8153	0.7634	0.7613
Caso 1	30	51 ∼ 70	**0.7929**	0.7293	**0.6065***	**0.6011***	**0.8512**	**0.8165**	**0.7655**	**0.7635**
Caso 2	30	51 ∼ 70	0.7704	0.7140	0.5790	0.5644	0.8356	0.8036	0.7475	0.7417
Baseline	30	71 ∼ 90	0.8638	0.8068	0.6900	0.6895	0.9044	0.8771	0.8388	0.8386
Caso 1	30	71 ∼ 90	0.8234	0.7632	0.6393	0.6339	0.8741	0.8426	0.7958	0.7938
Caso 2	30	71 ∼ 90	0.8309	0.7764	0.6573	0.6562	0.8835	0.8556	0.8172	0.8168
Baseline	40	30 ∼ 50	0.7758	0.7150	0.5808	0.5702	0.8397	0.8047	0.7487	0.7450
Caso 1	40	30 ∼ 50	0.7680	0.6994	0.5642	0.5526	0.8361	0.7948	0.7365	0.7319
Caso 2	40	30 ∼ 50	0.7628	0.7009	0.5444	0.5168	0.8283	0.7909	0.7231	0.7122
Baseline	40	51 ∼ 70	0.7900	0.7316	0.6019	0.5961	0.8502	0.8171	0.7640	0.7619
Caso 1	40	51 ∼ 70	0.7825	0.7187	0.5950	0.5898	0.8437	0.8083	0.7559	0.7540
Caso 2	40	51 ∼ 70	0.7620	0.7051	0.5612	0.5453	0.8289	0.7964	0.7358	0.7296
Baseline	40	71 ∼ 90	0.8661	0.8086	0.6907	0.6902	0.9060	0.8788	0.8399	0.8398
Caso 1	40	71 ∼ 90	0.8416	0.7750	0.6650	0.6648	0.8879	0.8539	0.8148	0.8148
Caso 2	40	71 ∼ 90	0.8219	0.7669	0.6449	0.6437	0.8768	0.8485	0.8096	0.8092
Baseline	50	30 ∼ 50	0.7775	0.7178	0.5831	0.5722	0.8410	0.8065	0.7501	0.7461
Caso 1	50	30 ∼ 50	0.7604	0.6926	0.5530	0.5423	0.8277	0.7879	0.7277	0.7236
Caso 2	50	30 ∼ 50	0.7361	0.6774	0.5113	0.4824	0.8053	0.7716	0.6986	0.6874
Baseline	50	51 ∼ 70	0.7902	0.7320	0.6018	0.5957	0.8494	0.8171	0.7634	0.7612
Caso 1	50	51 ∼ 70	0.7733	0.7094	0.5822	0.5775	0.8364	0.7999	0.7452	0.7434
Caso 2	50	51 ∼ 70	0.7348	0.6801	0.5281	0.5112	0.8072	0.7758	0.7122	0.7056
Baseline	50	71 ∼ 90	0.8647	0.8071	0.6895	0.6889	0.9060	0.8776	0.8389	0.8387
Caso 1	50	71 ∼ 90	0.8273	0.7613	0.6531	0.6530	0.8778	0.8435	0.8036	0.8036
Caso 2	50	71 ∼ 90	0.8046	0.7499	0.6229	0.6216	0.8645	0.8359	0.7950	0.7946

Tabela 22 – Comparação dos resultados do BPRMF para o cenário de partida-fria - Dataset Jester

Dataset Movielens			MAP				NDCG			
	X	Interactions	@5	@10	@50	@100	@5	@10	@50	@100
Baseline	1	$1 \sim 20$	0.0948	0.1037	0.1006	0.0931	0.1373	0.1626	0.2027	0.2145
Caso 1	1	$1 \sim 20$	**0.1034**	**0.1118**	**0.1046**	**0.0966**	**0.1482**	**0.1735**	**0.2064**	**0.2155**
Caso 2	1	$1 \sim 20$	**0.1055**	**0.1143**	**0.1097**	**0.1039**	**0.1505**	**0.1763**	**0.2096**	**0.2255**
Baseline	1	$21 \sim 40$	0.1404	0.1470	0.1328	0.1211	0.1918	0.2151	0.2461	0.2495
Caso 1	1	$21 \sim 40$	0.1343	0.1409	0.1286	0.1163	0.1853	0.2092	0.2418	0.2454
Caso 2	1	$21 \sim 40$	0.1292	0.1398	0.1273	0.1167	0.1753	0.2057	0.2380	0.2446
Baseline	1	$41 \sim 60$	0.2364	0.2414	0.1935	0.1661	0.3133	0.3415	0.3483	0.3382
Caso 1	1	$41 \sim 60$	**0.2388**	**0.2420**	0.1927	**0.1667**	**0.3183**	**0.3476**	**0.3489**	**0.3412**
Caso 2	1	$41 \sim 60$	**0.2378**	**0.2429**	**0.1961**	**0.1678**	0.3114	**0.3443**	**0.3500**	**0.3403**
Baseline	5	$1 \sim 20$	0.0948	0.1037	0.1006	0.0931	0.1373	0.1626	0.2027	0.2145
Caso 1	5	$1 \sim 20$	**0.1079**	**0.1162**	**0.1090**	**0.0990**	**0.1496**	**0.1796**	**0.2094**	**0.2170**
Caso 2	5	$1 \sim 20$	**0.1183**	**0.1243**	**0.1114**	**0.1033**	**0.1671**	**0.1866**	**0.2142**	**0.2257**
Baseline	5	$21 \sim 40$	0.1404	0.1470	0.1328	0.1211	0.1918	0.2151	0.2461	0.2495
Caso 1	5	$21 \sim 40$	0.1376	0.1456	**0.1337**	**0.1222**	0.1866	0.2113	0.2443	**0.2516**
Caso 2	5	$21 \sim 40$	0.1385	0.1448	**0.1332**	0.1205	0.1913	0.2110	**0.2485**	**0.2529**
Baseline	5	$41 \sim 60$	0.2364	0.2414	0.1935	0.1661	0.3133	0.3415	0.3483	0.3382
Caso 1	5	$41 \sim 60$	**0.2463**	**0.2483**	**0.2004**	**0.1735**	**0.3277**	**0.3520**	**0.3568**	**0.3478**
Caso 2	5	$41 \sim 60$	**0.2466**	**0.2483**	**0.2046***	**0.1735**	**0.3200**	**0.3449**	**0.3573**	**0.3447**
Baseline	10	$1 \sim 20$	0.0948	0.1037	0.1006	0.0931	0.1373	0.1626	0.2027	0.2145
Caso 1	10	$1 \sim 20$	**0.1167***	**0.1276***	**0.1148**	**0.1049**	**0.1634**	**0.1899**	**0.2159**	**0.2196**
Caso 2	10	$1 \sim 20$	**0.0984**	**0.1068**	0.0992	0.0905	0.1350	0.1593	0.2001	0.2062
Baseline	10	$21 \sim 40$	0.1404	0.1470	0.1328	0.1211	0.1918	0.2151	0.2461	0.2495
Caso 1	10	$21 \sim 40$	0.1348	0.1441	0.1305	0.1191	0.1815	0.2091	0.2422	0.2458
Caso 2	10	$21 \sim 40$	0.1324	0.1415	0.1268	0.1152	0.1796	0.2088	0.2388	0.2426
Baseline	10	$41 \sim 60$	0.2364	0.2414	0.1935	0.1661	0.3133	0.3415	0.3483	0.3382
Caso 1	10	$41 \sim 60$	**0.2390**	**0.2433**	**0.1974**	**0.1696**	**0.3168**	**0.3460**	**0.3538**	**0.3424**
Caso 2	10	$41 \sim 60$	0.2340	0.2358	0.1914	0.1624	0.3124	0.3400	0.3449	0.3370
Baseline	20	$1 \sim 20$	0.0948	0.1037	0.1006	0.0931	0.1373	0.1626	0.2027	0.2145
Caso 1	20	$1 \sim 20$	**0.1075***	**0.1149**	**0.1050**	**0.0962**	**0.1461**	**0.1698**	**0.2056**	0.2126
Caso 2	20	$1 \sim 20$	**0.1031**	**0.1100**	**0.1050**	**0.0941**	**0.1445**	**0.1665**	**0.2048**	0.2087
Baseline	20	$21 \sim 40$	0.1404	0.1470	0.1328	0.1211	0.1918	0.2151	0.2461	0.2495
Caso 1	20	$21 \sim 40$	0.1265	0.1345	0.1220	0.1099	0.1718	0.1985	0.2327	0.2375
Caso 2	20	$21 \sim 40$	0.1280	0.1386	0.1283	0.1144	0.1719	0.2041	0.2369	0.2408
Baseline	20	$41 \sim 60$	0.2364	0.2414	0.1935	0.1661	0.3133	0.3415	0.3483	0.3382
Caso 1	20	$41 \sim 60$	**0.2482**	**0.2512**	**0.2000**	**0.1715**	**0.3239**	**0.3528**	**0.3534**	**0.3453**
Caso 2	20	$41 \sim 60$	**0.2492**	**0.2554**	**0.2011**	**0.1737**	**0.3203**	**0.3521**	**0.3531**	**0.3438**
Baseline	30	$1 \sim 20$	0.0948	0.1037	0.1006	0.0931	0.1373	0.1626	0.2027	0.2145
Caso 1	30	$1 \sim 20$	**0.1203***	**0.1238**	**0.1132**	**0.1041**	**0.1600**	**0.1789**	**0.2136**	**0.2192**
Caso 2	30	$1 \sim 20$	**0.1010**	**0.1066**	0.0979	0.0916	**0.1396**	0.1592	0.1945	0.2069
Baseline	30	$21 \sim 40$	0.1404	0.1470	0.1328	0.1211	0.1918	0.2151	0.2461	0.2495
Caso 1	30	$21 \sim 40$	0.1344	0.1392	0.1284	0.1166	0.1893	0.2087	0.2427	0.2475
Caso 2	30	$21 \sim 40$	0.1315	0.1400	0.1282	0.1169	0.1837	0.2099	0.2386	0.2451
Baseline	30	$41 \sim 60$	0.2364	0.2414	0.1935	0.1661	0.3133	0.3415	0.3483	0.3382
Caso 1	30	$41 \sim 60$	0.2335	0.2374	0.1905	0.1641	0.3093	0.3393	0.3437	0.3360
Caso 2	30	$41 \sim 60$	**0.2462**	**0.2466**	**0.1985**	**0.1690**	**0.3172**	0.3385	0.3475	**0.3397**
Baseline	40	$1 \sim 20$	0.0948	0.1037	0.1006	0.0931	0.1373	0.1626	0.2027	0.2145
Caso 1	40	$1 \sim 20$	**0.1093**	**0.1178**	**0.1067**	**0.0988**	**0.1527**	**0.1782**	0.2024	0.2127
Caso 2	40	$1 \sim 20$	**0.1117***	**0.1176***	**0.1063**	**0.0963**	**0.1524**	**0.1765**	0.2007	0.2121
Baseline	40	$21 \sim 40$	0.1404	0.1470	0.1328	0.1211	0.1918	0.2151	0.2461	0.2495
Caso 1	40	$21 \sim 40$	0.1349	0.1410	0.1303	0.1184	0.1799	0.2034	0.2409	0.2461
Caso 2	40	$21 \sim 40$	0.1293	0.1376	0.1243	0.1116	0.1728	0.1992	0.2315	0.2357
Baseline	40	$41 \sim 60$	0.2364	0.2414	0.1935	0.1661	0.3133	0.3415	0.3483	0.3382
Caso 1	40	$41 \sim 60$	0.2325	0.2364	0.1930	0.1638	0.3104	0.3387	0.3461	0.3360
Caso 2	40	$41 \sim 60$	**0.2460**	**0.2462**	**0.2013**	**0.1702**	**0.3181**	0.3392	**0.3498**	**0.3396**
Baseline	50	$1 \sim 20$	0.0948	0.1037	0.1006	0.0931	0.1373	0.1626	0.2027	0.2145
Caso 1	50	$1 \sim 20$	**0.1061**	**0.1119**	**0.1061**	**0.0974**	**0.1448**	**0.1663**	0.2017	0.2123
Caso 2	50	$1 \sim 20$	**0.0998**	**0.1071**	0.0979	0.0891	**0.1436**	**0.1677**	0.1969	0.2050
Baseline	50	$21 \sim 40$	0.1404	0.1470	0.1328	0.1211	0.1918	0.2151	0.2461	0.2495
Caso 1	50	$21 \sim 40$	0.1301	0.1372	0.1261	0.1137	0.1811	0.2037	0.2378	0.2420
Caso 2	50	$21 \sim 40$	0.1322	0.1389	0.1254	0.1154	0.1810	0.2055	0.2367	0.2414
Baseline	50	$41 \sim 60$	0.2364	0.2414	0.1935	0.1661	0.3133	0.3415	0.3483	0.3382
Caso 1	50	$41 \sim 60$	0.2288	0.2325	0.1934	0.1656	0.3034	0.3297	0.3456	0.3370
Caso 2	50	$41 \sim 60$	**0.2407**	**0.2454**	**0.1978**	**0.1688**	**0.3136**	0.3412	**0.3496**	**0.3391**

Tabela 23 – Comparação dos resultados do BPRMF para o cenário de partida-fria - Dataset Movielens