Pedro Reis Pires

Representação Vetorial Distribuída em Sistemas de Recomendação

Sorocaba, SP

26 de Março de 2021

Pedro Reis Pires

Representação Vetorial Distribuída em Sistemas de Recomendação

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação (PPGCC-So) da Universidade Federal de São Carlos como parte dos requisitos exigidos para a obtenção do título de Mestre em Ciência da Computação. Linha de pesquisa: Computação Científica e Inteligência Computacional.

Universidade Federal de São Carlos – UFSCar

Centro de Ciências em Gestão e Tecnologia – CCGT

Programa de Pós-Graduação em Ciência da Computação – PPGCC-So

Orientador: Prof. Dr. Tiago Agostinho de Almeida

Sorocaba, SP

26 de Março de 2021

Reis Pires, Pedro

Representação vetorial distribuída em sistemas de recomendação / Pedro Reis Pires -- 2021.
140f.

Dissertação (Mestrado) - Universidade Federal de São Carlos, campus Sorocaba, Sorocaba
Orientador (a): Tiago Agostinho de Almeida
Banca Examinadora: Tiago Agostinho de Almeida, Anisio Mendes Lacerda, Leandro Balby Marinho
Bibliografia

1. Sistemas de recomendação. 2. Filtragem colaborativa. 3. Modelos de embeddings. I. Reis Pires, Pedro. II. Título.

Ficha catalográfica desenvolvida pela Secretaria Geral de Informática (SIn)

DADOS FORNECIDOS PELO AUTOR

Bibliotecário responsável: Maria Aparecida de Lourdes Mariano - CRB/8 6979

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências em Gestão e Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Relatório de Defesa de Dissertação
Candidato: Pedro Reis Pires

Aos 26/03/2021, às 14:00, realizou-se na Universidade Federal de São Carlos, nas formas e termos do Regimento Interno do Programa de Pós-Graduação em Ciência da Computação, a defesa de dissertação de mestrado sob o título: Representação Vetorial Distribuída em Sistemas de Recomendação, apresentada pelo candidato Pedro Reis Pires. Ao final dos trabalhos, a banca examinadora reuniu-se em sessão reservada para o julgamento, tendo os membros chegado ao seguinte resultado:

Participantes da Banca	**Função**	**Instituição**	**Resultado**	**Resultado Final**
Prof. Dr. Tiago Agostinho de Almeida	Presidente	UFSCar	Aprovado	Aprovado
Prof. Dr. Anisio Mendes Lacerda	Titular	UFMG	Aprovado	
Prof. Dr. Leandro Balby Marinho	Titular	UFCG	Aprovado	

Parecer da Comissão Julgadora*:

O candidato abordou um tema relevante e atual. A dissertação está bem escrita e estruturada. Quanto à apresentação e arguição, o candidato foi claro e demonstrou segurança e conhecimento do tema.

Encerrada a sessão reservada, o presidente informou ao público presente o resultado. Nada mais havendo a tratar, a sessão foi encerrada e, para constar, eu, Tiago Agostinho de Almeida, representante do Programa de Pós-Graduação em Ciência da Computação, lavrei o presente relatório, assinado por mim e pelos membros da banca examinadora.

Prof. Dr. Tiago Agostinho de Almeida

Representante do PPG: Tiago Agostinho de Almeida

Prof. Dr. Anisio Mendes Lacerda

Prof. Dr. Leandro Balby Marinho

Certifico que a defesa realizou-se com a participação à distância do(s) membro(s) Tiago Agostinho de Almeida, Anisio Mendes Lacerda, Leandro Balby Marinho e, depois das arguições e deliberações realizadas, o(s) participante(s) à distância está(ao) de acordo com o conteúdo do parecer da banca examinadora redigido neste relatório de defesa.

Prof. Dr. Tiago Agostinho de Almeida

(X) Não houve alteração no título ( ) Houve alteração no título. O novo título passa a ser:

Observações:
a) Se o candidato for reprovado por algum dos membros, o preenchimento do parecer é obrigatório.
b) Para gozar dos direitos do título de Mestre ou Doutor em Ciência da Computação, o candidato ainda precisa ter sua dissertação ou tese homologada pelo Conselho de Pós-Graduação da UFSCar.

Aos meus pais, Roberto e Mônica, e à minha companheira, Gabriela

Agradecimentos

Agradeço,

aos meus pais, Roberto e Mônica, por todo o amor, dedicação e ajuda que sempre me deram. Agradeço também por serem os principais responsáveis por quem sou hoje e por todas as minhas conquistas.

à minha companheira, Gabriela, pelo carinho e apoio nos melhores e piores momentos; pela companhia e paciência em um período atípico de pandemia e isolamento; e pelo amor e motivação que sempre me deu e, tenho certeza, continuará dando.

ao meu orientador, Tiago, por aceitar ser um mentor desde meus primeiros passos no mundo acadêmico e por todos os ensinamentos, orientações e sacrifícios que tornaram essa dissertação uma realidade.

aos meus amigos, que mesmo distantes continuaram presentes em minha vida, contribuindo para que o período de desenvolvimento dessa pesquisa fosse mais leve.

aos meus companheiros de pesquisa, Bruno e Amanda, pelas conversas, aprendizados e contribuições com os resultados aqui presentes.

aos meus companheiros de laboratório e amigos que diretamente contribuíram com esta pesquisa, possibilitando a realização de determinados experimentos: Alessandro, Antonio, Caio, Gabriel, Jade, Lucas, Narciso, Renato e Washington.

à TecSinapse, em especial ao Augusto, por todo o apoio, paciência e conhecimento transmitido durante o período em que trabalhamos juntos.

a todos meus professores, em especial aos da UFSCar, pelos inúmeros ensinamentos ao longo dos anos. Agradeço também à instituição, pelos recursos concedidos e por possibilitar um aprendizado gratuito e de qualidade.

à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), pelo apoio financeiro concedido para a realização desse estudo (Código de Financiamento 88882.426978/2019-01).

a todos que de alguma maneira contribuíram para a realização desse trabalho.

e, finalmente, a Deus, que sempre esteve comigo e me deu forças para continuar a cada dia.

"O prazer mais nobre é a alegria do entendimento"
(Leonardo da Vinci)

Resumo

Atualmente, com a crescente disponibilidade de conteúdo digital, e facilidade de acesso a uma elevada quantidade de dados, está cada vez mais difícil encontrar informações relevantes. Deste problema, nasceram os sistemas de recomendação, que analisam o comportamento de usuários para emitir recomendações personalizadas de itens. Atualmente, boa parte das empresas possui alguma forma de recomendação de conteúdo em seus canais ou serviços. Ainda assim, muitos problemas da área não estão devidamente solucionados, como a alta dimensionalidade e esparsidade que o modelo de representação comumente adotado possui. Métodos foram propostos para combater esses problemas, representando itens e usuários através de vetores densos em um espaço de dimensionalidade reduzida. Uma das técnicas mais recentes na literatura é o uso de métodos de representação vetorial distribuída, gerada por meio de redes neurais artificiais. Muitos dos últimos avanços feitos na área apresentaram resultados promissores quando comparados à outros métodos já consolidados, mas a grande maioria das propostas sugere o uso de arquiteturas neurais complexas e o emprego de dados de conteúdo, que muitas vezes podem não estar disponíveis para o cenário da aplicação. Essa dissertação estende os estudos na área de representação vetorial distribuída para recomendação. Através de uma revisão ampla da literatura, é proposto um novo modelo para geração da representação, computacionalmente eficiente e que demanda apenas *feedback* implícito dos usuários para ser treinado. Os resultados obtidos em avaliações extrínsecas e intrínsecas, indicam que o modelo é promissor para cenários onde há a necessidade de aplicação de métodos computacionalmente eficientes, atingindo resultados competitivos com métodos estado-da-arte que demandam maior poder computacional.

Palavras-chaves: Sistemas de Recomendação. Filtragem Colaborativa. Representação Vetorial Distribuída. Modelos de *Embeddings*.

Abstract

Today, with the constant growth in the number of available digital content and ease of access to a huge amount of data, it is increasingly difficult to find relevant information. From this problem, recommender systems have emerged, analyzing user's behavior to issue personalized item recommendations. Currently, most companies have some form of content recommendation on their channels or services. Even so, many problems in the area are not properly solved, such as the high dimensionality and sparsity that the commonly adopted representation model has. Methods have been proposed to solve these problems, representing items and users as dense vectors in a space of reduced dimensionality. One of the most recent techniques in the literature is the use of embeddings-based models, i. e., distributed vector representations generated through artificial neural networks. Many of the latest advances in the area have shown promising results when compared to other already consolidated methods, but the vast majority of proposals suggest the execution of complex neural architectures or the use of content data, which often may not be available int the scenario of the application. In this dissertation, we extend the studies in the field of embeddings-based recommender systems. Through a comprehensive literature review, a new model for generating distributed representation is proposed, with the key points of being computationally efficient and requiring only implicit feedback from users to be trained. The results obtained in extrinsic and intrinsic evaluations, indicate that the model is promising for scenarios where there is need for computationally efficient models, achieving competitive results with state-of-the-art methods that demand greater computational power.

Key-words: Recommender Systems. Collaborative Filtering. Distributed Vector Representation. Embedding-based Models.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ALS	*Alternating Least Squares*
BPR	*Bayesian Personalized Ranking*
CB-DCG	*Content-Based Discounted Cumulative Gain*
CB-IDCG	*Content-Based Ideal Discounted Cumulative Gain*
CB-NDCG	*Content-Based Normalized Discounted Cumulative Gain*
CBOW	*Continuous Bag of Words*
CKBE	*Collaborative Knowledge Base Embedding*
DCG	*Discounted Cumulative Gain*
F1	F-Medida
FM	*Factorization Machines*
GRU	*Gated Recurrent Unit*
IDCG	*Ideal Discounted Cumulative Gain*
ITM2V	Item2Vec
KNN	*k-Nearest Neighbors*
LDA	*Latent Drichlet Allocation*
LSA	*Latent Semantic Analysis*
LSI	*Latent Semantic Indexing*
LSTM	*Long Short-term Memory*
MAE	*Mean Absolute Error*
MLP	*Multilayer Perceptron*
INT2V	Interact2Vec
métrica@N	resultado da métrica para recomendação top-N
NDCG	*Normalized Discounted Cumulative Gain*

PLN	Processamento de Linguagem Natural
PREC	Precisão
REC	Revocação
RMSE	*Root Mean Squared Error*
RVD	Representação Vetorial Distribuída
SGD	*Stochastic Gradient Descent*
SOM	*Self-organizing Maps*
SVD	*Singular Value Decomposition*
SVM	*Support Vector Machines*
USR2V	User2Vec

Lista de símbolos

U	conjunto de usuários
I	conjunto de itens ou *feedback* implícito
R	conjunto ou matriz de interações entre usuários e itens
u ou v	usuário específico
i ou j	item específico
r_{ui}	valor da interação entre o usuário u e o item i
$\|.\|$	cardinalidade de conjunto
$\in$	pertence
$\notin$	não pertence
$\|$	tal que
$R_{\min}$	menor avaliação possível no *feedback* explícito
$R_{\max}$	maior avaliação possível no *feedback* explícito
R_u	conjunto de avaliações do usuário u
R_i	conjunto de avaliações do item i
I_u	conjunto de itens consumidos pelo usuário u
U_i	conjunto de usuários que consumiram o item i
$\vec{r_u}$	vetor das interações do usuário u de tamanho $\|I\|$
$\vec{r_{u_i}}$	valor da interação entre o usuário u e o item i
$\vec{r_i}$	vetor das interações do item i de tamanho $\|U\|$
$\vec{r_{i_u}}$	valor da interação entre o item i e o usuário u
$\bar{r_u}$	média entre as avaliações feitas pelo usuário u
$\bar{r_i}$	média entre as avaliações recebidas pelo item i
$\hat{r}_u i$	previsão da nota que o usuário u daria para o item i

N	quantidade de itens que compõe uma recomendação
$s(u, N)$	função para recomendar N itens para o usuário u
K	quantidade de vizinhos para KNN e demais métodos
$\mathcal{M}$	matriz qualquer para o SVD
$\mathcal{U}$	valores singulares das colunas de uma matriz para o SVD
Σ	matriz identidade para o SVD ou somatória
$\mathcal{V}$	valores singulares das linhas de uma matriz para o SVD
$\sum$	somatória
$\mathcal{J}$	camada de entrada do Item2Vec, User2Vec ou Interact2Vec
$\mathcal{H}$	camada intermediária do Item2Vec, User2Vec ou Interact2Vec
$\mathcal{O}$	camada de saída do Item2Vec, User2Vec ou Interact2Vec
M	tamanho de $\mathcal{H}$ e dimensão das *embeddings*
$\mathcal{W}$	primeira matriz de pesos do Item2Vec e matriz de pesos para usuários do Interact2Vec
$\mathcal{W}^u$	matriz de pesos para usuários do User2Vec
$\mathcal{W}^i$	matriz de pesos para itens do User2Vec
$\mathcal{W}'$	segunda matriz de pesos do Item2Vec e User2Vec, e matriz de pesos para itens do Interact2Vec
σ	função *softmax*
ϕ	função *sigmoid*
e	constante de Euler
T	transposta de uma matriz
$P(.\|.)$	probabilidade condicional
$z(i)$	quantidade de interações do item i, similar a $\|U_i\|$
ρ	parâmetro para controle da probabilidade de subamostragem de itens frequentes
G	conjunto de itens negativos para amostragem negativa

γ	parâmetro para controle da distribuição de probabilidade de seleção das amostras negativas
$\|\|.\|\|$	norma vetorial
λ	parâmetro para regularização da representação vetorial
$sim(x, y)$	similaridade entre dois objetos
$\forall$	para todo
β	peso da similaridade usuário-item para método de recomendação por similaridade ponderada
μ	peso da similaridade item-item para método de recomendação por similaridade ponderada
L	quantidade de itens recomendados pelos métodos que compõe o comitê de métodos
m	método de recomendação integrante do comitê
S	conjunto de recomendações para o comitê de métodos
$f(i, u, m)$	função para verificar se o item i está presente na recomendação $s_m(u, L)$
w_m	peso relacionado à capacidade preditiva de m para o comitê de métodos
w_r	peso relacionado à posição r do item no *ranking* para o comitê de métodos
V	conjunto de itens consumidos pelo usuário e similares ao item-alvo para o KNN
$\subset$	está contido em
$O(.)$	notação O-grande para complexidade computacional
C	número de iterações para redes neurais
$\gg$	muito maior que
$\approx$	aproximadamente igual a
E	esparsidade da base de dados ou *feedback* explícito
$\cap$	intersecção
$g(n, u)$	função responsável por verificar se o item da recomendação foi consumido pelo usuário

n	posição da recomendação top-N ordenada
Z	valor mínimo entre N e $\|I_u\|$
X_f^2	resultado estatístico para o teste não-paramétrico de Friedman
D	quantidade de conjuntos de dados observados no experimento
k	quantidade de métodos comparados no experimento
Q	média das posições nos *rankings* dos resultados por base de dados
CD	diferença crítica para o teste *post-hoc* de Bonferroni-Dunn
α	intervalo de confiança ou taxa de aprendizado
q_α	valor crítico para o intervalo de confiança α
t	distribuição de Student
f	número de fatores latentes descobertos para métodos de fatoração de matriz
∞	infinito
R^*	conjunto de pares usuário-item alvos para algoritmos de previsão de notas
$\vec{f}$	vetor de atributos do item
a	atributo específico dos itens
$\mathcal{C}$	matriz de similaridade entre itens de acordo com seus atributos
$\mathcal{E}$	matriz de similaridade entre itens de acordo com a representação vetorial gerada
N_x	vizinhanças para os itens construídas através da matriz de similaridade x

Sumário

Introdução

Com o desenvolvimento da tecnologia e aumento na quantidade de dados criados e compartilhados diariamente, uma quantidade cada vez maior de informação torna-se acessível. Este cenário trouxe diversos avanços para o mundo moderno. Qualquer usuário pode consumir milhares de filmes, músicas, produtos, e diversos outros itens. Porém, com esta enorme quantidade de informações, é evidente que apenas uma parcela seja de interesse de cada usuário.

Surgiu então, na década de 90, o conceito de Sistemas de Recomendação (ADOMAVICIUS; TUZHILIN, 2005): métodos que recebem como entrada um conjunto de informações relacionadas a usuários e itens e têm como objetivo encontrar um subconjunto de itens que seja do interesse de cada usuário. Com o passar dos anos e amadurecimento dos sistemas de recomendação, o seu conceito e utilização passou a se fazer cada vez mais presente (BOBADILLA et al., 2013). Atualmente, é difícil navegar pela Internet sem se deparar com alguma ferramenta que possua este propósito.

A popularização dos sistemas de recomendação fez com que empresas percebessem cada vez mais a importância de garantir uma melhor experiência para o cliente através de recomendações personalizadas. Em diversas empresas, a maioria dos itens acessados ou consumidos diariamente por seus usuários são frutos de seus algoritmos de recomendação (LINDEN; SMTITH; YORK, 2003; GOMEZ-URIBE; HUNT, 2015; GRBOVIC; CHENG, 2018).

Das diferentes classificações existentes, a Filtragem Colaborativa, que será abordada nesta dissertação, é a abordagem mais usual para sistemas de recomendação (JANNACH et al., 2010). Ela consiste no consumo de um conjunto de interações entre usuários e itens (GOLDBERG et al., 1992), como compras realizadas ou notas atribuídas, para gerar a recomendação. O objetivo da filtragem colaborativa é prever relações para pares usuário-item que ainda não possuem entrada no sistema, descobrindo assim o que recomendar para os usuários. Diversas técnicas foram propostas para o cenário de filtragem colaborativa (SCHAFER et al., 2007), das quais destacam-se os algoritmos de similaridade e de fatoração de matriz (JANNACH et al., 2010).

Os algoritmos de similaridade têm como principal pressuposto o fato que usuários com gostos semelhantes para certos itens terão gostos semelhantes para os demais. Dessa forma, a recomendação é feita baseada em usuários ou itens similares (HERLOCKER; KONSTAN; RIEDL, 2002; SARWAR et al., 2001).

Em cenários reais, o volume de dados aumenta a cada instante. Estima-se que o número de assinantes da Netflix tenha aumentado em aproximadamente 600% do terceiro

semestre de 2011 para o primeiro de 2019, alcançando 158 milhões de usuários[1]. A Amazon, apenas em seu portal estadunidense, conta com aproximadamente 120 milhões de itens à venda[2]. Adicionalmente, esses números crescem de forma muito mais acelerada do que o número de interações entre usuários e itens. Consequentemente, dados que alimentam técnicas de filtragem colaborativa tornam-se cada vez mais esparsos e representados em espaços de elevada dimensionalidade.

Um grande volume de dados com alta esparsidade faz com que métodos baseados em vizinhança demandem muitos recursos computacionais, como tempo de processamento e memória (SARWAR et al., 2002). Na tentativa de sanar este problema, foram desenvolvidas técnicas baseadas em fatoração de matriz, de forma que uma matriz de relação usuário-item pudesse ser decomposta em duas de dimensionalidade menor (KOREN; BELL; VOLINSKY, 2009). Por apresentarem bons resultados, estas técnicas figuram hoje como uma das mais tradicionais da área (BOBADILLA et al., 2013). Entretanto, reduzir a dimensionalidade através de fatoração de matrizes esparsas podem comprometer a qualidade das recomendações geradas pelo sistema (SARWAR et al., 2000b).

Estudos recentes vêm buscando alterar este cenário, representando itens de forma vetorial distribuída, reduzindo a dimensionalidade da representação, mas preservando o significado das relações (BARKAN; KOENIGSTEIN, 2016; ZHAO et al., 2019). Estes estudos são baseados em técnicas consolidadas na área de processamento de linguagem natural, capazes de mapear palavras para vetores sem perder relações semânticas (BENGIO et al., 2003; MIKOLOV; YIH; ZWEIG, 2013). Entretanto, uma grande parcela das propostas recentes no assunto tem como estratégia o uso de redes neurais complexas (ZARZOUR; AL-SHARIF; JARARWEH, 2019; SIDANA et al., 2021) ou demandam dados adicionais às interações entre usuários e itens (VASILE; SMIRNOVA; CONNEAU, 2016; ZHANG et al., 2016), dificultando sua aplicação em diversos cenários reais.

Esta dissertação estende os estudos na aplicação de modelos de representação vetorial distribuída em sistemas de recomendação de filtragem colaborativa. Através de uma abordagem de fácil aplicação e com baixa demanda computacional, foi desenvolvido um método capaz de gerar, de forma conjunta e eficiente, representações vetoriais de itens e usuários em um mesmo espaço vetorial de baixa dimensionalidade. Sendo um modelo de representação vetorial distribuída, o método apresentado busca mitigar os problemas de alta esparsidade e elevada dimensionalidade encontrados na área de recomendação. Sua principal característica é sua eficiência, podendo ser treinado em cenários de recursos computacionais escassos, sem comprometer a qualidade das recomendações geradas.

[1] Quantidade de usuários da Netflix extraídos de: <https://www.statista.com/statistics/250934/quarterly-number-of-netflix-streaming-subscribers-worldwide/>. Acesso em 10/03/2021.

[2] Quantidade de produtos da Amazon extraídos de <https://www.scrapehero.com/number-of-products-on-amazon-april-2019/>. Acesso em 10/03/2021.

Objetivos e contribuições

O principal objetivo desta dissertação é propôr um novo método de rede neural artificial para geração de representações vetoriais distribuídas de itens e usuários no contexto de sistemas de recomendação, que seja computacionalmente eficiente e não demande dados adicionais durante o treinamento, podendo assim ser aplicado na maioria dos cenários reais.

Em resumo, as principais contribuições deste trabalho são:

- Proposta e avaliação de um novo método para geração de representações vetoriais distribuídas de itens e usuários, com as vantagens de ser computacionalmente eficiente e de fácil aplicação;
- Sugestão da aplicação de técnicas de avaliação intrínseca comumente utilizadas na área de processamento de linguagem natural para a área de sistemas de recomendação;
- Apresentação de uma nova métrica capaz de medir a qualidade intrínseca de uma representação vetorial.

Organização

Essa dissertação está estruturada da seguinte forma:

1. No Capítulo 1, é apresentada uma revisão abrangente da área de sistemas de recomendação, abordando desde a sua origem até o cenário atual e explicando conceitos básicos e métodos populares na literatura. É também definida a notação matemática utilizada ao longo da dissertação, e são apresentadas as principais limitações existentes na área;
2. No Capítulo 2, é apresentada uma revisão bibliográfica minuciosa sobre a recente área de representações vetoriais distribuídas para recomendação, geradas por meio de redes neurais artificiais. Adicionalmente, é explicado o funcionamento de dois métodos pioneiros para geração de representação vetorial através de redes neurais no contexto de sistemas de recomendação;
3. No Capítulo 3, é detalhado o Interact2Vec, método de geração de representações vetoriais de itens e usuários proposto nesta dissertação. É explicado sua arquitetura e funcionamento, assim como estratégias utilizadas no aprendizado e métodos de recomendação possíveis de serem aplicados;

4. No Capítulo 4, é reportado o protocolo experimental executado para avaliar o desempenho do método proposto em comparação com outros da literatura, além dos resultados obtidos em tarefas de avaliação extrínseca e intrínseca;

5. Finalmente, são apresentadas as conclusões, análises finais e direcionamentos para pesquisas futuras.

1 Sistemas de recomendação

Com o desenvolvimento da tecnologia e popularização da internet, tornou-se muito mais fácil produzir, disponibilizar e ter acesso aos mais variados tipos de conteúdo. Este cenário trouxe grandes avanços para o mundo moderno, permitindo que usuários possam consumir uma vasta quantidade de filmes, músicas, produtos e diversos outros itens. Porém, com um enorme volume de informações disponíveis, é esperado que apenas uma pequena parcela seja relevante para o usuário.

Surgiu então o conceito de Sistemas de Recomendação: ferramentas que recebem como entrada um conjunto de informações relacionadas a usuários e itens e têm como objetivo encontrar um subconjunto de itens que seja do interesse de cada usuário (ADOMAVICIUS; TUZHILIN, 2005). Com o passar dos anos e aprimoramento dos sistemas de recomendação, o seu conceito e utilização passou a se fazer cada vez mais presente (BOBADILLA et al., 2013). De fato, é cada vez mais difícil navegar pela Internet sem se deparar com alguma ferramenta com este propósito.

Por definição, um sistema de recomendação pode ser descrito como um ambiente contendo um conjunto de usuários e um conjunto de itens que se relacionam entre si. O objetivo de um sistema de recomendação é aprender informações relevantes sobre os usuários, itens e interações usuário-item realizadas, para prever novas interações significativas. Desta forma, o sistema torna-se capaz de recomendar, para um dado usuário alvo, um conjunto de itens relevantes (ADOMAVICIUS; TUZHILIN, 2005).

Neste capítulo, são abordados conceitos básicos da área de sistemas de recomendação. É apresentada uma breve história da origem do assunto como área de pesquisa e ferramenta comercial, seguido de termos e notações matemáticas que serão utilizadas ao longo desta dissertação. Em seguida, é oferecida uma revisão bibliográfica sobre sistemas de recomendação baseados em filtragem colaborativa, seus métodos e cenário atual. Por fim, é disponibilizada uma breve análise sobre os desafios da área.

1.1 Origens e cenário atual

O início dos Sistemas de Recomendação pode ser atribuído a diferentes áreas de pesquisa que propuseram as primeiras abordagens de personalização baseadas em informações de uma entidade alvo (ADOMAVICIUS; TUZHILIN, 2005). Estudos relacionados a gestão empresarial para tratativa de clientes (MURTHI; SARKAR, 2003) e marketing com base em conhecimento prévio do consumidor (LILIEN; KOTLER; MOORTHY, 1992) são exemplos de aplicações pioneiras em diferentes áreas com conceitos que hoje são comuns

dentro dos sistemas de recomendação.

O primeiro sistema de recomendação surgiu na área da ciência cognitiva, com o Grundy (RICH, 1979), que gerava recomendações personalizadas mapeando usuários em grupos baseados em esteriótipos. Para realizar a recomendação, o sistema consumia um conjunto de palavras que descrevessem o usuário, e buscava classificá-lo dentro de grupos de interesse, recomendando assim algum produto que julgasse relevante. Muitas das ideias do Grundy foram aplicadas em sistemas de recomendação futuros, especialmente em sistemas baseados em conteúdo ou informações demográficas (PAZZANI, 1999).

A área firmou-se como objeto de pesquisa na década de 90 (PARK et al., 2012), com trabalhos precursores que solucionavam problemas dentro de um contexto de recomendação utilizando a opinião explícita de usuários, ou seja, avaliações objetivas sobre itens recomendados.

Proposto por Goldberg et al. (1992), o Tapestry foi o primeiro sistema a utilizar *feedback* explícito de usuários dentro de um cenário de recomendação. A ferramenta consistia em um filtro de correio eletrônico que utilizava anotações de usuários com gostos semelhantes para decidir quais e-mails exibir para o usuário alvo. O trabalho foi importante por dar início a uma série de propostas seguintes que também se utilizavam de opiniões passadas de usuários para gerar recomendações, além de ter definido diversos termos e conceitos comumente utilizados na área nos dias atuais (PAZZANI, 1999).

Nos anos seguintes, foram apresentados os primeiros sistemas que se utilizavam das mesmas técnicas do Tapestry para automatizar o processo de recomendação (ADOMAVICIUS; TUZHILIN, 2005). Tais sistemas possuíam funcionamentos bastante similares, diferenciando-se principalmente na área de aplicação. Como exemplos, tem-se o GroupLens, sistema proposto por Resnick et al. (1994) capaz de filtrar artigos jornalísticos, o Ringo, recomendador de álbuns e artistas musicais proposto por Shardanand e Maes (1995), e o recomendador de vídeos e filmes de Hill et al. (1995).

Com o passar dos anos, a área presenciou diversos avanços e teve um grande crescimento na literatura (PARK et al., 2012). Propostas inovadoras foram apresentadas, gerando diferentes abordagens e classificações para sistemas de recomendação, assunto abordado na seção seguinte.

A área ganhou bastante popularidade e destaque após a competição Netflix Prize, organizada pela empresa de *streaming* de vídeo Netflix Inc. em 2006 (BENNETT; LANNING, 2007). O evento foi possivelmente o principal responsável pelo elevado crescimento no número de propostas presentes na literatura e ampliação do interesse comercial e acadêmico pelo assunto (HALLINAN; STRIPHAS, 2016), .

Apresentando uma base de dados de dimensões nunca antes vistas na área, atingindo projeção internacional e oferecendo um prêmio financeiramente atraente, a competição

fez com que a pesquisa na área alcançasse avanços extremamente relevantes em um curto período de tempo, popularizando métodos estado-da-arte (KOREN, 2008), gerando estratégias de recomendação complexas e inovadoras (KOREN, 2009a; TÖSCHER; JAHRER, 2009), e atraindo novos pesquisadores e grupos de pesquisa (JACKSON, 2017).

Em paralelo com a popularização da área de pesquisa dentro do ambiente acadêmico, sistemas de recomendação foram também amplamente incorporados em cenários comerciais: diversas empresas de tecnologia perceberam a importância de garantir uma experiência única para o cliente através de recomendações personalizadas, como é o caso de Airbnb (GRBOVIC; CHENG, 2018), Amazon (LINDEN; SMTITH; YORK, 2003), Netflix (GOMEZ-URIBE; HUNT, 2015) e YouTube (COVINGTON; ADAMS; SARGIN, 2016). Em muitas delas, a maioria dos itens acessados ou consumidos diariamente por seus usuários são frutos de algoritmos de recomendação (ARORA, 2016), e investimentos bilionários são feitos anualmente em pesquisas da área (MCALONE, 2016).

Atualmente, sistemas de recomendação são uma área de pesquisa em constante desenvolvimento (LU et al., 2015). Adaptando-se ao cenário tecnológico atual e buscando enfrentar desafios presentes na área, muitas tendências vem ganhando espaço dentro da literatura em recomendação, como o uso de múltiplos critérios de avaliação para compôr uma nota entre usuário e item (LIU; MEHANDJIEV; XU, 2011), a combinação de recomendadores com redes sociais através de sistemas de recomendação sociais (GUY, 2015), a adaptação de sistemas de recomendação para um cenário de Internet das Coisas (FOROUZANDEH et al., 2017), o uso de redes neurais e *deep learning* para recomendação (ZHANG et al., 2019b) e a aplicação de modelos neurais para representação vetorial distribuída (RVD) (GRBOVIC et al., 2015).

Das tendências citadas, este trabalho possui enfoque na última, tendo como objetivo estender os estudos na área de recomendação baseada em RVD. O tópico é abordado em detalhes no Capítulo 2.

1.2 Categorias de sistemas de recomendação

Com o amadurecimento da área, diferentes estratégias foram propostas para construir recomendações personalizadas. De maneira mais ampla, as principais diferenças entre as diversas propostas de sistemas de recomendação relacionam-se ao tipo de informação consumida pelo sistema e a maneira como ele aprende sobre seus usuários ou itens. Essa divisão fez com que sistemas de recomendação pudessem ser categorizados de diferentes formas (BOBADILLA et al., 2013).

Uma das maneiras mais aceitas para classificação de sistemas de recomendação divide-os de acordo com as abordagens utilizadas durante o aprendizado e particularidades dos dados consumidos pelos modelos (CANDILLIER; MEYER; BOULLé, 2007). Dessa

forma, são utilizadas quatro categorias principais: filtragem baseada em conteúdo, filtragem demográfica, filtragem colaborativa e filtragem híbrida. Cada uma das categorias é brevemente explicada nas seções seguintes.

1.2.1 Filtragem baseada em conteúdo

Sistemas de recomendação baseados em conteúdo têm como característica principal utilizar os metadados dos itens combinados com o histórico de interações do usuário para gerar a recomendação (LOPS; GEMMIS; SEMERARO, 2011).

Tais sistemas assumem a hipótese que usuários terão uma tendência de consumir itens que possuem características em comum com itens que consumiram anteriormente (THORAT; GOUDAR; BARVE, 2015). Estas características podem se manifestar de diferentes formatos, desde termos textuais que descrevem o item (por exemplo, título e sinopse para filmes), até dados estruturados (por exemplo, artista e gênero musical para músicas) (PAZZANI; BILLSUS, 2007).

De forma geral, a filtragem baseada em conteúdo possui uma forte relação com a área de recuperação da informação. Muitas técnicas provenientes da área, como etapas de pré-processamento e formas de representação (BAEZA-YATES; RIBEIRO-NETO, 1999), são utilizadas nas etapas iniciais dos sistemas baseados em conteúdo (LOPS; GEMMIS; SEMERARO, 2011). Após a extração do conteúdo dos itens, torna-se possível a construção de um perfil de interesses do usuário, gerando a recomendação com base em suas interações (METEREN; SOMEREN, 2000).

Sistemas desta categoria possuem vantagens em relação a problemas de *cold-start*, cenário no qual determinado item não possui interações prévias no sistema antes do momento da recomendação (explicado de forma mais detalhada na Seção 1.7). Por este motivo, a filtragem baseada em conteúdo é uma abordagem comumente utilizada em aplicações práticas (PAZZANI; BILLSUS, 2007). Ainda assim, ela possui certas desvantagens, como a forte dependência à existência de metadados e a ausência de serendipidade nas recomendações geradas (LOPS; GEMMIS; SEMERARO, 2011).

1.2.2 Filtragem demográfica

Da mesma forma que a filtragem baseada em conteúdo utiliza informações descritivas relacionadas aos itens para compôr a recomendação, sistemas de recomendação de filtragem demográfica utilizam características e dados relacionados ao perfil do usuário para aprender sobre o mesmo (BOBADILLA et al., 2013). A hipótese assumida por estes tipos de sistema é que usuários que pertencem a um mesmo grupo demográfico apresentarão uma tendência de compartilhar os mesmos interesses (PAZZANI, 1999).

Abordado pioneiramente por Krulwich (1997), modelos de filtragem demográfica se

utilizam de dados pessoais e informações relacionadas ao estilo de vida dos usuários, por exemplo, bens financeiros, estrutura familiar, rotina e interesses, para construir um perfil descritivo do usuário alvo. Em seguida, através de métodos de agrupamento ou medidas de semelhanças, é possível associar usuários que compartilham os mesmos gostos em relação aos itens do sistema, gerando a recomendação (LU et al., 2015).

De maneira similar a filtragem baseada em conteúdo, métodos demográficos podem ser utilizados para se extrair informação relevante de usuários que ainda não interagiram com o sistema, permitindo uma primeira recomendação mais assertiva. Ainda assim, é uma abordagem altamente dependente destes dados, e que pode não possuir poder suficiente para generalizar os diferentes tipos de clientes, gerando recomendações pouco variadas (PAZZANI, 1999).

1.2.3 Filtragem colaborativa

A filtragem colaborativa é uma das abordagens mais comuns em sistemas de recomendação (BOBADILLA et al., 2013). Por não depender de dados adicionais, apenas do histórico de interações, ela se torna bastante fácil de ser reproduzida em diversas aplicações reais, aumentando sua popularização (JANNACH et al., 2010).

Algoritmos de filtragem colaborativa assumem a hipótese que usuários com um padrão de interações similar no passado tenderão a manter o mesmo comportamento no futuro (GOLDBERG et al., 1992). Assim, a recomendação para um usuário-alvo pode ser construída observando suas interações passadas e comparando-as com o histórico de interações dos demais usuários.

A abordagem presenciou diversos avanços desde o surgimento da área até os dias atuais, estando presente em diversos sistemas de recomendação modernos utilizados por grandes empresas de tecnologia (LINDEN; SMTITH; YORK, 2003; GOMEZ-URIBE; HUNT, 2015; GRBOVIC; CHENG, 2018). Mesmo com sua alta popularidade, a recomendação baseada em filtragem colaborativa enfrenta diversos desafios, como o problema do *cold-start* e a elevada esparsidade e dimensionalidade de seus dados, explicados na Seção 1.7.

Este trabalho, assim como o método proposto (Capítulo 3), dedica-se ao estudo da recomendação através de filtragem colaborativa. Desta forma, uma revisão mais completa da área é apresentada na Seção 1.6.

1.2.4 Filtragem híbrida

Combinando as demais abordagens apresentadas anteriormente, a filtragem híbrida é uma técnica que ganhou destaque dentro da área devido aos bons resultados alcançados e alto poder de adaptação (BOBADILLA et al., 2013).

As primeiras propostas combinando diferentes técnicas de recomendação surgiram no final da década de 90, poucos anos após a consolidação da área de sistemas de recomendação como objeto de pesquisa (BALABANOVIĆ; SHOHAM, 1997; BASU; HIRSH; COHEN, 1998; CLAYPOOL et al., 1999). Métodos híbridos mostraram-se uma estratégia interessante por utilizar características de uma abordagem para contornar as deficiências apresentadas pelas demais abordagens (SCHEIN et al., 2002).

Muitos avanços foram feitos na área de recomendação híbrida, com diversas estratégias diferentes de combinação de técnicas sendo propostas (BURKE, 2002). Ainda que muitos desafios da área continuem sendo um impedimento para a aplicação de determinados métodos em um cenário híbrido, como a ausência de metadados ou um histórico de interações pouco completos, é bastante comum o uso da abordagem em sistemas de recomendação comerciais modernos, comprovando seu poder para a recomendação (GOMEZ-URIBE; HUNT, 2015).

1.3 *Feedback* explícito e implícito

Independente da abordagem de aprendizado adotada pelo sistema de recomendação, é necessário que haja um conjunto passado de interações que descrevam as relações dos usuários com os itens do sistema (BURKE; FELFERNIG; GÖKER, 2011). As interações passadas entre um usuário e um item são tradicionalmente chamadas de *feedbacks*, e podem ser explícitas ou implícitas (RICCI; ROKACH; SHAPIRA, 2011).

O *feedback* explícito é caracterizado como uma avaliação feita por um usuário sobre um item de forma explícita, ou seja, ativamente informando o sistema seu grau de satisfação com determinado item, o que é comumente feito através de uma nota dentro de um intervalo (por exemplo, 1–5) (HERLOCKER et al., 2004).

O *feedback* implícito, ao contrário, é capturado pelo sistema conforme o usuário interage com o mesmo. Em nenhum momento o usuário informa diretamente ao sistema se determinado item é de seu agrado, mas relações usuário-item são construídas conforme as interações ocorrem. O ato de ouvir uma música, assistir um filme, comprar um produto e ler um artigo são exemplos de possíveis *feedbacks* implícitos (HERLOCKER et al., 2004).

Em oposição ao *feedback* explícito, o *feedback* implícito não carrega a informação sobre o grau de satisfação ou descontentamento de um usuário em relação a um item. Por ser caracterizado como um conjunto de interações que simplesmente ocorreram, não é possível inferir que um usuário aprecia um item pelo fato de ter interagido com ele. Da mesma forma, não é possível afirmar que um usuário não tem interesse nos itens com os quais não interagiu: é possível que ele apenas desconheça-os (LIN; KAMAR; HORVITZ, 2014). Ainda assim, essa é uma suposição normalmente realizada em métodos de recomendação com *feedback* implícito (HU; KOREN; VOLINSKY, 2008; CREMONESI;

KOREN; TURRIN, 2010), por facilitar a aplicação de determinados modelos sem grandes prejuízos para o resultado final. Outra estratégia utilizada em domínios específicos é a definição de diferentes tipos de interação, de forma que mais informação seja extraída das relações implícitas entre o usuário e o sistema (ZHAO et al., 2019).

1.4 Objetivos da recomendação

O objetivo principal de um sistema de recomendação é aprender sobre as relações entre usuários e itens para recomendar itens para um usuário alvo. Tradicionalmente, esta recomendação pode ocorrer de duas maneiras diferentes: através de uma previsão de nota ou no formato de um ranqueamento top-N (SHANI; GUNAWARDANA, 2011).

Na previsão de nota, o sistema recebe como entrada um par usuário–item e tem como objetivo prever qual avaliação o usuário em questão geraria para o item alvo. Por se tratar de um problema de regressão, recomendadores deste tipo são normalmente avaliados através de medidas de erro (HERLOCKER et al., 2004).

A previsão de nota, também chamada de recomendação baseada em *feedback* explícito, tornou-se uma das formas mais comuns de se avaliar sistemas de recomendação por ter sido o esquema avaliativo utilizado no Netflix Prize (BENNETT; LANNING, 2007). Desta forma, muitos métodos estado-da-arte na área foram propostos e avaliados apenas dentro deste cenário (SARWAR et al., 2001; KOREN; BELL; VOLINSKY, 2009; RENDLE, 2010).

Sistemas destinados para recomendação, ou ranqueamento, top-N possuem como objetivo receber um usuário alvo e gerar uma sequência ordenada de N itens que serão de seu interesse (MUSTAFA; FROMMHOLZ, 2015). Sua avaliação é feita através de medidas de acurácia, que avaliam a qualidade da seleção e o ordenamento gerado (SHANI; GUNAWARDANA, 2011). Embora menos popular no início da literatura, a recomendação top-N é atualmente o formato mais adotado por sistemas de recomendação comerciais, por permitir uma experiência mais confortável para o usuário do sistema (DESHPANDE; KARYPIS, 2004).

Embora ambas as estratégias de recomendação tenham como objetivo prever relações entre usuários e itens, não é possível afirmar que modelos eficazes na previsão de nota manterão sua qualidade no ranqueamento top-N. Em um cenário avaliativo tradicional, um sistema de previsão de notas é igualmente beneficiado por acertar interações positivas e negativas, enquanto que um sistema de ranqueamento top-N deve se especializar na seleção apenas de interações positivas, ou seja, quando o item agrada ao usuário (STECK, 2013). Desta forma, ambos objetivos de recomendação e cenários avaliativos podem ser considerados na hora de construir um sistema de recomendação, por possibilitarem visões complementares em relação ao seu funcionamento.

1.5 Notação matemática

Com o objetivo de formalizar a definição de um sistema de recomendação e facilitar o entendimento dos métodos e protocolos utilizados neste trabalho, será utilizada a notação proposta por Ekstrand e Konstan (2019). Assim, um sistema de recomendação é composto pelos relacionamentos entre duas entidades principais: um conjunto U de $|U|$ usuários e um conjunto I de $|I|$ itens.

Os usuários do sistema interagem com os itens, compondo um histórico de interações. As interações conhecidas pelo sistema podem ser representadas por meio de uma matriz esparsa R de dimensões $|U| \times |I|$. Para facilitar o entendimento, R também pode ser interpretado como um conjunto de valores r_{ui} que representam a interação entre um usuário $u \mid u \in U$ e um item $i \mid i \in I$. Assim, se o usuário u interagiu previamente com o item i, tem-se que $r_{ui} \in R$; do contrário, $r_{ui} \notin R$.

Em um cenário baseado em *feedback* explícito, R armazena as avaliações explícitas informadas pelos usuários. Assim, r_{ui} contém a nota que o usuário u atribuiu para o item i, que poderá variar entre dois valores R_{min} e R_{max}, sendo $R_{\text{min}} < R_{\text{max}}$. O conjunto de avaliações informadas por um usuário u são representadas por R_u, e o conjunto de avaliações recebidas pelo item i são representadas por R_i. Para um cenário de *feedback* implícito, r_{ui} não terá conteúdo, ou seja, o sistema não conhecerá nenhuma informação explícita sobre as interações além de sua própria ocorrência.

O subconjunto de itens consumidos por um usuário u é definido como $I_u = \{i \in I : r_{ui} \in R_u\}$. Já o subconjunto de usuários que consumiram o item i é definido como $U_i = \{u \in U : r_{ui} \in R_i\}$. Consequentemente, a quantidade de itens consumidos por um usuário pode ser representado por $|I_u|$, e a quantidade de usuários que interagiram com um item por $|U_i|$.

Em muitos métodos, é necessário realizar operações aritméticas ou vetoriais com as interações de um usuário ou item. Para tal, as interações de um usuário poderão ser representadas vetorialmente como um vetor $\vec{r_u}$ de tamanho $|I|$. Em um cenário explícito, cada posição $\vec{r_{u_i}}$ possui a avaliação do usuário u para o item i. Já em um cenário implícito, $\vec{r_u}$ é um vetor binário, ou seja, $\vec{r_{u_i}} = 1$ se $r_{ui} \in R$ e 0 caso contrário. De maneira semelhante, as interações de um item são representadas como um vetor $\vec{r_i}$. O valor médio entre as avaliações informadas por um usuário é representado como $\bar{r_u}$, e o valor médio entre as avaliações recebidas por um item como $\bar{r_i}$.

Um sistema de recomendação pode ter dois objetivos principais, de acordo com o tipo desejado da recomendação. Para recomendação explícita, o sistema deve prever a avaliação que um usuário u daria para um item i, o que será representado como $\hat{r}_{ui}$. Na recomendação implícita, também conhecida como top-N, o sistema deve, para um dado usuário u, montar um ordenamento de itens a serem recomendados. Essa forma de

recomendação será representada como uma função $s(u, N)$ que retornará para o usuário alvo u uma sequência ordenada de N itens.

Demais notações e operações exclusivas dos métodos de recomendação serão definidas em suas seções específicas.

1.6 Filtragem colaborativa e modelos de representação

Sistemas de recomendação de filtragem colaborativa surgiram junto com o início da área de sistemas de recomendação, com o trabalho de Goldberg et al. (1992), e hoje figuram como uma das abordagens mais comuns e bem sucedidas na área (JANNACH et al., 2010). Ainda assim, é uma área com diversos desafios a serem superados e que ainda possui espaço para muitos avanços (SU; KHOSHGOFTAAR, 2009).

Técnicas de filtragem colaborativa podem ser classificados de duas maneiras: baseadas em memória e baseadas em modelo (BOBADILLA et al., 2013). Cada uma destas classificações é discutida adiante.

1.6.1 Filtragem colaborativa baseada em memória

Métodos de recomendação baseados em memória caracterizam-se por utilizar todas as interações do sistema a qualquer momento que é requerida uma recomendação (SU; KHOSHGOFTAAR, 2009). Em outras palavras, sistemas desta categoria não ajustam um modelo inteligente para ser reutilizado nas predições. Dessa forma, não é necessária nenhuma etapa de aprendizado antes da recomendação, mas a cada recomendação gerada é necessário um novo processamento computacional por parte do método.

A grande maioria dos métodos baseados em memória é composta por algoritmos de filtragem colaborativa baseados em vizinhança, isto é, métodos que se utilizam de medidas de distância e similaridade para compôr vizinhanças de itens ou usuários, as quais são utilizadas para gerar a recomendação (HERLOCKER; KONSTAN; RIEDL, 2002).

1.6.1.1 Métodos baseados em vizinhança

Métodos baseados em vizinhanças figuraram como a estratégia mais popular para sistemas de recomendação no início da área (HERLOCKER; KONSTAN; RIEDL, 2002). Sistemas pioneiros, como o GroupLens (RESNICK et al., 1994) e o Ringo (SHARDANAND; MAES, 1995), funcionavam através da execução do método K-vizinhos mais próximos (KNN) sobre a base de interações. Em ambos, os usuários do sistema eram representados como vetores de notas utilizadas para avaliar os itens e a recomendação era feita por meio da construção de uma vizinhança: para um dado par usuário–item, eram encontrados usuários semelhantes ao usuário alvo através de medidas de similaridade entre os vetores,

e a nota para o item era estimada através de uma média ponderada entre as notas dadas pelos outros usuários.

Os primeiros sistemas de recomendação baseados em vizinhança diferenciavam-se apenas em relação a algumas decisões pequenas sobre o funcionamento do modelo, como a medida de similaridade adotada e a maneira que os vizinhos eram selecionados (BREESE; HECKERMAN; KADIE, 1998).

Nos anos seguintes, estratégias complementares ao KNN surgiram na literatura, como o uso de agrupamento para delimitar a vizinhança (UNGAR; FOSTER, 1998), a redução de esparsidade através de avaliações geradas por agentes automáticos baseados em conteúdo (SARWAR et al., 1998) e o uso de modelos probabilísticos em conjunto com as métricas de similaridade (PENNOCK et al., 2000).

Até recentemente, uma larga parcela dos métodos de vizinhança eram baseados em usuário, ou seja, para estimar a avaliação de um usuário para um item, eram consultadas as notas atribuídas por outros usuários similares (SARWAR et al., 2000a). Entretanto, uma nova abordagem baseada em item começou a ganhar espaço na literatura, tendo diversas vantagens e alcançando resultados superiores em vários domínios, como evidenciado por Karypis (2001).

Pesquisas seguintes reforçaram a qualidade da recomendação orientada a item em cenários de *feedback* explícito, tanto para previsão de nota (SARWAR et al., 2001) quanto para ranqueamento top-N (DESHPANDE; KARYPIS, 2004). Em problemas de *feedback* implícito, Linden, Smtith e York (2003) mostraram que aplicar o KNN sobre uma matriz de itens que co-ocorrem pode trazer resultados interessantes, usando como caso de uso o sistema de recomendação da varejista eletrônica Amazon.

Devido a popularização da tecnologia, sistemas de recomendação passaram a ter que lidar com uma quantidade significativa de usuários e itens, aumentando consideravelmente a dimensionalidade das representações vetoriais usadas por métodos de vizinhança e comprometendo o desempenho computacional. Tal cenário, aliado ao surgimento e notável superioridade de métodos baseados em fatoração de matriz (KOREN; BELL; VOLINSKY, 2009), diminuíram o interesse científico relacionado a algoritmos de vizinhança para recomendação (PARK et al., 2012). Ainda assim, muitas variações para o KNN e outros métodos de vizinhança foram propostos em pesquisas recentes.

Algumas das novas abordagens para recomendação através de vizinhança são: o uso de informações de conteúdo para preencher avaliações não existentes e enriquecer os dados consumidos pelos métodos (MELVILLE; MOONEY; NAGARAJAN, 2002; DEGEMMIS; LOPS; SEMERARO, 2007; SU; KHOSHGOFTAAR; GREINER, 2008); a alteração da maneira que a similaridade é calculada, adicionando pesos de acordo com características da aplicação (JIN; CHAI; SI, 2004; LEMIRE, 2005; WANG; LIAO; ZHANG, 2013); a

decomposição da matriz de similaridade em dimensões menores, para redução de esforço computacional (DESROSIERS; KARYPIS, 2011); mudanças na forma de realizar a regressão e calcular a nota prevista após encontrar a vizinhança (LI; ZHANG, 2018); e agrupamento de usuários e itens usando informações de domínio ou metadados (SUBRAMANIYASWAMY; LOGESH, 2017; AHUJA; SOLANKI; NAYYAR, 2019).

1.6.2 Filtragem colaborativa baseada em modelo

Ao contrário dos métodos baseados em memória, a filtragem colaborativa baseada em modelo tem como principal característica a criação de um modelo inteligente ajustado sobre as interações passadas, que é reutilizado no momento de gerar a recomendação. Como vantagens, a recomendação baseada em modelo normalmente apresenta resultados melhores, além de ser capaz de gerar recomendações com maior agilidade. Em contrapartida, os modelos inteligentes apresentam uma tendência a ficar desatualizados conforme o tempo passa, sendo necessário uma constante etapa de retreinamento para manter a qualidade das recomendações (SU; KHOSHGOFTAAR, 2009).

Métodos baseados em modelos podem ser classificados de três maneiras diferentes, de acordo com a estratégia usada no aprendizado: métodos baseados em fatoração de matriz, métodos baseados em grafos e métodos baseados em redes neurais.

1.6.2.1 Métodos baseados em fatoração de matriz

Para aliviar o problema da alta dimensionalidade da matriz de interações devido ao elevado volume de usuários e itens, foram propostos métodos capazes de reduzir a dimensionalidade da representação. Inspirados em técnicas clássicas de redução de dimensionalidade, especialmente na Decomposição de Valores Singulares (SVD), esses métodos têm como objetivo decompor a matriz de interação em matrizes menores, computando fatores latentes que descrevem os itens e usuários em espaços de baixa dimensionalidade. Por este motivo, recebem o nome de métodos de fatoração de matriz (KOREN; BELL, 2011).

O SVD é uma técnica conhecida dentro da ciência de dados e inteligência artificial para reduzir a dimensionalidade de representações matriciais. Matematicamente, o objetivo da técnica é decompor uma determinada matriz $\mathcal{M}$ em três matrizes $\mathcal{U} \times \Sigma \times \mathcal{V}$, cujo produto reconstruirá $\mathcal{M}$. Após a decomposição, $\mathcal{U}$ armazenará os valores singulares das colunas, $\mathcal{V}$ os valores singulares das linhas e Σ será uma matriz identidade utilizada apenas para possibilitar a reconstrução (KOREN; BELL, 2011).

Embora já tenha sido aplicada em diversas áreas de estudo (DEERWESTER et al., 1990), o SVD apresenta barreiras para sua aplicação direta na filtragem colaborativa: por definição, a técnica é aplicável apenas sobre matrizes densas, o que não é o caso

da filtragem colaborativa, que possui matrizes de interações esparsas. Por este motivo, as primeiras propostas referentes ao uso de decomposição matricial para recomendação sugeriam preencher avaliações faltantes com valores padrões ou heurísticas (KIM; YUM, 2005; SARWAR et al., 2001; CREMONESI; KOREN; TURRIN, 2010). Essa técnica apresentou resultados promissores, mas demandava alto poder de processamento de acordo com o tamanho do conjunto de dados.

Métodos de fatoração de matriz começaram a ganhar destaque na área após propostas de adaptação do SVD para considerar apenas as interações existentes, utilizando técnicas de otimização como o ALS (*Alternating Least Squares*) (BELL; KOREN, 2007). Entretanto, foi com os estudos de Funk (2016) para o Netflix Prize que a abordagem se consolidou dentro da área da recomendação. O pesquisador mostrou como otimizar o problema através de um otimizador baseado em gradiente descente estocástico, estratégia usada em diversos outros trabalhos subsequentes e responsável por tornar métodos de fatoração matricial o estado-da-arte na área (KOREN; BELL; VOLINSKY, 2009; PATEREK, 2007).

Também para o Netflix Prize, foi proposto o uso de *feedback* implícito para melhorar a fatoração, mudando a função objetivo do SVD (SALAKHUTDINOV; MNIH, 2007). O novo modelo foi chamado de SVD++ e apresentou resultados significativamente melhores que seu antecessor (KOREN, 2008). Estendendo ainda mais o modelo, Koren (2009b) observou que era possível incorporar o momento da interação no aprendizado, melhorando a qualidade da recomendação. Daí surgiu o timeSVD++.

Anos depois, Rendle (2010) percebeu que muitos dos avanços do SVD ocorreram devido a incorporação de outras informações no modelo, além das interações em si. Diante disso, foi proposto o método de FM (*Factorization Machines*), uma junção de modelos de decomposição matricial e SVM (*Support Vector Machines*), capazes de incorporar qualquer dado adicional disponível no momento da fatoração. Experimentos com o FM atingiram resultados comparáveis ao timeSVD++, sendo mais genérico e de fácil aplicação em diferentes domínios (RENDLE, 2012).

Paralelamente ao amadurecimento dos métodos mencionados, Hu, Koren e Volinsky (2008) notaram a ausência de algoritmos para recomendação utilizando bases de dados apenas de *feedback* implícito. Para preencher esta lacuna, desenvolveram uma variação implícita de decomposição matricial, utilizando um conceito de taxa de confiança nas interações para preencher a matriz. No ano seguinte, Rendle et al. (2009) também apresentaram um método de fatoração baseado *feedback* implícito, o BPR (*Bayesian Personalized Ranking*). O método, que obteve resultados promissores, tinha como destaque avaliar o ordenamento da recomendação como função-objetivo, utilizando técnicas probabilísticas para isso.

Muitos estudos foram conduzidos sobre métodos baseados em fatoração de matriz,

sendo até hoje uma das abordagens mais estudadas e aplicadas (BOBADILLA et al., 2013). Algumas das propostas realizadas nos últimos anos dizem respeito a: uso de informações sociais para melhorar a fatoração (MA et al., 2011); aplicação de técnicas temporais (como, por exemplo, janelas de esquecimento) sobre as bases de dados (MATUSZYK; SPILIOPOULOU, 2014; MATUSZYK et al., 2015); aceleração do método para funcionamento em grandes bases de dados, através de novas estratégias de otimização (LUO et al., 2014), paralelismo (YU et al., 2014) e uso de hardwares gráficos (LI et al., 2018); enriquecimento da fatoração de matriz com técnicas probabilísticas (HERNANDO; BOBADILLA; ORTEGA, 2016; BOBADILLA et al., 2017); e uso de redes neurais profundas em parceria com a fatoração (LARA-CABRERA; GONZÁLEZ-PRIETO; ORTEGA, 2020).

1.6.2.2 Métodos baseados em grafos

Métodos baseados em grafos tem como característica principal a mudança na forma como usuários, itens e interações são representados dentro do sistema. Neste cenário, a representação é feita através de um grafo, onde usuários e itens normalmente são os vértices, e interações são arestas, podendo ser ponderadas ou não (no caso de sistemas de *feedback* explícito) (DESROSIERS; KARYPIS, 2011).

O primeiro sistema de recomendação baseado em grafos foi proposto por Aggarwal et al. (1999). Nele, foi utilizado um grafo para conectar usuários entre si, de acordo com suas preferencias. A recomendação então era feita através da descoberta do caminho mais curto entre o usuário-alvo e demais usuários. Anos depois, Mirza, Keller e Ramakrishnan (2003) apresentaram uma abordagem semelhante, com a diferença que usuários eram conectados seguindo relações sociais.

Os usos pioneiros de grafos para recomendação utilizavam apenas relações entre usuários. Huang, Chung e Chen (2004) apresentaram um dos primeiros grafos para recomendação que conectasse usuários com itens. A recomendação era então realizada através de diferentes técnicas de teoria dos grafos e recuperação da informação. Uma segunda estratégia para a recomendação, proposta por Huang, Chen e Zeng (2004), consistia em contabilizar o número de caminhos entre um usuário e um item para compôr a recomendação.

Outras técnicas propostas para recomendadores baseados em grafo foram: a aplicação de métodos baseados em passeio aleatório (*random-walk*) para percorrer o grafo e gerar a recomendação (GORI; PUCCI, 2007; FOUSS et al., 2007; LEE et al., 2012); a combinação de métodos de grafos com modelos neurais (YING et al., 2018; ZHANG et al., 2019a); e o uso de grafos para construir representações vetoriais de usuários e itens (CRICHTON et al., 2018; VERMA et al., 2019; CENIKJ; GIEVSKA, 2020).

1.6.2.3 Métodos baseados em redes neurais

Com a popularização geral da área de redes neurais, o uso de modelos neurais para recomendação tem crescido expressivamente nos últimos anos. Com diversas arquiteturas sendo propostas, é esperado que a área apresente grandes avanços no futuro (ZHANG et al., 2019b).

A primeira aplicação de uma rede neural para recomendação foi apresentada por Roh, Oh e Han (2003), que adaptaram um algoritmo de mapas de Kohonen (SOM, ou *Self-organizing Maps*) para o contexto de filtragem colaborativa. Anos mais tarde, (CHRISTAKOU; STAFYLOPATIS, 2005) combinaram dados de filtragem baseada em conteúdo e filtragem colaborativa para alimentar uma rede MLP (*Multilayer Perceptron*) responsável por gerar a recomendação.

Outras propostas surgiram na literatura com o passar dos anos, como o uso de redes neurais para mineração de dados em cenários de recomendação (AMATRIAIN; PUJOL, 2015); a aplicação de *auto-encoders* (SEDHAIN et al., 2015); a combinação de redes neurais profundas com métodos de decomposição matricial, como o FM (ZHANG; DU; WANG, 2016); e o uso de atributos ou metadados de itens para alimentar as redes neurais (SHAN et al., 2016).

Métodos recentes de redes neurais artificiais para filtragem colaborativa têm como objetivo aprender RVDs de baixa dimensionalidade (GRBOVIC et al., 2015; BARKAN; KOENIGSTEIN, 2016). Após serem geradas, as RVDs podem ser utilizadas por métodos baseados em memória – sem demandar um elevado custo computacional – ou como entrada para outros métodos baseados em modelo. Uma revisão sobre recomendação baseada na criação de RVDs por redes neurais é oferecida no Capítulo 2.

1.7 Desafios

Embora tenha presenciado muitos avanços desde seu surgimento, a área de sistemas de recomendação ainda possui diversos desafios a serem superados (RICCI; ROKACH; SHAPIRA, 2011). Um dos obstáculos mais tradicionais dentro da área é o problema da partida fria, ou *cold-start*. Ele ocorre especialmente em cenários de filtragem colaborativa, por se caracterizar como a necessidade de se gerar recomendações para um novo usuário do sistema, ou recomendar itens recém-cadastrados e nunca consumidos (KHSURO; ALI; ULLAH, 2016). Devido ao fato de a grande maioria dos algoritmos de recomendação ser altamente dependente de um histórico de interações para construir uma recomendação, se um produto ou usuário não encontra-se nos dados utilizados para treinamento do sistema, torna-se impossível incluí-los na recomendação (SCHEIN et al., 2002).

Muitas soluções foram propostas para o problema do *cold-start* (LAM et al.,

2008). De forma geral, tais soluções estão relacionadas ao uso de diferentes abordagens de recomendação, como baseada em conteúdo, demográfica ou híbrida. Através do consumo de dados adicionais referentes aos itens e usuários, é possível construir um perfil de interesse mesmo se não há interações passadas. Ainda assim, tais soluções são inaplicáveis em diferentes cenários, fazendo com que o *cold-start* seja até hoje um dos mais clássicos desafios da recomendação (KHSURO; ALI; ULLAH, 2016).

Outros dois desafios bastante presentes na área estão relacionados a maneira como os dados de interações são comumente representados. São eles os problemas de alta esparsidade e elevada dimensionalidade, e podem ocorrer tanto em sistemas de filtragem colaborativa como em sistemas baseados em conteúdo ou híbridos (DRACHSLER; HUMMEL; KOPER, 2008).

No formato mais usual, usuários e itens de um sistema de recomendação são representados de forma vetorial, cujas dimensões estão diretamente relacionadas ao número de itens no catálogo e quantidade de usuários do sistema (SCHAFER et al., 2007). Entretanto, com o desenvolvimento da tecnologia, muitos sistemas de recomendação modernos possuem uma enorme quantidade de itens e usuários, fazendo com que a representação tradicional alcance espaços vetoriais de alta dimensionalidade (LINDEN; SMTITH; YORK, 2003). Esse fator, além de prejudicar a qualidade de diversos algoritmos de recomendação, gera problemas de escalabilidade e alta demanda computacional (KHSURO; ALI; ULLAH, 2016).

Embora o tamanho do catálogo de itens e base de usuários de sistemas de recomendação cresça constantemente, o número de interações usuário-item não apresenta o mesmo crescimento, gerando um problema de alta esparsidade (HUANG; CHEN; ZENG, 2004). Dados esparsos prejudicam o desempenho de diversos algoritmos de recomendação tradicionais, sendo assim uma das maiores barreiras da área (SU; KHOSHGOFTAAR, 2009). Muitas estratégias foram propostas para minimizar o problema, mas, com o desenvolvimento da tecnologia e a popularização da internet, a tendência é que este problema se agrave cada vez mais (BILLSUS; PAZZANI, 1998; SARWAR et al., 2002).

Apesar dos problemas supracitados serem os mais tradicionais da área, sistemas de recomendação modernos ainda enfrentam outros obstáculos: questões relacionadas a segurança da informação, como a privacidade dos usuários; problemas infraestruturais, como disponibilidade e latência dos sistemas; confiabilidade no conteúdo, como a existência interações e avaliações falsas; e problemas comportamentais, como usuários com gostos bastante incomuns ou sistemas altamente especializados e enviesados (KHSURO; ALI; ULLAH, 2016).

As propostas apresentadas neste trabalho têm como objetivo principal estudar formas de representação de dimensionalidade reduzida para sistemas de recomendação. Assim, o principal objetivo desta pesquisa é, através de técnicas computacionalmente

eficientes, combater os problemas de alta dimensionalidade e esparsidade.

1.8 Considerações finais

Este capítulo apresentou os principais conceitos referentes a área de sistemas de recomendação, contextualizando o assunto historicamente e definindo termos e notações de uso comum dentro da área.

Primeiramente, descreveu-se o início do assunto como objeto de pesquisa, apontando trabalhos pioneiros e importantes dentro da literatura, seguido de eventos relevantes para a área. Também foi apresentada a importância que sistemas de recomendação possuem em aplicações atuais, tanto no âmbito acadêmico quanto comercial, e possíveis tendências.

Foram introduzidos conceitos e termos importantes para o entendimento do assunto. As diferentes categorias e classificações de algoritmos de recomendação foram apresentadas, dividindo-os entre filtragem baseada em conteúdo, demográfica, colaborativa e híbrida. Em seguida, foram apresentados os tipos de dados consumidos pelos sistemas em relação a maneira que são obtidos, podendo representar interações explícitas ou implícitas do usuário com os itens do sistema.

Em seguida, foi definida a notação matemática comumente adotada na literatura, que será utilizada neste trabalho, seguido de uma explicação completa sobre métodos para filtragem colaborativa, abordagem de recomendação do método proposto nesta dissertação.

Finalmente, foram elencados diversos desafios e problemas na área, que servem como possíveis tópicos de pesquisa até os dias atuais, tais como os problemas de *cold-start* e alta dimensionalidade e esparsidade nas formas de representação adotadas.

Uma das estratégias recentes dentro da área para aliviar o problema de elevada dimensionalidade e esparsidade em sistemas de recomendação modernos é o uso de modelos de RVD. Baseados em redes neurais e capazes de aprender representações densas de baixa dimensionalidade para usuários e itens, modelos de RVD vêm ganhando destaque na literatura, apresentando resultados competitivos com métodos estado-da-arte e sendo aplicados com sucesso em sistemas comerciais. Com o objetivo de expandir os estudos na área, o capítulo seguinte oferece uma revisão bibliográfica de modelos de RVD para sistemas de recomendação. Serão apresentadas diversas propostas presentes na literatura, destacando características relevantes sobre os modelos existentes e analisando pontos interessantes para guiar tópicos de pesquisa dentro da área.

2 Modelos neurais de representação vetorial distribuída

Antes mesmo dos primeiros trabalhos sobre sistemas de recomendação, a área de Processamento de Linguagem Natural (PLN) já enfrentava desafios semelhantes aos apresentados na Seção 1.7. Por muitos anos, o formato mais utilizado para representar documentos em problemas de PLN consistia em vetores de mesmo tamanho do vocabulário conhecido pela aplicação, com conteúdo variando de acordo com as palavras presentes no documento (GOLDBERG, 2016). Com o crescimento no tamanho das bases de dados textuais e da quantidade de informação produzida, tais vetores tornaram-se extremamente esparsos e com alta dimensionalidade (LOCHTER et al., 2018), gerando problemas computacionais análogos aos enfrentados por algoritmos de recomendação (GOLDBERG; HIRST, 2017).

Muitas técnicas propostas na área de PLN para resolver tais barreiras foram bastante similares ao que, anos depois, seria utilizado para combater o mesmo problema na recomendação. Um exemplo é um dos modelos mais influentes na área, o LSA (*Latent Semantic Analysis*), ou LSI (*Latent Semantic Indexing*), que consiste na aplicação da técnica de SVD sobre uma matriz de ocorrência de termos em documentos (DEERWESTER et al., 1990).

Após o LSI, a técnica de representar documentos ou palavras em vetores de dimensionalidade reduzida ganhou bastante destaque na área (SCHÜTZE, 1993; BLEI; NG; JORDAN, 2003). Entre as estratégias propostas com esta finalidade, o uso de redes neurais se mostrou como um dos mais promissores, superando métodos estado-da-arte como o LSA e o LDA (*Latent Drichlet Allocation*) (MIKOLOV; YIH; ZWEIG, 2013).

A aplicação de modelos neurais para representar palavras e documentos surgiu com o trabalho pioneiro de Bengio et al. (2003). Nele, é proposta uma rede neural simples para aprendizado conjunto de um modelo probabilístico e de *embeddings* de palavras. O termo "*embeddings*", cunhado por Bengio et al. (2003), é definido como uma representação vetorial distribuída, densa e de baixa dimensionalidade, que carrega significado intrínseco em relação ao objeto que representa. Por serem capazes de representar diferentes objetos e entidades como vetores de dimensionalidade reduzida, muitos domínios de aplicação podem se aproveitar de RVDs. No contexto textual, o modelo é comumente utilizado para representar palavras, frases e documentos (GOLDBERG, 2016). Outras áreas, como a pesquisa de mercado (CHAMBERLAIN et al., 2017) e a biomedicina (CHEN et al., 2020), também agregaram o estudo de RVD para solução de seus problemas específicos. Como será mostrado nas seções seguintes, a área de sistemas de recomendação pode se beneficiar

com o uso de representações vetoriais distribuídas, aplicando-as para representar desde entidades básicas, como usuários e itens (GRBOVIC et al., 2015), a conceitos avançados, como preferências (VALCARCE et al., 2019) e tipos de interação (ZHAO et al., 2019).

A aplicação de RVD na área de PLN cresceu muito nos anos seguintes, se estendendo para diferentes domínios (MIKOLOV et al., 2013). O poder que essa abordagem traz para a resolução de diferentes tarefas foi primeiramente evidenciado por Collobert e Weston (2008). Entretanto, foi com a apresentação do Word2Vec, que a técnica ganhou bastante visibilidade. O Word2Vec é um conjunto de modelos neurais para geração de RVDs de palavras baseados em contexto, que se mostraram poderosos sem demandar grande esforço computacional (MIKOLOV et al., 2013; MIKOLOV et al., 2013). Atualmente, o uso de RVD é uma das principais estratégias para resolução de diversos problemas na área de PLN (CAMACHO-COLLADOS; PILEHVAR, 2018).

Acompanhando os avanços que modelos de RVD propiciaram para o Processamento de Linguagem Natural, pesquisas recentes vem propondo formas de se utilizar RVD dentro da área de sistemas de recomendação. A primeira abordagem sobre o assunto foi apresentada por Grbovic et al. (2015), que adaptaram dois modelos tradicionais de PLN para gerar RVDs de itens e usuários em um contexto de recomendação de produtos via correio eletrônico. Desde então, diversos outros trabalhos apresentaram novas arquiteturas ou técnicas para a recomendação através de RVDs. Tradicionalmente, os métodos de RVD para recomendação podem ser classificados em três grupos diferentes, de acordo com as representações geradas: métodos que geram RVDs apenas para itens, métodos que geram RVDs apenas para usuários, e métodos que geram RVDs para itens e usuários, de forma concomitante ou não, e utilizam ambos na recomendação.

Nas seções a seguir, é apresentado um levantamento da literatura de RVD em sistemas de recomendação, dividindo-a de acordo com as classificações recém-apresentadas.

2.1 Representação vetorial distribuída de itens

Sistemas de recomendação baseados em modelos de RVDs de itens têm como objetivo aprender vetores densos de baixa dimensionalidade para representar os itens do catálogo de um sistema de recomendação. Com a representação gerada, as RVDs podem ser utilizadas em métodos baseados em memória, como métodos de vizinhança (BARKAN; KOENIGSTEIN, 2016; VALCARCE et al., 2019), ou métodos baseados em modelo, como fatoração de matriz e redes neurais (GREENSTEIN-MESSICA; ROKACH; FRIEDMAN, 2017; ZARZOUR; AL-SHARIF; JARARWEH, 2019).

A primeira proposta de utilização de modelos neurais para geração de RVDs de itens foi apresentada por Grbovic et al. (2015). Nela, os autores introduziram o Prod2Vec, modelo neural capaz de aprender RVDs de produtos em um cenário de recomendação via correio

eletrônico. A arquitetura do modelo foi fortemente inspirada no Skip-gram (MIKOLOV et al., 2013), modelo linguístico pertencente ao Word2Vec para geração de RVDs de palavras.

Para possibilitar a adaptação para o cenário de recomendação, recibos fiscais de compra foram tratados como sentenças, e os itens contidos neles tratados como palavras. O Prod2Vec então ordenava os recibos cronologicamente, e construía um conceito de relação entre itens de acordo com conjuntos de itens comprados consecutivamente, através de uma estratégia de janela deslizante. A estratégia apresentou resultados com acurácia superior a de outros recomendadores heurísticos usados como *baseline*.

No ano seguinte, Barkan e Koenigstein (2016) apresentaram o Item2Vec, uma proposta similar ao Prod2Vec, novamente adaptando a arquitetura do Skip-gram para recomendação. A diferença, no entanto, foi na reformulação do conceito de contexto: quaisquer dois itens comprados por um mesmo usuário eram considerados como relacionados, utilizando assim uma janela de tamanho variável para capturar itens correlatos. Foi assumida a hipótese que interações passadas eram igualmente importantes, independente do momento em que ocorreram. O Item2Vec foi comparado com o SVD em diversas tarefas de avaliação intrínseca (isto é, tarefas responsáveis por medir a capacidade da representação em carregar informação de conteúdo sobre os itens), apresentando resultados superiores em todas elas. Devido ao fato do Item2Vec servir como uma das bases para o modelo de RVD proposto nesta dissertação, sua explicação detalhada é oferecida na Seção 2.4.1.

Assim como o Prod2Vec e o Item2Vec, muitos métodos propostos subsequentemente na literatura usaram o Word2Vec como base. A explicação por trás de tal fato se dá pela simplicidade do modelo e facilidade de aplicação. Como mostrado por Rudolph et al. (2016), as motivações por trás do Word2Vec podem ser generalizadas para diferentes domínios, incluindo a área de recomendação. No trabalho, os autores apresentaram o conceito de "Exponential Family Embeddings", reformulações das redes neurais de PLN que aprendem RVDs de forma probabilística usando distribuição de Poisson ou Bernoulli de acordo com o contexto de aplicação. Foi então conduzido um cenário de recomendação de produtos a fim de evidenciar a aplicabilidade do método.

Tanto o Prod2Vec quanto o Item2Vec são métodos puramente de filtragem colaborativa, gerando as representações distribuídas apenas através do histórico de interação dos usuários. Pesquisas futuras buscaram adaptar os métodos para um cenário de filtragem híbrida, consumindo também o conteúdo descritivo dos itens para enriquecimento das RVDs: o Meta-Prod2vec, proposto por Vasile, Smirnova e Conneau (2016), alterou a função de custo do Prod2Vec para também considerar metadados dos itens durante o processo de aprendizado das RVDs, alcançando resultados melhores que o modelo antecessor. Já o Attr2Vec, de FU et al. (2017), estendeu a arquitetura do Item2Vec para predizer não apenas itens relacionados com um determinado item alvo, mas também informação estruturada do item, como atributos e metadados. A RVD final de um item passou a ser uma composição

ponderada de RVDs de atributos aprendidas pela rede neural. A avaliação do método foi feita em comparação com o SVD, também apresentando resultados superiores em diferentes bases de dados.

A ideia de utilizar o conteúdo dos itens para enriquecimento das RVDs foi amplamente utilizada na literatura em diversas outras propostas. Zhang et al. (2016) apresentaram o CKBE (*Collaborative Knowledge Base Embedding*), um *framework* de recomendação capaz de aprender RVDs de filmes combinando quatro representações vetoriais diferentes, uma obtida sobre o *feedback* implícito dos usuários, e outras sobre dados estruturados (gênero, diretor e ator), dados textuais (sinopse) e dados visuais (pôster). Para aprender as representações, foram empregados diferentes métodos de *deep learning* baseados em *auto-encoders* bayesianos (WANG; WANG; YEUNG, 2015). Cheng et al. (2016) elaboraram um modelo neural composto por uma componente capaz de prever a avaliação por meio de uma regressão linear simples, e outra contendo diversas camadas e aprendendo representações vetoriais de itens dado suas informações de conteúdo. Por fim, Greenstein-Messica, Rokach e Friedman (2017) apresentaram dois modelos de geração de RVDs para sistemas baseados em sessão (LUDEWIG; JANNACH, 2018), ambos inspirados em arquiteturas neurais para PLN (MIKOLOV et al., 2013; PENNINGTON; SOCHER; MANNING, 2014). Inicialmente, o trabalho avaliou representações vetoriais de produtos geradas apenas com dados implícitos, de forma semelhante ao Item2Vec e Prod2Vec. Em uma segunda abordagem, metadados descritivos dos itens e um identificador baseado no preço de venda dos mesmos foram usados para melhorar o modelo. Para construir as recomendações finais, as RVDs serviram de entrada para uma rede neural recorrente GRU (*Gated Recurrent Unit*), também comumente aplicada em tarefas textuais (CHO et al., 2014).

Assim como nos trabalhos de Greenstein-Messica, Rokach e Friedman (2017) e Cheng et al. (2016), outras propostas de RVD também construíram a etapa de aprendizado e recomendação empregando redes neurais mais complexas. Boa parte destes trabalhos converteram o problema da recomendação para um problema de sequência e previsão do próximo passo: como o usuário interage com os itens do sistema em uma determinada ordem, é possível descobrir o próximo item a ser interagido observando a ordem de seu histórico. Este é o caso das pesquisas de Hidasi et al. (2016) e Tan, Xu e Liu (2016), que propuseram a aplicação de redes neurais recorrentes GRU para um cenário de recomendação baseada em sessão, utilizando como entrada do modelo o produto corrente sendo visualizado pelo usuário e o seu histórico de cliques no sistema, respectivamente. Baseando-se na mesma estratégia, mas com outra arquitetura de rede neural, Tang e Wang (2018) concatenaram RVDs de itens para representar uma sequência de interações como uma "imagem", permitindo assim o uso de uma rede neural convolucional para predição do item seguinte e ajuste das RVDs.

Ainda que novas propostas tenham sugerido o uso de arquiteturas neurais complexas,

grande parte da pesquisa recente ainda possui enfoque em modelos mais simples inspirados no Word2Vec: Grbovic e Cheng (2018) apresentaram um modelo inspirado no Skip-gram para geração de RVDs de locais para alocação, utilizado pela empresa de hospedagem comunitária Airbnb, que tinha como diferencial o uso de itens globais (o anúncio final selecionado pelo usuário) durante o processo de aprendizado. O Prefs2Vec, de Valcarce et al. (2019), é outro exemplo de arquitetura neural simples. O método é bastante semelhante ao Prod2Vec, porém possui uma arquitetura inspirada no CBOW (*Continuous Bag of Words*) (MIKOLOV et al., 2013), outro modelo que compõe o Word2Vec. O método foi aplicado em cenários de recomendação de diferentes domínios, apresentando resultados competitivos quando comparado a outros métodos tradicionais da área.

Outra abordagem comum para construção de representação vetorial é a execução de modelos de PLN diretamente sobre informações textuais dos itens. Siswanto, Tjong e Saputra (2018) representaram itens como textos descritivos para treinamento dos modelos Skip-gram e CBOW, obtendo uma representação distribuída final através da média entre as RVDs dos termos de cada item. De forma semelhante, Collins e Beel (2019) aplicaram o Doc2Vec (LE; MIKOLOV, 2014) – modelo de geração de RVDs de documentos ou sentenças completas – para montar a representação de artigos científicos dentro do Mr. DLib, sistema de recomendação de pesquisas acadêmicas (BEEL et al., 2011).

Por fim, uma estratégia recente dentro da área, é o uso de diferentes tipos de interação no momento de captura do *feedback* implícito, permitindo assim a construção de RVDs que representem não apenas determinado item, mas também a maneira como o usuário interagiu com ele, como proposto por Zhao et al. (2019).

2.2 Representação vetorial distribuída de usuários

Modelos neurais para aprendizado apenas de RVDs de usuários são menos comuns na literatura. Uma possível explicação para isso é o fato de muitas pesquisas já consolidadas mostrarem como sistemas de recomendação baseados em item tendem a apresentar resultados superiores aos baseados em usuário (SARWAR et al., 2001; LINDEN; SMTITH; YORK, 2003; DESHPANDE; KARYPIS, 2004). Na maioria das propostas que envolvem RVDs de usuários, RVDs de itens também são aprendidas e utilizadas na recomendação, como mostrado na seção seguinte.

A primeira proposta de geração de RVDs apenas de usuários é o modelo User2Vec, de Żołna e Romański (2017). Nele, as ações do usuário dentro do sistema são mapeadas para uma sequência que alimenta uma rede neural LSTM (*Long Short-term Memory*) (HOCHREITER; SCHMIDHUBER, 1997), responsável por aprender a representação vetorial.

Outra abordagem para geração de RVDs de usuário é o Prefs2Vec (VALCARCE et al., 2019), modelo baseado no CBOW que também pode ser utilizado para geração de

RVDs de itens. A escolha da entidade alvo que será representada vetorialmente é feita antes da etapa de aprendizado no modelo, sendo necessário alterar apenas os dados inseridos como entrada na rede.

2.3 Representação vetorial distribuída de itens e usuários

Muitos dos modelos propostos na área de RVD para recomendação tem como objetivo aprender representações distribuídas de itens e usuários. Embora nem todos os modelos aprendam as RVDs de forma concomitante, sendo necessário em alguns casos técnicas diferentes para aprender cada tipo de RVD, todos tem como característica em comum utilizar as duas representações de forma conjunta para compôr a recomendação.

A primeira abordagem capaz de gerar RVDs de usuários e itens foi a rede neural de Grbovic et al. (2015), também nomeada de User2Vec. Inspirada na versão Distributed Memory do modelo Paragraph Vector (LE; MIKOLOV, 2014), a rede é capaz de aprender RVDs de itens e usuários simultaneamente, fazendo uma composição das RVDs de um usuário e de determinados produtos de seu histórico de compra para alimentar o modelo. A representação foi comparada no mesmo cenário utilizado para o Prod2Vec e apresentou resultados superiores a todos os métodos avaliados. Entretanto, a representação mostrou uma queda de acurácia significativa conforme o sistema era mantido em produção sem atualizar as representações utilizadas. Uma explicação em maiores detalhes é oferecida na Seção 2.4.2, devido ao fato de o modelo ter servido como base para o método proposto nesta dissertação.

Outras propostas seguintes basearam-se nos modelos do Word2Vec para aprender as RVDs: Grbovic e Cheng (2018) apresentaram um modelo para a empresa de hospedagem Airbnb que agrupava usuários e anúncios através de heurísticas, aprendendo RVDs de tipos de usuários e tipos de locação por uma arquitetura semelhante ao Skip-gram. Yoon e Lee (2018) geraram RVDs de filmes concatenando informações referentes aos metadados dos itens para alimentar uma rede inspirada no Skip-gram, e RVDs de usuários através de uma média ponderada entre as representações dos filmes e as notas atribuídas pelo usuário.

Outras abordagens presentes na literatura adaptaram métodos de PLN que aprendem vetores de sentenças, como o Doc2Vec (LE; MIKOLOV, 2014), com o objetivo de aprender RVDs através das descrições dos itens. Behera et al. (2017) apresentaram um *framework* de recomendação para comércio eletrônico composto por três diferentes tipos de representações vetoriais: RVDs de itens, aprendidas pela aplicação do Doc2Vec sobre metadados dos itens, RVDs de usuários e RVDs de carrinhos de compra, ambas obtidas pela média das representações dos itens comprados pelo usuário. Já Hasanzadeh, Fakhrahmad e Taheri (2020) geraram RVDs textuais sobre *reviews* feitas por usuários sobre itens. as RVDs

de itens foram obtidas através da média entre as representações das *reviews* recebidas, e as RVDs de usuários pela média das representações das *reviews* escritas.

Por fim, outra estratégia presente na literatura é o uso de modelos neurais mais complexos, como o uso de *deep learning* para fatoração matricial proposto por He et al. (2017) e Xue et al. (2017), capaz de aprender RVDs de itens e usuários enquanto decompõe uma matriz de interações através de uma rede neural; o RecDNNing (ZARZOUR; AL-SHARIF; JARARWEH, 2019), arquitetura neural profunda capaz de aprender RVDs de usuários e itens concomitantemente enquanto soluciona uma tarefa de predição de avaliação explícita; o SeoulHouse2Vec (JUN et al., 2020), de funcionamento semelhante ao RecDNNing, mas aplicado em um problema do mercado imobiliário; e o NERVE (SIDANA et al., 2021), que recebe como entrada um usuário e um par de itens, e aprende as RVDs dos três objetos simultaneamente enquanto prevê a taxa de preferência do usuário para cada um dos itens.

2.4 Métodos clássicos de representação vetorial distribuída

O Item2Vec, de Barkan e Koenigstein (2016), e o User2Vec, de Grbovic et al. (2015), são métodos pioneiros no uso de RVD para sistemas de recomendação. Diferente de outras propostas publicadas posteriormente, os dois modelos possuem certas características em comum: o emprego de redes neurais rasas e o uso apenas de *feedback* implícito para aprendizado das RVDs. Desta forma, caracterizam-se como modelos de baixa complexidade computacional que não demandam dados adicionais como avaliações ou metadados de itens e usuários.

Inspirado nesses dois modelos, a principal proposta desta dissertação é o Interact2Vec. Capaz de aprender RVDs de itens e usuários simultaneamente, o modelo proposto tem como principal diferencial o baixo custo computacional e a necessidade apenas de *feedback* implícito.

A arquitetura neural do Interact2Vec é fortemente inspirada na arquitetura do Item2Vec, enquanto seu objetivo e função-custo assemelha-se ao User2Vec. Por este motivo, as seções a seguir detalham o funcionamento matemático por trás do Item2Vec e do User2Vec.

2.4.1 Item2Vec

O Item2Vec é uma rede neural artificial que tem como objetivo a geração de RVDs de itens. Ela é composta por três camadas: uma de entrada, uma intermediária e uma de saída, como mostrado na Figura 1.

A camada de entrada $\mathcal{J}$ possui tamanho $|I|$ (número de itens do catálogo), corres-

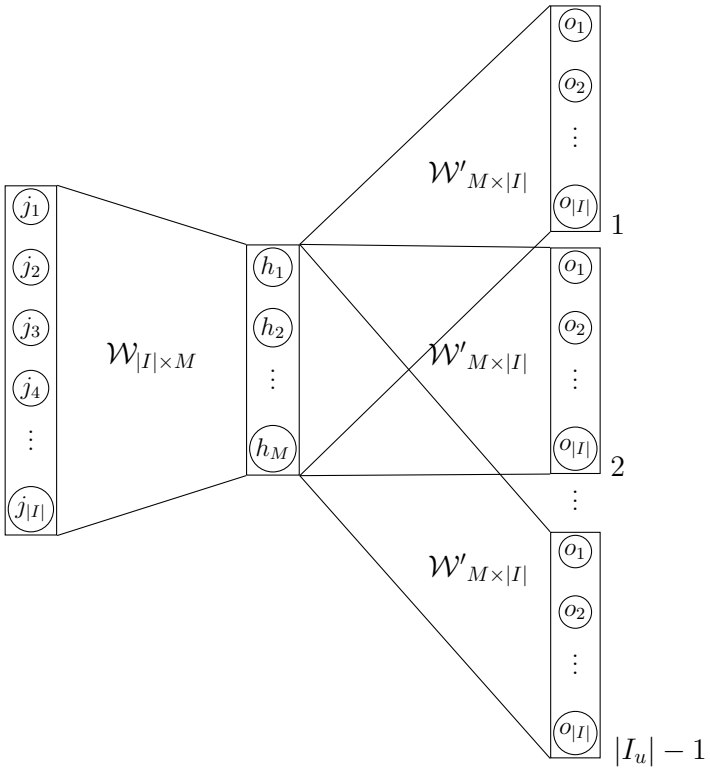

Figura 1 – Arquitetura do Item2Vec (adaptada de Lochter et al. (2018))

pondendo a um item i representado como um vetor *one-hot* (nesta forma de representação, todas as posições do vetor possuem valor 0, com exceção de uma, mapeada para o item representado e com conteúdo de valor 1). A camada intermediária $\mathcal{H}$ possui tamanho M, que corresponde a dimensionalidade desejada para a representação vetorial final de cada item. Entre a camada de entrada e a intermediária, há uma matriz de pesos $\mathcal{W}$, de tamanho $|I| \times M$, que serve como ativação para a camada inicial através da operação $\mathcal{H} = \mathcal{J} \times \mathcal{W}$. A camada de saída $\mathcal{O}$ é constituída de $|I_u| - 1$ vetores, cada um de tamanho $|I|$, conectados a camada intermediária por uma mesma matriz $\mathcal{W}'$, de tamanho $M \times |I|$, e uma função de ativação *softmax*. Assim, $\mathcal{O} = softmax(\mathcal{H}^T \times \mathcal{W}')$.

São percorridos os usuários u e, para cada fase de propagação da rede, um item $i \in I_u$ é informado como entrada. Na camada de saída $\mathcal{O}$, tem-se $|I_u| - 1$ vetores (a quantidade de itens consumidos pelo usuário removendo o item de entrada), cada um com tamanho $|I|$. Cada um destes vetores é associado a um item $j \in I_u \mid j \neq i$, também codificado na forma *one-hot*. Para cada vetor, a rede calcula o erro obtido e atualiza os pesos através de *backpropagation*, utilizando um otimizador SGD (*Stochastic Gradient Descent*).

O objetivo principal da rede é maximizar a função objetivo dada pela seguinte

equação:

$$\frac{1}{|U|} \sum_{u \in U} \sum_{i \in I_u} \sum_{\substack{j \in I_u \\ j \neq i}} log\ \sigma(i, j) \tag{2.1}$$

onde σ representa a função de *softmax*.

Ao final do aprendizado, as matrizes de pesos $\mathcal{W}$ e $\mathcal{W}'$ armazenarão representações vetoriais densas de cada item em um espaço reduzido, de dimensionalidade M. Tais representações podem ser utilizadas da seguinte forma para gerar recomendações: dado um item i, calcula-se a similaridade entre o vetor deste item, dado por $\mathcal{W}_i$, e os demais vetores de itens e recomenda-se aqueles cujas representações vetoriais são mais semelhantes a i. Em seus experimentos, Barkan e Koenigstein (2016) utilizaram de similaridade de cosseno sobre as RVDs e alcançaram resultados superiores ao SVD em tarefas de avaliação intrínseca.

2.4.2 User2Vec

Ao contrário do Item2Vec, que aprende apenas RVDs de itens através de relações item-item, o User2Vec aprende concomitantemente RVDs de usuários e de itens através de relações usuário-item. A arquitetura da rede é ilustrada na Figura 2.

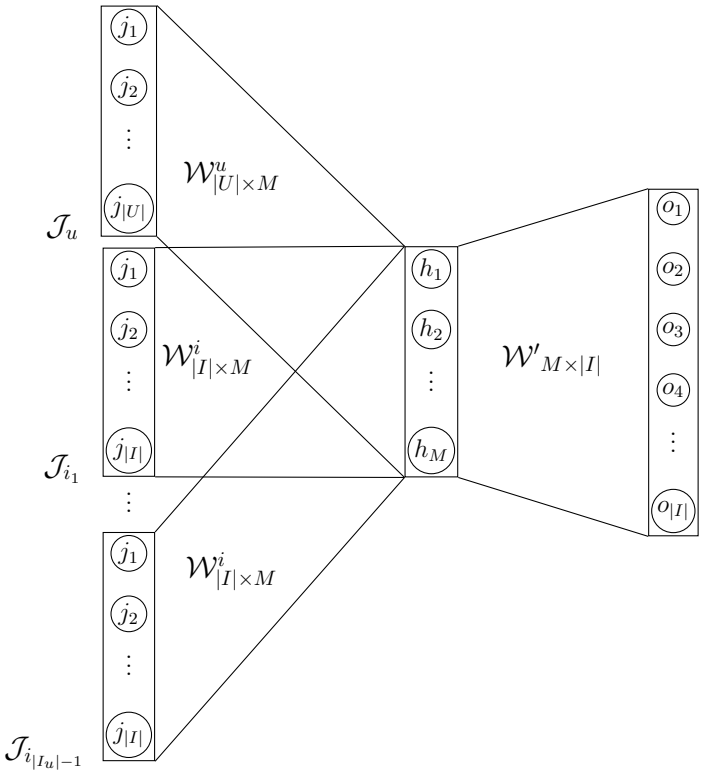

Figura 2 – Arquitetura do User2Vec (adaptada de Lochter et al. (2018))

O modelo possui uma camada de entrada, uma intermediária e uma de saída. A camada de entrada $\mathcal{J}$ corresponde a um vetor $\mathcal{J}_u$ em formato *one-hot* para representar um usuário u, e $|I_u| - 1$ vetores $\mathcal{J}_i$, também em *one-hot*, para representar os itens que u consumiu (com exceção de um, que é o item a ser previsto pela rede neural). Assim, o vetor de usuários possui tamanho $|U|$ (número de usuários), e os demais vetores tamanho $|I|$ (quantidade de itens no catálogo).

A camada intermediária $\mathcal{H}$ é conectada a camada $\mathcal{J}$ através de duas matrizes de pesos $\mathcal{W}^u$ e $\mathcal{W}^i$, a primeira de tamanho $|U| \times M$, para os vetores de usuários, e a segunda de tamanho $|I| \times M$, para os vetores de itens. Após a projeção, vetores de usuários e itens devem ser combinados através da média dos vetores. Assim $\mathcal{H} =$ média$(\mathcal{J}_u \times \mathcal{W}^u,\ \mathcal{J}_{i_1} \times \mathcal{W}^i,\ \mathcal{J}_{i_2} \times \mathcal{W}^i,\ ...,\ \mathcal{J}_{i_{|I_u|-1}} \times \mathcal{W}^i)$.

A camada de saída $\mathcal{O}$ contém um único vetor de tamanho $|I|$, correspondendo a um item $i \mid i \in I_u$, em formato *one-hot*, conectado a $\mathcal{H}$ por uma matriz $\mathcal{W}'$ e uma função *softmax*.

O modelo recebe como entrada um usuário u e todos os itens i consumidos por ele, com exceção de um, que deve ser previsto pela rede. Tem-se então a seguinte função objetivo, que deve ser maximizada:

$$\frac{1}{|U|} \sum_{u \in U} \sum_{i \in I_u} (log\ \sigma(u, i) + \sum_{\substack{j \in I_u \\ j \neq i}} log\ \sigma(i, j)) \tag{2.2}$$

onde σ representa a função de *softmax*.

Ao término do treinamento, $\mathcal{W}^u$ conterá as RVDs dos usuários e $\mathcal{W}^i$ as RVDs de itens. A recomendação é realizada através da similaridade entre a representação do usuário-alvo e dos itens do catálogo.

2.5 Considerações finais

Neste capítulo, foi apresentada uma revisão referente ao uso de RVD para sistemas de recomendação. Os modelos foram divididos de acordo com seu objetivo principal: gerar RVDs de itens, de usuários ou de ambos. Como visto, modelos neurais para geração de RVDs são um tópico relativamente novo na área, mas em constante crescimento e, em muitos casos, apresentando resultados promissores quando comparados a outros métodos tradicionais.

Ao analisar a história da área, percebe-se como a grande maioria das variações propostas após as primeiras abordagens do uso de RVDs tem como característica em comum o uso de informações de conteúdo ou metadados para enriquecimento da representação. Embora os resultados alcançados por estes modelos sejam competitivos, eles possuem a

desvantagem de não serem facilmente aplicados em qualquer cenário, visto que dependem de dados adicionais que poderão não estar disponíveis. Adicionalmente, muitas outras propostas usam modelos neurais apenas para a geração de RVDs de itens, não utilizando RVDs de usuários na recomendação ou construindo-as através de heurísticas simples, como média de outras RVDs. Propostas mais recentes utilizam modelos bastante complexos, de difícil aplicação e execução em cenários onde os recursos computacionais são escassos.

Após a revisão da literatura, foi apresentado o funcionamento e a arquitetura do Item2Vec e User2Vec, dois modelos de rede neural para RVD, pioneiros na área de recomendação. Ambos destacam-se por sua simplicidade e praticidade de aplicação, visto que são redes neurais rasas e que dependem apenas do histórico de *feedback* implícito, sem demandar metadados ou outras informações de conteúdo.

Tendo o Item2Vec e o User2Vec como inspiração, é apresentado no capítulo seguinte um novo modelo neural para aprendizado de RVD no contexto de sistemas de recomendação. Nomeado como Interact2Vec, o modelo possui uma arquitetura de rede similar ao Item2Vec, sendo computacionalmente mais eficiente, e objetivo semelhante ao do User2Vec, aprendendo simultaneamente RVDs de itens e usuários.

3 O método Interact2Vec

O Interact2Vec é um novo modelo neural proposto para geração de RVDs de usuários e itens de maneira simultânea, que busca atender as hipóteses presentes na Tabela 1. Dessa forma, o modelo é composto por uma arquitetura neural rasa, computacionalmente eficiente. Para o treinamento, o modelo proposto requer apenas a matriz de interações usuário-item, sendo de fácil aplicação para a maioria dos cenários.

Objetivo	Hipótese
Baixa dimensionalidade e esparsidade	A representação aprendida pelo modelo deve possuir baixa dimensionalidade e ser densa, evitando assim o problema de alta esparsidade
Eficiência computacional	O modelo deve apresentar um custo computacional igual ou inferior aos demais métodos de recomendação baseados em representações vetoriais
Praticidade de aplicação	O modelo deve demandar apenas uma matriz de interações de *feedback* implícito para ser treinado, sem a necessidade de dados de conteúdo ou informações adicionais

Tabela 1 – Hipóteses do Interact2Vec

Neste capítulo, é detalhado o funcionamento do Interact2Vec, inclusive sua arquitetura e função-objetivo. São também apresentadas técnicas importantes empregadas durante a etapa de aprendizado e são sugeridas diferentes formas de se utilizar as RVDs para construir a recomendação. Por fim, é apresentada análise a de complexidade computacional e limitações do modelo.

3.1 Arquitetura da rede e função de custo

O Interact2Vec é uma rede neural artificial simples, de arquitetura inspirada no Item2Vec (BARKAN; KOENIGSTEIN, 2016) e objetivo inspirado no User2Vec (GRBOVIC et al., 2015). A rede possui apenas três camadas: uma camada de entrada $\mathcal{J}$, uma camada intermediária $\mathcal{H}$ e uma camada de saída $\mathcal{O}$, conforme ilustrado na Figura 3.

A camada de entrada $\mathcal{J}$ corresponde a um vetor de tamanho $|U|$ (número de usuários do sistema) representando um usuário-alvo $u \in U$ codificado em formato *one-hot*. A camada intermediária $\mathcal{H}$ possui tamanho M, correspondente ao número de dimensões desejadas para as RVDs finais. Ela é conectada a camada $\mathcal{J}$ por uma matriz de pesos

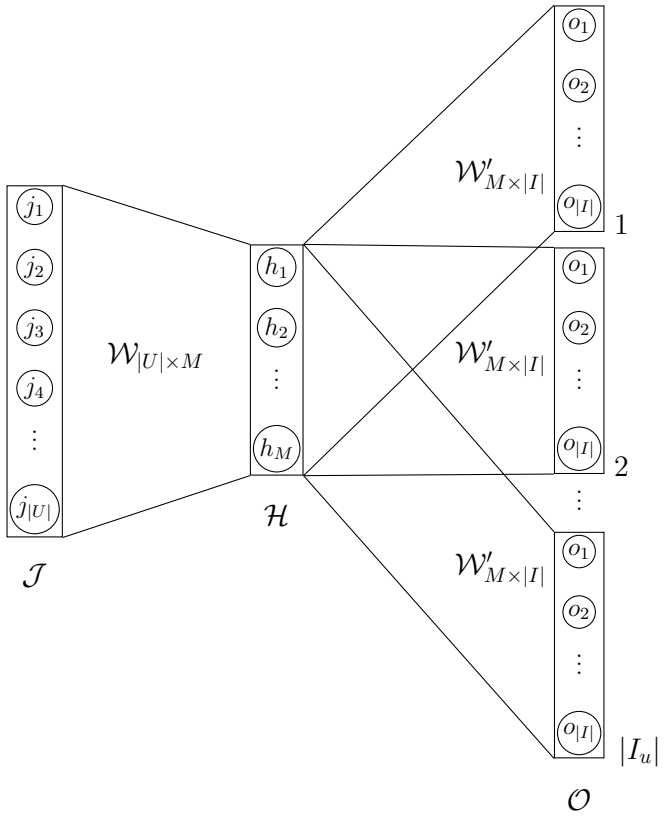

Figura 3 – Arquitetura do Interact2Vec

$\mathcal{W}$ de dimensões $|U| \times M$, de forma que $\mathcal{H} = \mathcal{J} \times \mathcal{W}$. A camada de saída $\mathcal{O}$ é formada por $|I_u|$ vetores, cada um de tamanho $|I|$ (número de itens do catálogo), representando os itens consumidos pelo usuário u, em formato *one-hot*. Cada um dos vetores é conectado a $\mathcal{H}$ por uma mesma matriz de pesos $\mathcal{W}'$ de dimensões $M \times |I|$ e uma função de ativação *softmax*. Assim, $\mathcal{O} = softmax(\mathcal{H}^T \times \mathcal{W}')$.

De maneira similar ao User2Vec, o Interact2Vec aprende RVDs de usuários e itens através de relações usuário-item. Seu objetivo principal é, dado um usuário u como entrada, prever como saída da rede todos os itens i consumidos previamente por u, ou seja, $i \in I_u$. Matematicamente, tem-se que seu objetivo é maximizar a Equação 3.1.

$$\frac{1}{|U|} \sum_{u \in U} \left(\frac{1}{|I_u|} \sum_{i \in I_u} \log \sigma(u, i) \right) \tag{3.1}$$

sendo σ a função *softmax* descrita na Equação 3.2:

$$\sigma(u, i) = \frac{e^{(\mathcal{W}_u \mathcal{W}_i'^T)}}{\sum_{j \in I} e^{(\mathcal{W}_u \mathcal{W}_j'^T)}} \tag{3.2}$$

onde W_u e $W_i'^T$ representam as RVDs do usuário u e item i, respectivamente.

Para cada iteração, o erro é computado como a soma das diferenças entre as

one-hots esperadas e as saídas preditas, como em um problema tradicional de classificação, e os pesos da rede são atualizados por *backpropagation*. Após o treinamento, feito ao longo de C iterações na rede, as linhas da matriz $\mathcal{W}$ serão as RVDs de usuários, e as colunas da matriz $\mathcal{W}'$ serão as RVDs de itens.

Como pode ser visto, por compartilharem uma arquitetura semelhante, a função objetivo do Interact2Vec é similar a função objetivo do Item2Vec (Equação 2.1). Contudo, enquanto que no Item2Vec o modelo busca maximizar sua capacidade preditiva sobre pares de itens consumidos por um usuário, no Interact2Vec, ele maximiza sua capacidade de predizer um único item dado um usuário.

3.1.1 Arquitetura simplificada

O treinamento da rede neural do Interact2Vec é feito através da multiplicação das RVDs de usuário e item para cada par usuário-item, atualizando os pesos de forma que essa multiplicação retorne 0 ou 1, de acordo com a ocorrência ou não da interação.

Quando a camada de entrada $\mathcal{J}$, contendo u em formato *one-hot*, é multiplicada pela matriz de pesos $\mathcal{W}$, a saída será a linha da matriz referente ao usuário u atual, ou seja, sua representação. Por esse motivo, a camada intermediária $\mathcal{H}$ pode ser chamada de camada de projeção, visto que ela projeta uma RVD.

Na etapa seguinte, o vetor de representação de usuário projetado é multiplicada pelas RVDs de todos os itens, armazenadas em $\mathcal{W}'$. Considerando que as saídas desejadas são os itens consumidos pelo usuário, representados em formato *one-hot*, pode-se concluir que a rede busca aprender pesos de forma que:

$$\mathcal{W}_u \times \mathcal{W}_i'^T = \begin{cases} 1, & \text{se o usuário } u \text{ interagiu com o item } i \\ 0, & \text{caso contrário} \end{cases} \tag{3.3}$$

Dessa forma, uma outra maneira de interpretar o modelo é: para cada possível par usuário-item, a rede deve *(i)* recuperar as RVDs de usuário e item consultando tabelas *look-up*; *(ii)* calcular seu produto vetorial, como feito na Equação 3.3; (iii) transformar o valor obtido para uma escala entre 0 e 1, usando uma função de ativação *sigmoid* ϕ, (Equação 3.4), e *(iv)* atualizar o conteúdo das RVDs para aproximar o resultado obtido do desejado. A arquitetura da rede passa a ser então a presente na Figura 4.

$$\phi(u, i) = \frac{1}{1 + e^{-(\mathcal{W}_u \mathcal{W}_j'^T)}} \tag{3.4}$$

Interpretar a arquitetura desta maneira pode ajudar a simplificar a implementação do modelo e possibilita o uso de diferentes técnicas tradicionais de aprendizado em métodos de RVD, que serão explicadas na seção seguinte.

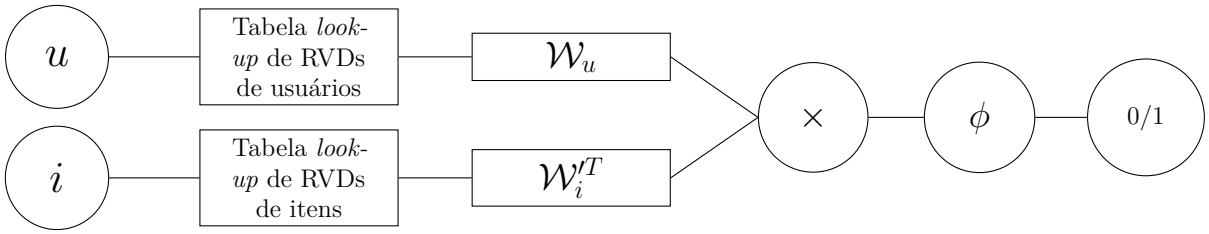

Figura 4 – Arquitetura do funcionamento prático do Interact2Vec

3.2 Estratégias de otimização do aprendizado

Assim como outros modelos neurais para geração de RVD, o Interact2Vec emprega diversas técnicas durante a etapa de aprendizado com o intuito de aumentar seu poder de generalização e reduzir seu custo computacional. Isto permite que o método se adapte melhor para o domínio da aplicação e não demande elevado poder de processamento. Entretanto, tem a desvantagem de possuir uma quantidade maior de parâmetros para serem ajustados.

As estratégias empregadas pelo Interact2Vec, são: subamostragem de itens frequentes, amostragem negativa e regularização dos modelos vetoriais. Cada uma destas técnicas são apresentadas nas seções seguintes.

3.2.1 Subamostragem de itens frequentes

Experimentos com métodos de RVD na área de Processamento de Linguagem Natural mostraram que realizar uma etapa de subamostragem de palavras frequentes antes do treinamento do modelo – removendo assim ocorrências de palavras muito comuns, como *"the"* ou *"you"* – pode contribuir significativamente para seu poder de generalização e, consequentemente, aumentar a acurácia em aplicações práticas (MIKOLOV et al., 2013). A estratégia, já consolidada em tarefas textuais, foi amplamente adotada em modelos de RVD para recomendação (BARKAN; KOENIGSTEIN, 2016; VASILE; SMIRNOVA; CONNEAU, 2016; CASELLES-DUPRéS; LESAINT; ROYO-LETELIER, 2018). Neste cenário, a remoção é feita no nível de interações, com o objetivo de reduzir o número de interações de itens populares, isto é, que são bastante recorrentes dentro da base de dados. Para isso, é normalmente adotada uma função de cálculo de probabilidade para permanência dos itens. Uma das mais utilizadas, por ser implementada em modelos Word2Vec (MIKOLOV et al., 2013), é reproduzida na Equação 3.5.

$$P(\text{descarte} \mid i) = \left(\sqrt{\frac{z(i)}{\rho}} + 1\right) \cdot \frac{\rho}{z(i)} \tag{3.5}$$

na qual, $z(i)$ é uma função responsável por retornar a quantidade de ocorrências do item dentro da base de dados, ou seja, $|U_i|$, e ρ é uma constante parametrizável responsável por

controlar a taxa de subamostragem.

Utilizando a equação apresentada, é possível parametrizar ρ e mudar a frequência de remoção de itens. Dessa forma, quanto maior o valor de ρ, maior a probabilidade de uma interação ser removida da base, o que também aumenta de forma proporcional a $z(i)$.

3.2.2 Amostragem negativa

Gerar classificações positivas ou negativas para todos os possíveis pares usuário-item do sistema é um processo bastante custoso computacionalmente, crescendo quadraticamente conforme novos usuários e itens são adicionados. Para aliviar esse problema, Mikolov et al. (2013) propuseram uma técnica batizada de *negative sampling*, ou amostragem negativa. Através dela, a atualização dos pesos da rede deixa de ser feita considerando todos os itens "negativos" (ou seja, não consumido pelo usuário), e passa a observar o resultado da predição apenas sobre um conjunto G composto por itens negativos selecionados de forma aleatória, sendo $|G|$ (o número de itens negativos) uma constante parametrizável. Dessa forma, a função objetivo da rede, anteriormente expressa pela Equação 3.1, passa a ser a função presente na Equação 3.6, que também deve ser maximizada.

$$\frac{1}{|U|}\sum_{u\in U}\left(\frac{1}{|I_u|}\sum_{i\in I_u}\left(\log\sigma(u,i)-\sum_{j\in G}\log\sigma(u,j)\right)\right) \tag{3.6}$$

O conteúdo de G (isto é, os itens negativos) deve ser atualizado a cada item positivo ($i \in I_u$) para cada iteração da rede. A seleção é feita de forma aleatória, seguindo uma distribuição de probabilidade calculada de acordo com a frequência de cada item, como proposta por Mikolov et al. (2013). O objetivo da seleção é fazer com que itens mais frequentes tenham maior probabilidade de serem selecionados, sendo a probabilidade de seleção calculada pela Equação 3.7.

$$P(i) = \frac{z(i)^\gamma}{\sum_{j\in I} z(j)^\gamma} \tag{3.7}$$

sendo que, $z(i)$ representa a quantidade de interações do item, ou seja, $|U_i|$, e γ é uma constante parametrizável com o intuito de balancear a probabilidade de seleção entre itens populares e itens pouco frequentes. Dentro da área de PLN, é comum o uso de $\gamma = 0.75$ (MIKOLOV et al., 2013), de forma que itens frequentes ainda possuam maior chance de serem selecionados. O valor é comumente adotado na área de recomendação, entretanto experimentos empíricos conduzidos por Caselles-Duprés, Lesaint e Royo-Letelier (2018) mostraram que o uso de valores negativos para γ, fazendo assim com que G tenha maiores chances de ser composto por itens menos frequentes, pode melhorar consideravelmente a qualidade das representações.

3.2.3 Regularização dos vetores de representação

Com o objetivo de evitar um possível super-ajustamento por parte do modelo, o Interact2Vec ajusta o conteúdo das RVDs ponderando-o por um termo de regularização, buscando assim melhorar a capacidade de generalização das representações aprendidas, como mostrado por Zhang, Lu e Shai (2018).

Foram aplicados termos de regularização de regressão de Ridge (L2) em cada uma das matrizes de RVDs $\mathcal{W}$ e $\mathcal{W}'$. Desta forma, o objetivo da rede passa a ser maximizar a função presente na Equação 3.8:

$$\frac{1}{|U|}\sum_{u\in U}\left(\frac{1}{|I_u|}\sum_{i\in I_u}\left(\log\,\sigma(u,i)-\sum_{j\in G}\log\,\sigma(u,j)\right)\right)-\lambda\sum_{i\in I}||\mathcal{W}'^{T}_{i}||^2-\lambda\sum_{u\in U}||\mathcal{W}_u||^2 \quad (3.8)$$

onde λ é uma constante parametrizável, responsável por definir a importância da regularização durante o ajuste do conteúdo das RVDs. Quanto maior o valor de λ, mais o conteúdo das representações vetoriais será regularizado, evitando assim uma grande variação nos valores que compõe a representação.

3.3 Estratégias de recomendação

O Interact2Vec, assim como o Item2Vec e o User2Vec, não é por si próprio um método de recomendação, dado que seu objetivo é apenas criar uma representação vetorial distribuída de usuários e itens. Após as RVDs terem sido criadas pelo modelo, elas podem ser consumidas por outros métodos de recomendação. Por representarem usuários e itens em um mesmo espaço vetorial, recomendadores baseados em vizinhança podem ser empregados para construir a recomendação final. Em adição, o fato de as RVDs apresentarem dimensionalidade reduzida permite que até mesmo técnicas de elevado custo computacional sejam executadas em um intervalo de tempo aceitável.

Por consumir apenas dados de *feedback* implícito, ignorando avaliações explícitas, o tipo de recomendação destinado para as RVDs do Interact2Vec é o ranqueamento top-N. Ainda assim, é possível utilizá-las em um cenário de previsão de nota, usando-as para o cálculo das distâncias entre itens.

Nas seções seguintes, são apresentados métodos que podem utilizar as RVDs do Interact2Vec para gerar a recomendação final, tanto para problemas de ranqueamento top-N quanto de previsão de notas.

3.3.1 Ranqueamento top-N

No ranqueamento top-N, o sistema de recomendação deve selecionar um conjunto ordenado de N itens para recomendar a um usuário alvo. Os algoritmos de recomendação

adotados para esta tarefa tem como característica o consumo e processamento da representação vetorial gerada pelo Interact2Vec, aplicando medidas de similaridade e técnicas de filtragem para selecionar e ordenar os N itens da recomendação.

São propostas cinco técnicas diferentes para a recomendação: *(i)* similaridade usuário-item, que verifica quais as RVDs de itens são mais similares a *embedding* do usuário alvo no espaço vetorial; *(ii)* similaridade item-item, que utiliza apenas as RVDs de itens para calcular uma similaridade entre o usuário alvo a um possível item a ser recomendado; *(iii)* similaridade ponderada, que combina as duas técnicas anteriores; *(iv)* combinação de RVDs, que transforma as RVDs de itens concatenando-as com RVDs de usuários, para depois calcular uma similaridade item-item; e *(v)* comitê de métodos, que gera a recomendação através de uma votação entre todos os métodos anteriores. Cada uma dessas técnicas é explicada em detalhes nas seções seguintes.

3.3.1.1 Similaridade usuário-item

Durante a etapa de aprendizado das RVDs, o Interact2Vec busca aproximar usuários e itens que interagiram entre si, distanciando os demais. Isso faz com que usuários e itens em comum acabem sendo atraídos para uma mesma vizinhança.

A partir do momento que há uma vizinhança compartilhada entre usuários e itens, a recomendação top-N para um usuário alvo pode ser feita através dos N itens mais próximos do usuário no espaço vetorial, que ainda não tenham sido consumidos por ele, como ilustrado na Figura 5

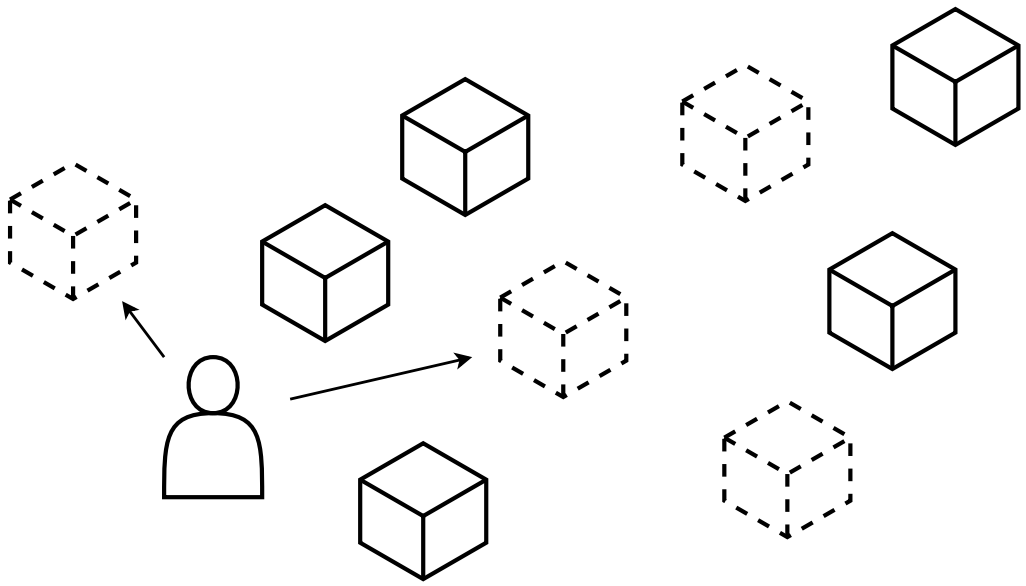

Figura 5 – Recomendação por similaridade usuário-item.

O usuário é representado pela imagem de pessoa e os itens pelas imagens de cubo, sendo aqueles de borda tracejada os itens não consumidos pelo usuário.

Dessa forma, dado um usuário u, é necessário calcular a similaridade entre u e todo item i ainda não consumido por ele, isto é, $sim(u,\ i)\ \forall\ i \mid i \notin I_u$. A recomendação é feita selecionando os N mais similares.

Para calcular a similaridade entre uma *embedding* de usuário e uma de item, diversas medidas de similaridade ou vizinhança podem ser empregadas (DESROSIERS; KARYPIS, 2011). Entretanto, é recomendado que seja utilizada a similaridade de cosseno, por ser a medida de similaridade que mais aproxima-se da função objetivo otimizada pelo Interact2Vec. A similaridade de cosseno é uma medida que avalia a similaridade angular entre dois vetores, como mostrado na Equação 3.9.

$$sim(u,\ i) = \frac{\mathcal{W}_u \cdot \mathcal{W}'^{T}_{i}}{||\mathcal{W}_u|| \times ||\mathcal{W}'^{T}_{i}||} = \frac{\sum_{l=1}^{M} \mathcal{W}_{u_l} \times \mathcal{W}'^{T}_{i_l}}{\sqrt{\sum_{l=1}^{M} \mathcal{W}_{ul}} \times \sqrt{\sum_{l=1}^{M} \mathcal{W}'^{T}_{i_l}}} \tag{3.9}$$

onde $\mathcal{W}_u$ representa a *embedding* do usuário u, e $\mathcal{W}'^{T}_{i}$ a *embedding* do item.

3.3.1.2 Similaridade item-item

Uma segunda abordagem possível de ser utilizada para gerar a recomendação com as RVDs do Interact2Vec é desconsiderar a representação vetorial dos usuários, e construir a recomendação através de um cálculo de similaridade item-item, estratégia tradicionalmente adotada na literatura para recomendação top-N (DESHPANDE; KARYPIS, 2004). Assim, a seleção de itens a serem recomendados é feita com base na representação vetorial dos itens previamente consumidos pelo usuário, como na Figura 6.

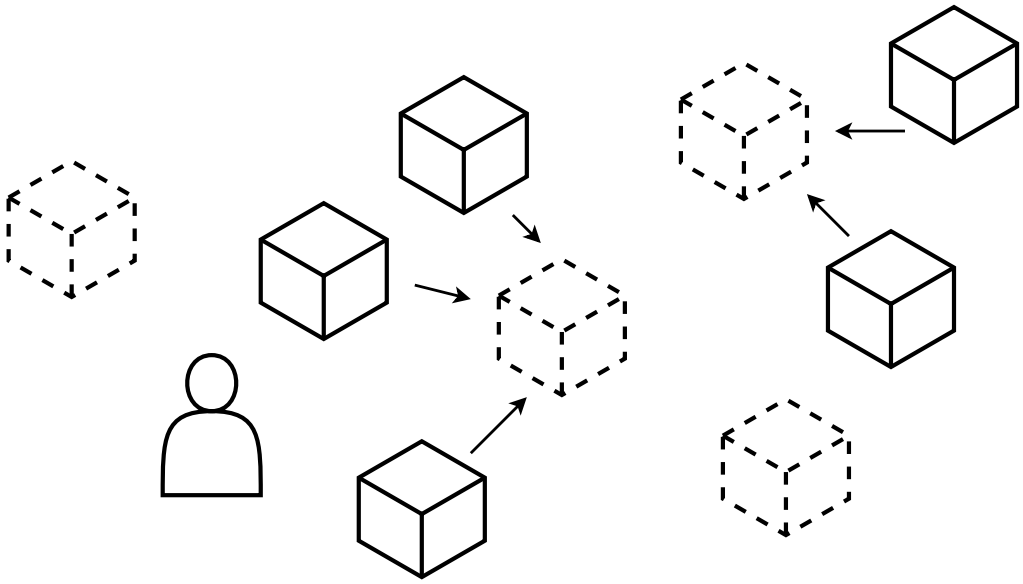

Figura 6 – Recomendação por similaridade item-item.

O usuário é representado pela imagem de pessoa e os itens pelas imagens de cubo, sendo aqueles de borda tracejada os itens não consumidos pelo usuário.

Neste método de recomendação, ainda é computada uma similaridade entre o usuário-alvo e os itens não consumidos, representada por $sim(u,\ i)$ para que os N mais similares sejam recomendados. Entretanto, $sim(u,\ i)$ é calculada através da similaridade média entre os itens consumidos pelo usuário e o restante do catálogo, como mostrado na Equação 3.10. Assim, é necessário utilizar apenas as RVDs de itens.

$$sim(u, i) = \frac{\sum_{j \in I_u} sim(i,\ j)}{|I_u|} \tag{3.10}$$

sendo $sim(i,\ j)$ a similaridade de cosseno entre as RVDs dos itens i e j, respectivamente $\mathcal{W}'^{T}_{i}$ e $\mathcal{W}'^{T}_{j}$, calculadas da mesma maneira como mostrado na Equação 3.9.

3.3.1.3 Similaridades ponderadas

Em muitos cenários, a disposição dos itens e usuários no espaço vetorial pode ocasionar divergências entre o retorno dos dois métodos de recomendação supracitados. Assim, pode ser interessante combiná-los, de forma que a recomendação final seja construída através de uma votação ponderada entre os dois métodos, como apresentado na Figura 7.

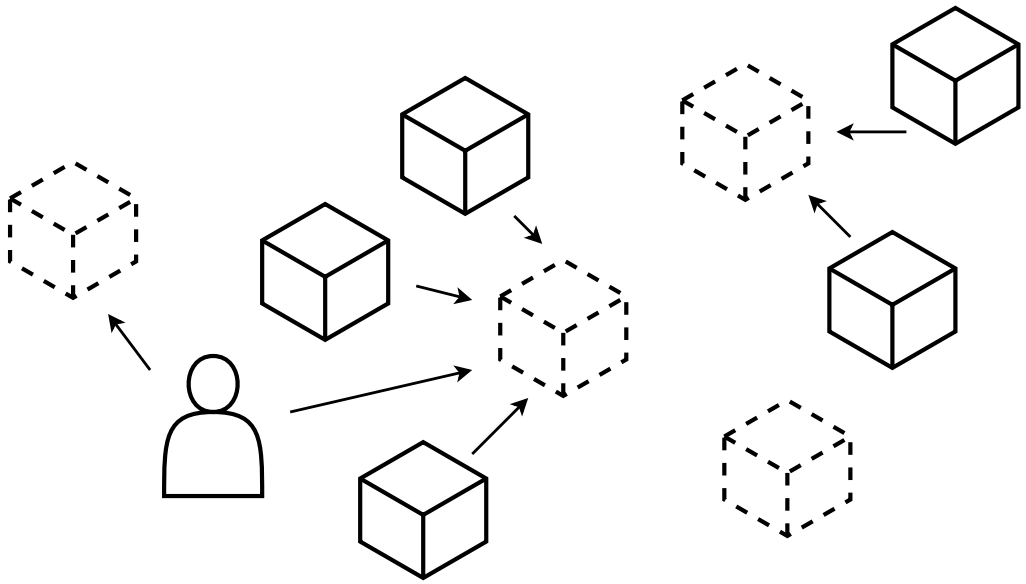

Figura 7 – Recomendação por similaridade ponderada entre usuário-item e item-item.

O usuário é representado pela imagem de pessoa e os itens pelas imagens de cubo, sendo aqueles de borda tracejada os itens não consumidos pelo usuário.

Nesta abordagem, é necessário calcular a similaridade $sim_w(u,\ i)$ entre o usuário alvo e todos os itens não consumidos por ele, recomendando os top-N mais similares. O valor de $sim_w(u,\ i)$ pode ser calculado através da Equação 3.11.

$$sim_w(u,\ i) = \frac{\beta \times sim_u(u,\ i) + \mu \times sim_i(u,\ i)}{\beta + \mu} \tag{3.11}$$

na qual $sim_u(u,\ i)$ é responsável por calcular a similaridade usuário-item, como explicado na Seção 3.3.1.1, $sim_i(u,\ i)$ é responsável por calcular a similaridade item-item, como explicado na Seção 3.3.1.2, e β e μ são constantes responsáveis por definir o peso que cada método de recomendação terá sobre a recomendação final.

3.3.1.4 Combinação de representações vetoriais distribuídas

A recomendação através da combinação de representações vetoriais distribuídas tem como estratégia aplicar a recomendação por similaridade item-item (Seção 3.3.1.2) sobre RVDs de itens transformadas. A transformação sobre as RVDs de itens é feita combinando-as com as RVDs de usuários, de forma que uma única representação vetorial carregue informação sobre os dois tipos de entidades.

De maneira geral, a combinação é feita por meio de concatenação da *embedding* de item com as de usuários. Entretanto, há duas maneiras de gerar essa concatenação: através da média entre as RVDs de usuário ou através da concatenação geral das RVDs.

Na abordagem da média, a nova *embedding* para um dado item i é caracterizada como $\left[\mathcal{W}'^{T}_{i},\ \frac{\sum_{u \in U_i} \mathcal{W}_u}{|U_i|}\right]$, ou seja, a concatenação entre a antiga *embedding* de i e a média entre as RVDs de todos os usuários que interagiram com i ($u \in |U_i|$). Adicionalmente, ao invés de calcular a média entre todos os usuários, é possível limitar a apenas uma quantidade de K usuários mais similares.

Já na abordagem de concatenação geral, a nova *embedding* para i passa a ser $\left[\mathcal{W}'^{T}_{i},\ \mathcal{W}_1,\ \mathcal{W}_2,\ \mathcal{W}_...,\ \mathcal{W}_K\right]$, isto é, a concatenação entre a antiga *embedding* de i e as RVDs de K usuários que consumiram i ($u \in |U_i|$) e que possuam uma representação vetorial similar a i.

A escolha por qual formato de combinação utilizar pode ser parametrizável, assim como o valor de K nas duas abordagens, permitindo assim uma melhor adaptação ao domínio da aplicação.

3.3.1.5 Comitê de métodos

A estratégia de recomendação baseada em um comitê de métodos tem como objetivo executar todas as demais estratégias de recomendação, explicadas nas seções anteriores, e construir uma recomendação final top-N por meio de votação entre os resultados de cada recomendação.

Para um dado usuário u, cada um dos métodos m deve gerar uma recomendação $s_m(u, L)$ composta por L itens, tal que $L \geq N$. Estratégias que possuam parâmetros ajustáveis, como β e $f\mu$ para a similaridade ponderada ou o formato da combinação e K para a combinação de RVDs, podem gerar múltiplas recomendações para diferentes

combinações de parâmetros. No final, todas as recomendações top-L geradas irão compôr um conjunto S de recomendações.

Com o conjunto construído, a recomendação final pode ser feita através de uma votação pelos itens selecionados por cada $s_m(u, L) \in S$. Há diferentes maneiras de se conduzir a votação: aplicando ou não um peso w_m para cada método, dado seu desempenho individual na recomendação, e aplicando ou não um peso w_r para cada item dado sua posição nos ranqueamentos gerados. De forma genérica, é possível calcular uma pontuação para cada item i na recomendação para um usuário u através da função score(i, u) presente na Equação 3.12, e montar a recomendação final através da seleção dos N itens de maior pontuação.

$$\text{score}(i,\ u) = \sum_{m=1}^{|S|} (f(i,\ u,\ m) \times w_m \times w_r) \tag{3.12}$$

onde $f(i,\ u,\ m)$ é uma função responsável por verificar se o item i está presente na recomendação $s_m(u, L)$ (Equação 3.13), w_m é um peso baseado na qualidade preditiva do método m que gerou a recomendação $s_m(u, L)$ (como a acurácia, por exemplo) e w_r é um peso baseado na posição do item i no ranqueamento $s_m(u, L)$. Quando não é desejado ponderar a pontuação pela qualidade do método ou pela ordem do ranqueamento, se faz necessário utilizar o valor 1 para w_m e w_r, respectivamente.

$$f(i,\ u,\ m) = \begin{cases} 1, & \text{se } i \in s_m(u,\ L) \\ 0, & \text{caso contrário} \end{cases} \tag{3.13}$$

A motivação para uso de um comitê de métodos para recomendação é que cada uma das estratégias de recomendação propostas possam contribuir para gerar a recomendação final, mantendo assim os itens mais acordantes entre as estratégias.

3.3.2 Previsão de notas

Na previsão de notas, o método recomendador deve ser capaz de, dado um par $(u,\ i)$ de usuário e item, estimar com qual nota o usuário classificaria o item.

Por não consumir *feedback* explícito durante o aprendizado, tratando como similares todos os itens consumidos pelo usuário independente da nota atribuída, as representações vetoriais geradas pelo Interact2Vec não são recomendadas para um cenário de previsão de notas. Ainda assim, é possível gerar a previsão através de uma abordagem de K-vizinhos mais próximos.

3.3.2.1 K-vizinhos mais próximos

O K-vizinhos mais próximos (KNN) é um método tradicional de recomendação explícita que prevê uma nota para um par usuário-item (isto é, $\hat{r}_{ui}$) observando as notas atribuídas a itens similares. Utilizando as RVDs do Interact2Vec, a estimativa pode ser calculada da maneira mostrada na Equação 3.14.

$$\hat{r}_{ui} = \frac{1}{|V|} \sum_{j \in V} r_{uj} \tag{3.14}$$

sendo V um conjunto dos K itens j mais similares a i e previamente consumidos por u ($V \subset I_u$), e r_{uj} a nota atribuída por u a estes itens. A similaridade entre dois itens i e j é calculada através da similaridade de cosseno entre as RVDs de i e j, através da Equação 3.9.

3.4 Quantidade de operações

Um dos principais objetivos do Interact2Vec em relação a outros métodos baseados em RVD é apresentar maior simplicidade e eficiência computacional, sendo assim uma rede neural com poucas camadas, que demanda um reduzido número de etapas de propagação para ser ajustada. Para compreender esta característica do modelo, é apresentado nesta seção a quantidade de operações realizadas para treinamento da representação, comparando-a com a complexidade de outros métodos tradicionais na área, como o KNN, o SVD e o Bayesian Personalized Ranking (BPR), e outros modelos de RVD, como o Item2Vec e o User2Vec.

Considerando todo o processo de um sistema de recomendação, o Interact2Vec, assim como outros modelos de RVD e de fatoração de matriz, possui duas etapas: a geração da representação e a recomendação em si. Cada uma das etapas será tratada de forma independente para o cálculo da complexidade computacional do modelo, visto que, além de não exercerem influência entre si, é possível adotar diferentes técnicas de recomendação após o aprendizado das RVDs, conforme apresentado na Seção 3.3.

Para o cálculo da quantidade de operações, será considerado que operações básicas como adição e multiplicação são constantes no tempo, isto é, $O(1)$. Adicionalmente, será considerado que qualquer multiplicação matricial é implementada na forma mais básica, possuindo assim complexidade $O(n^3)$. Embora seja possível otimizar a multiplicação por meio do algoritmo de Strassen (1969), a nova complexidade não ocasionará em mudanças na comparação entre os modelos, permitindo assim que a solução cúbica seja adotada para a explicação. Por fim, para todos os modelos, a etapa de otimização não será considerada, visto que é altamente dependente do algoritmo otimizador adotado. Assim, apenas as operações da função de custo de cada modelo serão comparadas.

3.4.1 Geração da representação vetorial

Para o cálculo da complexidade computacional de geração das RVDs do Interact2Vec, pode-se decompor o funcionamento da rede em cinco etapas, facilitando assim o cálculo e o entendimento. São elas: *(i)* a propagação e ajuste de pesos para uma única amostra; *(ii)* para um item positivo e seus $|G|$ itens negativos; *(iii)* para um único usuário; *(iv)* para todos os usuários; e *(v)* para todas as iterações.

A propagação e ajuste de pesos para uma única amostra consiste em alimentar a rede com um único usuário u tendo como objetivo a previsão de um único item i. Para o cálculo, será adotada a implementação explicada na Seção 3.1.1, correspondente ao uso de tabelas *look-up* para acesso às RVDs. Assim, é necessário recuperar a *embedding* de u e de i, e calcular o produto escalar entre as duas. A recuperação das RVDs pode ser feito em complexidade $O(1)$, sendo uma simples consulta a uma tabela indexada. Já o produto escalar, por se tratar de uma série de operações de adição e multiplicação sobre os elementos das RVDs, possuirá complexidade diretamente proporcional ao tamanho da representação vetorial, isto é, $O(2M)$. Desta forma, a propagação de uma única amostra pela rede possui complexidade $O(M)$. O ajuste dos pesos, por realizar as mesmas operações sobre os vetores, não alterará a complexidade.

Considerando o uso de amostragem negativa (Seção 3.2.2), para cada item positivo alimentado à rede, é necessário realizar o mesmo procedimento de propagação e ajuste para $|G|$ itens negativos. Sendo assim, o ajuste da rede para um único item positivo repetirá uma operação de complexidade $O(M)$ por $|G| + 1$ vezes, possuindo assim complexidade $O((|G| + 1) \times M) = O(|G| \times M + M) = O(|G| \times M)$.

A etapa de ajuste da rede para um item positivo é feita para cada item consumido pelo usuário. Dessa forma, para um dado usuário u, a operação será realizada $|I_u|$ vezes, apresentando assim uma complexidade $O(|I_u| \times |G| \times M)$.

Como a rede é treinada com as interações de todos os usuários da base, tem-se que a operação acima será executada $|U|$ vezes. Na realidade, a rede realiza uma etapa de ajuste de um item positivo para cada interação usuário-item na base. Assim, a complexidade computacional para uma única iteração da rede será $O(|R| \times |G| \times M)$. Finalmente, como o ajuste do modelo é feito ao longo de C iterações, sua complexidade computacional final será $O(C \times |R| \times |G| \times M)$.

Desta forma, tem-se que o custo computacional de gerar as RVDs do Interact2Vec está diretamente relacionado a quatro fatores: a dimensionalidade desejada para a representação, o número de amostras negativas, a quantidade de interações usuário-item na base e o número de iterações do treinamento. Entretanto, na prática, o conteúdo de $|R|$ – quantidade de interações na base – é expressivamente superior aos demais valores ($|R| \gg \{M,\ |G|,\ C\}$), permitindo concluir que o tempo de ajuste do Interact2Vec cresce

de forma praticamente linear ao número de interações na base de dados, tendo uma complexidade $O(|R|)$.

O User2Vec, por realizar operações similares e ser ajustado para cada par usuário-item na base, também apresenta uma complexidade $O(|R|)$. Ainda assim, o modelo possui uma operação adicional ao Interact2Vec: o cálculo da média entre as RVDs de itens com a de usuário para compor o conteúdo a ser projetado na camada intermediária. Entretanto, a média entre um conjunto de vetores de tamanho M terá complexidade $O(M)$, não impactando na complexidade final (embora, na prática, isso possa resultar em um desempenho levemente superior por parte do Interact2Vec).

O Item2Vec funciona de maneira similar ao Interact2Vec, mas apresenta uma diferença significativa. Por ter como objetivo receber um item consumido por um usuário e prever os demais itens consumidos por ele, para cada interação entre um usuário u e um item i, o modelo realiza $|I_u| - 1$ etapas de propagação. Como o restante das operações do Item2Vec são similares ao Interact2Vec, tem-se que a complexidade computacional do modelo é aproximadamente $O(C \times |U| \times (|I_u| - 1) \times |G| \times M)$, estando assim condicionado ao número de itens consumidos por cada usuário.

Em um cenário onde cada usuário consome apenas dois itens (havendo assim apenas duas etapas de propagação por usuário), a complexidade computacional do Item2Vec passa a ser $O(C \times |U| \times 2 \times |G| \times M) \approx O(C \times |R| \times |G| \times M) \approx O(|R|)$, assemelhando-se ao Interact2Vec. Entretanto, no pior caso onde cada usuário consome todos os itens do catálogo, tem-se $O(C \times |R| \times (|R| - 1) \times |G| \times M) \approx O(|R|^2)$, tendo desempenho quadrático em relação ao tamanho da base. Na prática, a quantidade de itens consumida por cada usuário será pequena quando comparada ao número de itens no catálogo, garantindo uma complexidade aproximada de $O(|R|)$. Entretanto, o Interact2Vec, por realizar um número inferior de operações, sempre apresentará um desempenho melhor que o Item2Vec, aumentando sua vantagem em cenários onde usuários interagem com muitos itens.

Técnicas de fatoração de matriz para recomendação implícita, também possuem uma etapa de geração da representação vetorial (no caso, da descoberta dos fatores latentes). Em sua versão direta, o SVD de Hu, Koren e Volinsky (2008) possui complexidade de $O(|U| \times |I|)$, o que pode alcançar valores que tornam sua aplicação inviável em cenários reais. Desta forma, os autores propuseram técnicas para otimizá-lo, atingindo complexidade $O(M^2 \times |R| + M^3)$ para cada uma das C iterações, onde M corresponde ao número de fatores latentes, apresentando assim complexidade final $O(C \times M^2 \times |R| + C \times M^3)$. De forma semelhante ao Interact2Vec, a quantidade de operações do SVD cresce de forma linear em conjunto com o número de interações, visto que $|R| \gg \{M,\ C\}$. Ainda assim, em cenários onde o número de fatores latentes do SVD for maior que o número de amostras negativas selecionadas pelo Interact2Vec (isto é, $M > |G|$), o modelo neural apresentará um número de operações levemente inferior, podendo assim apresentar maior eficiência

que o método de fatoração de matriz.

O BPR (RENDLE et al., 2009), outro popular método de fatoração de matriz para *feedback* implícito, realiza uma etapa de previsão do relacionamento entre usuário e itens para cada tupla (usuário, item_1, item_2). Esta previsão pode ser feita de diferentes maneiras, com a mais eficiente sendo através do cálculo do produto vetorial entre os fatores latentes dos vetores de representação, o que possui complexidade $O(M)$. Se realizado para todos os conjuntos possíveis de usuário e par de itens, a operação seria realizada $|R| \times |I|$ vezes ao longo de C iterações, obtendo assim uma complexidade computacional de $O(C \times |R| \times I \times M)$. Neste cenário, o BPR seria extremamente mais custoso que o Interact2Vec, visto que $|I| \gg |G|$. Entretanto, os autores do método propuseram o uso de reamostragem de *bootstrap* para acelerar a execução, selecionando um número reduzido e aleatório de amostras e encerrando a execução do método em caso de convergência, o que, segundo experimentos, faz com que a quantidade de operações do método cresça de forma linear a $|R|$, obtendo comportamento semelhante ao Interact2Vec, isto é, $O(|R|)$.

3.4.2 Geração da recomendação

Embora existam diversas técnicas diferentes para consumir a representação vetorial e recomendar itens, todas envolvem ao menos uma etapa de multiplicação matricial, seja para reconstrução da matriz ou para o cálculo de medidas de similaridade. Assim, é possível afirmar que gerar a recomendação com o Interact2Vec, o Item2Vec, o User2Vec, o SVD ou o BPR sempre apresentará complexidade cúbica.

O KNN, método que não possui etapa de geração de representação vetorial, também constrói a recomendação por meio de uma multiplicação matricial para cálculo de similaridade e descoberta da vizinhança, teoricamente apresentando a mesma complexidade que os demais métodos. Entretanto, em métodos de RVD ou fatoração de matriz, a multiplicação matricial é realizada sobre matrizes de dimensão reduzida $|U| \times M$ ou $|I| \times M$, enquanto no KNN a representação utilizada possui dimensões $|U| \times |I|$.

Como a dimensionalidade das RVDs é sempre significativamente menor que o número de usuários e itens da base, isto é, $M \ll |U|$ ou $|I|$, é possível afirmar que a recomendação do KNN sobre a base de interações completa será sempre menos computacionalmente eficiente do que as recomendações geradas sobre modelos de representação reduzida, tal como o Interact2Vec.

3.5 Limitações

O Interact2Vec é um modelo de representação vetorial distribuída para recomendação capaz de aprender RVDs de itens e usuários de forma simultânea e eficiente. Embora possua vantagens em relação a outros modelos, ele apresenta certas desvantagens e li-

mitações, que devem ser observadas com atenção no momento de aplicá-lo em cenários reais.

O Interact2Vec possui vários hiper-parâmetros que podem ser ajustados. Antes de sua execução, é necessário definir: o tamanho M das RVDs, a taxa de subamostragem de itens frequentes ρ, a quantidade de amostras negativas $|G|$, a potência de distribuição da probabilidade de amostras negativas γ e o parâmetro de regularização λ, além de outras características gerais de redes neurais, como o número de iterações da rede, a taxa de aprendizado do otimizador e o tamanho do *batch* para ajuste dos pesos.

Possuir uma elevada quantidade de hiper-parâmetros para serem ajustados não é uma propriedade exclusiva do Interact2Vec, ocorrendo também no Item2Vec, User2Vec e diversos outros métodos de RVD ou baseados em redes neurais. Entretanto, isso pode ser visto como uma limitação que o método apresenta contra outros mais simples, como o KNN, e pode dificultar sua aplicação em certos cenários. Por outro lado, o elevado número de parâmetros confere maior capacidade de adaptação do método a diferentes cenários.

Outro fator importante a ser observado, é o fato de seu aprendizado depender das relações entre usuário e item. Como mostrado na literatura (SARWAR et al., 2001; GRBOVIC et al., 2015), métodos baseados nos usuários do sistema precisam ser atualizados mais frequentemente para manterem sua qualidade, visto que as preferências dos usuários mudam de forma mais acelerada que as características dos itens (comparando-os a métodos baseados apenas em itens, como o Item2Vec e o KNN). Todavia, em adição o Interact2Vec ser um modelo eficiente e passível de atualizações constantes (a depender do tamanho do conjunto de dados), gerar RVDs de usuários pode ser interessante para outras aplicações dentro da área de sistemas de recomendação, como, por exemplo, o agrupamento de usuários semelhantes, tornando-se assim uma alternativa interessante de acordo com o domínio.

Por fim, ao não utilizar dados de *feedback* explícito no aprendizado, o modelo não é recomendado para a tarefa de previsão de notas. Ainda assim, a grande maioria dos sistemas de recomendação modernos é baseada em abordagens top-N, fazendo com que o Interact2Vec seja um método hábil para uma larga parcela de aplicações reais.

3.6 Considerações finais

Neste capítulo, foi apresentado o Interact2Vec, um novo modelo proposto para geração de RVDs de itens e usuários para recomendação. Através de uma rede neural rasa, computacionalmente eficiente, o modelo tem como principais objetivos ser capaz de gerar representações de baixa dimensionalidade e esparsidade apresentando quantidade de operações reduzida e facilidade de aplicação.

Em um primeiro momento, a arquitetura do modelo foi apresentada, assim como seu funcionamento. Em seguida, foram listadas técnicas utilizadas pelo modelo durante o aprendizado, sendo elas: a subamostragem de itens frequentes, para aumento no poder de generalização da representação; a amostragem negativa, para redução no número de operações realizadas; e a regularização das RVDs, com o intuito de evitar superajustamento.

Após a criação da RVD, é possível utilizar os vetores de representação de diversas maneiras para construir a recomendação. Foram então apresentadas cinco técnicas diferentes: similaridade entre usuário e item, similaridade entre itens, a união destas duas técnicas através de uma similaridade ponderada, a criação de uma nova representação combinando as outras duas, e um comitê de seleção composto por todas as quatro estratégias apresentadas, além de uma técnica baseada no algoritmo KNN para previsão de notas. Esta vasta quantidade de possíveis técnicas de recomendação mostram como a representação gerada pode ser consumida de diferentes maneiras, aumentando assim sua adaptatividade para diferentes domínios de aplicação.

Em seguida, foi calculada a complexidade computacional do modelo, comparando-o a outros métodos da área. Possuindo complexidade linear, o Interact2Vec mostrou-se uma opção promissora como recomendador baseado em geração de representação vetorial com baixo custo computacional.

Por fim, foram listadas as limitações e os problemas existentes com o método, sendo eles a alta quantidade de parâmetros a serem ajustados, a possível queda de qualidade ao longo do tempo por ser um método baseado em usuário, e a não utilização de *feedback* explícito, que limita o uso do modelo para solução de determinadas tarefas.

Apesar das limitações, as características e vantagens apresentadas pelo Interact2Vec fazem com que o mesmo seja um modelo de interessante aplicação para problemas reais. Para visualizar seu desempenho em cenários práticos, no capítulo seguinte são apresentados experimentos com o intuito de aferir a acurácia do modelo em tarefas como previsão de notas e ranqueamento top-N, observações sobre o conteúdo intrínseco da representação e uma análise do impacto da escolha de valores para os hiper-parâmetros do modelo.

4 Experimentos e resultados

Nesta seção, é descrito o protocolo experimental executado para avaliar o método proposto e compará-lo com os demais métodos da literatura. Inicialmente, são apresentadas as bases de dados utilizadas nos experimentos e o pré-processamento realizado. Em seguida, são conduzidos dois tipos de experimentos diferentes: uma avaliação extrínseca, com o objetivo de quantificar a qualidade dos métodos para gerar uma recomendação; e uma avaliação intrínseca, para compreender a capacidade das representações de aprender informações relacionadas ao conteúdo dos objetos representados. Em cada um dos experimentos, são explicadas as métricas de avaliação empregadas e os resultados finais, seguido de uma análise dos mesmos.

4.1 Base de dados e pré-processamento

Para realizar os experimentos, foram selecionados os conjuntos de dados apresentados na Tabela 4.1. A seleção se deu por serem bases amplamente presentes na literatura, disponíveis publicamente, que abordam diferentes áreas de aplicação e que possuem diferentes características entre si.

Os conjuntos marcados com asterisco (*) fazem parte do repositório do GroupLens[1], grupo de pesquisa do Departamento de Ciência da Computação e Engenharia da Universidade de Minnesota, referência e pioneiros na área de recomendação. Anime[2] é um conjunto de dados extraído do portal MyAnimeList, maior sistema de recomendação de obras de animação orientais. BestBuy[3] e NetflixPrize[4] são bases de dados utilizadas em competições organizadas pela Association for Computing Machinery (ACM) e Netflix, respectivamente. CiaoDVD e Filmtrust[5] são bases coletadas de um repositório *online* de bases de dados para sistemas de recomendação. RetailRocket[6] é a base de dados do sistema de recomendação RetailRocket, utilizado por mais de mil lojas de comércio eletrônico.

Na Tabela 4.1, as colunas $|U|$, $|I|$ e $|R|$ contém, respectivamente, o número de usuários, itens e interações de cada uma das bases. A coluna E contém a taxa de esparsidade,

[1] *GroupLens - Datasets*. Disponível em: <https://grouplens.org/datasets/>. Acesso em 08/03/2021.
[2] *Anime Recommendations Database*. Disponível em <https://www.kaggle.com/CooperUnion/anime-recommendations-database>. Acesso em 08/03/2021.
[3] *Data Mining Hackathon on BIG DATA (7GB) Best Buy mobile web site*. Disponível em: <https://www.kaggle.com/c/acm-sf-chapter-hackathon-big>. Acesso em 08/03/2021.
[4] *Netflix Prize data*. Disponível em: <https://www.kaggle.com/netflix-inc/netflix-prize-data>. Acesso em 08/03/2021.
[5] *CiaoDVD dataset* e *Filmtrust*. Disponível em: <https://github.com/caserec/Datasets-for-Recommender-Systems>. Acesso em 08/03/2021.
[6] Retailrocket recommender system dataset. Disponível em: <https://www.kaggle.com/retailrocket/ecommerce-dataset>. Acesso em 08/03/2021.

isto é, a quantidade de interações não existentes proporcional ao número máximo de interações possíveis ($E = 1 - \frac{|R|}{|U| \times |I|}$). A coluna "Tipo" contém o formato das interações: *feedback* explícito (E) ou *feedback* implícito (I). Nas bases Anime e Book-Crossing, o conteúdo da coluna é "E+I" por possuírem interações de ambos os tipos. Por fim, "Domínio" indica qual a área de aplicação da base de dados.

Conjunto	$\|U\|$	$\|I\|$	$\|R\|$	E	**Tipo**	**Domínio**
Anime	37.128	10.697	1.476.495	99,63%	E+I	Séries e Filmes
BestBuy	1.268.702	69.858	1.862.782	99,99%	I	*e-commerce*
Book-Crossing*	59.517	246.724	716.109	99,99%	E+I	Livros
CiaoDVD	17.615	16.121	72.345	99,97%	E	Séries e Filmes
DeliciousBookmarks*	1.867	69.223	104.799	99,92%	I	Páginas web
Filmtrust	1.508	2.071	35.494	98,86%	E	Séries e Filmes
Last.FM*	1.892	17.632	92.834	99,72%	I	Músicas
MovieLens*	162.541	59.047	25.000.095	99,74%	E	Séries e Filmes
NetflixPrize	480.189	17.770	100.480.507	98,82%	E	Séries e Filmes
RetailRocket	11.719	12.025	21.270	99,98%	I	*e-commerce*

Tabela 2 – Conjuntos de dados utilizados para treinamento e avaliação dos modelos de recomendação.

O pré-processamento envolveu a remoção de todas as entradas duplicadas, isto é, interações repetidas para um mesmo usuário-item e com avaliações (se disponível) iguais, mantendo apenas uma. Para interações inconsistentes, ou seja, aquelas que possuem par usuário-item em comum, mas avaliações diferentes, todos os exemplos foram removidos. Em seguida, para os conjuntos que possuem mais de um tipo de interação, Last.FM e RetailRocket, foram filtradas apenas as interações mais assertivas para indicar preferência de um usuário sobre um item. Para a Last.FM, apenas interações do tipo "ouvir um artista" foram consideradas, descartando as interações de "adicionar *tag* a um artista". Para a RetailRocket, apenas interações de compra de item foram mantidas, excluindo as interações de visualização, clique e adicionar ao carrinho. Os valores presentes na Tabela 4.1 já consideram essas filtragens.

4.2 Avaliação extrínseca

A avaliação extrínseca é o tipo de avaliação mais tradicional para sistemas de recomendação. Nela, os métodos são avaliados de acordo com sua qualidade de gerar recomendações, de forma a aferir seus desempenhos em um sistema de recomendação real.

Foram executadas duas tarefas diferentes de recomendação: o ranqueamento top-N e a previsão de nota. Para ambas, foi adotado o mesmo protocolo experimental para separação da base e ajuste de parâmetros, como apresentado na seção seguinte.

4.2.1 Protocolo experimental

Para realizar o treinamento e avaliação dos modelos, os conjuntos de dados – já pré-processados – foram divididos em três subconjuntos (treinamento, validação e teste), de forma aleatória, em proporção respectiva de 8:1:1. Em seguida, foi realizado um ajuste de parâmetros para cada modelo, seguido do experimento final para avaliação dos mesmos, utilizando os melhores parâmetros encontrados na etapa anterior. Tanto para o ajuste de parâmetros quanto para o experimento final, o procedimento realizado para manipulação dos dados foi o mesmo, que encontra-se descrito a seguir.

Inicialmente, devido a limitações computacionais, selecionou-se de forma aleatória uma subamostragem de interações nas bases de treinamento mais volumosas. A técnica foi empregada na otimização de parâmetros e no experimento final, porém foram utilizadas taxas de subamostragem diferentes em cada cenário. Como o ajuste de parâmetros é composto por múltiplas execuções do modelo, ao contrário do experimento final, que caracteriza-se por apenas uma, as taxas de subamostragem para o primeiro foram menores que para o segundo, como pode ser visto na Tabela 3. As bases de dados não presentes na tabela foram processadas com 100% das interações em ambos os cenários.

Conjunto	**Ajuste de parâmetros**	**Experimento final**
MovieLens	10%	100%
NetflixPrize	5%	25%

Tabela 3 – Taxas de subamostragem das bases de dados utilizadas nos experimentos

Após a subamostragem, foram removidos das bases de treinamento todos os usuários que possuíam apenas uma interação. O procedimento foi realizado para adequar a base aos métodos Item2Vec e User2Vec, que dependem que usuários consumam ao menos dois itens diferentes para poder alimentar suas redes neurais. Em seguida, eliminou-se o problema de *cold-start* nas bases de validação e teste, removendo todos os itens e usuários que não encontravam-se na base de treinamento. Desta forma, ao prever a nota ou gerar a recomendação, os métodos teriam a disposição todos os usuários e itens requisitados.

Após as etapas de remoção de usuários com uma única interação, e exclusão de itens e usuários não conhecidos pela base de treinamento, os modelos foram executados e avaliados em suas tarefas específicas. Para o ajuste de parâmetros, foram testados diferentes combinações de valores de forma exaustiva, por meio de busca em grade, e aqueles que apresentaram o melhor resultado foram utilizados no experimento final. Para o experimento final, foram executados todos os protocolos de manipulação dos dados mencionados, com a única diferença de que a base de treinamento foi combinada com a de validação, enriquecendo os dados disponíveis para o modelo aprender antes de ser aplicado sobre a base de teste. Uma representação gráfica das etapas executadas durante o experimento pode ser vista na Figura 8.

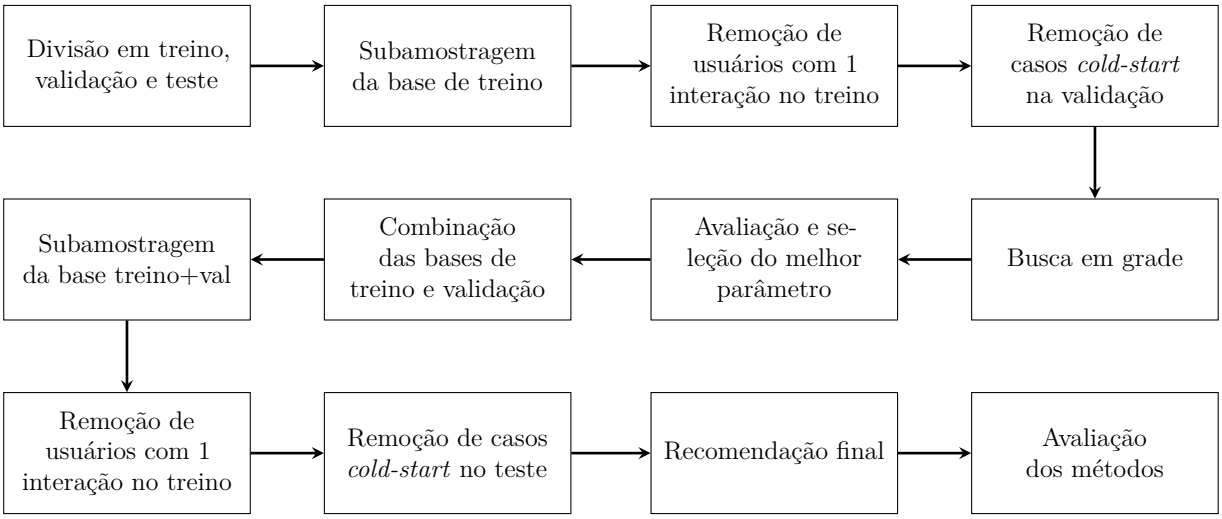

Figura 8 – Etapas do protocolo experimental

A descrição das métricas utilizadas, os métodos comparados e os parâmetros ajustados encontram-se a seguir, nas seções específicas de cada tarefa.

4.2.2 Ranqueamento top-N

A tarefa de ranqueamento top-N figura como a mais adotada entre sistemas de recomendação comerciais (DESHPANDE; KARYPIS, 2004). Nela, o método de recomendação deve, para cada usuário alvo, construir uma lista ordenada com N itens a serem recomendados (MUSTAFA; FROMMHOLZ, 2015).

Sistemas de recomendação top-N são avaliados através de uma comparação entre os itens recomendados e os itens consumidos pelo usuário, medindo o quão assertivo são os recomendadores. Com este objetivo, foi conduzido um experimento simulando um ranqueamento top-N. Métodos de recomendação foram treinados sobre a base de treinamento, recebendo como entrada um conjunto de *feedback* implícito entre usuários e itens (ou seja, interações sem valor quantitativo explícito, como uma nota ou uma preferência), para que fossem capazes de recomendar grupos ordenados de N itens para os usuários presentes nas bases de validação e, posteriormente, teste.

Nas seções a seguir, são apresentadas as medidas de avaliação, os métodos de *feedback* implícito selecionados, detalhes sobre o ajuste de parâmetros de cada um dos métodos e os resultados finais, acompanhados de uma devida análise.

4.2.2.1 Métricas e estratégias de avaliação

Com o intuito de avaliar as recomendações geradas por cada modelo, foram calculadas dois tipos de métricas diferentes, comumente utilizadas na literatura para recomendação top-N (SHANI; GUNAWARDANA, 2011): métricas de acurácia, com o objetivo de medir

a capacidade do método em selecionar itens para compor a recomendação, e de ranqueamento, para compreender sua qualidade em ordenar os itens selecionados. Para medir a acurácia, foram utilizadas a Precisão, Revocação e F-Medida. Para o ranqueamento, foi calculado o Ganho Cumulativo Descontado Normalizado (NDCG).

A Precisão é uma métrica que quantifica a qualidade dos itens selecionados pelo método recomendador. Desta forma, calcula-se quantos dos N itens selecionados foram consumidos pelo usuário. Repetindo este procedimento para todos os usuários, é possível calcular uma Precisão geral através da média (SARWAR et al., 2000a). Por estar diretamente associada ao número de itens recomendados, isto é, ao valor de N, é tradicionalmente utilizado o termo Prec@N, que representa a Precisão do método para uma recomendação top-N. Assim, sendo $s(u,\ N)$ uma função de um recomendador, responsável por retornar um conjunto de N itens recomendados para o usuário u, e I_u o conjunto de itens consumidos pelo usuário, a Prec@N pode ser calculada pela Equação 4.1:

$$\text{Prec@}N\ = \frac{1}{|U|} \sum_{u \in U} \frac{|s(u,\ N) \cap I_u|}{N} \tag{4.1}$$

A Revocação é uma medida similar à Precisão, entretanto, a métrica busca quantificar quanto dos itens consumidos foram recomendados pelo sistema, também extraindo uma média por usuário. Assim como ocorre na Precisão, a Revocação é altamente influenciável pelo valor de N, sendo também comumente referida como Rec@N. Seu cálculo é feito pela Equação 4.2:

$$\text{Rec@}N\ = \frac{1}{|U|} \sum_{u \in U} \frac{|s(u,\ N) \cap I_u|}{|I_u|} \tag{4.2}$$

Devido a maneira como são calculadas, a Precisão e a Revocação possuem uma característica importante: normalmente, quanto maior o valor de N, menor será a Precisão e maior será a Revocação, visto que mais itens serão recomendados. Assim, a comparação através de um único valor de N pode ser prejudicial para a análise. Para contornar este problema, há duas práticas comumente empregadas, tendo a última sido utilizada neste experimento: o uso de Curvas de Precisão e Revocação, uma técnica de visualização na qual as métricas são calculadas para diversos valores de N e projetadas como um gráfico de progressão; e o cálculo da F-Medida (RIJSBERGEN, 1979), média harmônica entre a Precisão e a Revocação. Assim como as duas métricas, a F-Medida também está condicionada a N, sendo referenciada como F1@N. Seu cálculo é feito como mostrado na Equação 4.3:

$$\text{F1@}N\ = 2 \times \frac{\text{prec@}N\ \times \text{Rec@}N}{\text{prec@}N\ + \text{Rec@}N} \tag{4.3}$$

Todas as métricas mencionadas são utilizadas para medir quantitativamente a qualidade da seleção de itens para compôr a recomendação. Entretanto, nenhuma é capaz de avaliar a capacidade do método de ordenar os itens recomendados. Para tal, são utilizadas métricas de avaliação de ranqueamento, que normalmente penalizam recomendações cujos itens corretos (isto é, aqueles consumidos pelo usuário) não encontram-se entre os primeiros itens recomendados. Com o objetivo de aferir a capacidade dos métodos em não apenas recomendar itens relevantes mas também ordená-los, foi calculado o Ganho Cumulativo Descontado Normalizado (JÄRVELIN; KEKÄLÄINEN, 2002), que atribui um peso em crescimento logarítmico para beneficiar recomendações cuja ordem favorece os itens corretos. Assim como as demais métricas, é utilizado o termo NDCG@N para se referir à métrica em uma recomendação top-N específica.

Para calcular o NDCG@N, são necessárias três etapas. Na primeira, é calculado o Ganho Cumulativo Descontado (DCG@N) para a recomendação (Equação 4.4):

$$\text{DCG@}N \ = \frac{1}{|U|} \sum_{u \in U} \sum_{n=1}^{N} \frac{g(n,u)}{log_2(n+1)} \tag{4.4}$$

na qual, $g(n,\ u)$ é uma função responsável por verificar se o item recomendado na posição n foi consumido pelo usuário u (Equação 4.5):

$$g(n,\ u) = \begin{cases} 1, & \text{se } s(u\ , N)_n \in I_u \\ 0, & \text{caso contrário} \end{cases} \tag{4.5}$$

onde $s(u\ , N)$ é uma função responsável por gerar uma recomendação de N itens para o usuário u de forma ordenada.

Após o cálculo do DGC@N, é necessário calcular o Ganho Cumulativo Descontado Ideal (IDCG@N), ou seja, o valor alcançado se todos os possíveis itens corretos fossem recomendados na ordem apropriada (Equação 4.6):

$$\text{IDCG@}N \ = \frac{1}{|U|} \sum_{u \in U} \sum_{n=1}^{Z} \frac{1}{log_2(n+1)} \tag{4.6}$$

na qual, Z representa o valor mínimo entre o número de itens recomendados e o número de itens consumidos pelo usuário, isto é, $Z = min(N,\ |I_u|)$

Por fim, o NDCG@N pode ser calculado normalizando o DCG@N pelo valor máximo possível de ser alcançado, o IDCG@N (Equação 4.7):

$$\text{NDCG@}N \ = \frac{\text{DCG@}N}{\text{IDCG@}N} \tag{4.7}$$

Todas as métricas mencionadas, utilizadas para avaliar a recomendação top-N, variam no intervalo $[0, 1]$, com valores maiores indicando resultados superiores.

Após o cálculo das métricas, foi aplicado o teste estatístico não-paramétrico de Friedman sobre o NDCG@15 para verificar se há diferenças estatisticamente significativas entre os resultados obtidos (FRIEDMAN, 1937). O teste verifica se é possível rejeitar a hipótese nula de que todos os métodos são equivalentes. Isto é feito considerando a posição ocupada por cada método em um *ranking* construído de acordo com o desempenho na métrica escolhida por base de dados, de forma que métodos com resultados melhores figurem em posições iniciais no *ranking*. A hipótese nula é rejeitada se o valor crítico definido para um intervalo de confiança α é menor do que o valor obtido pela Equação 4.8:

$$X_F^2 = \frac{12D}{k(k+1)} \left[\sum_{m=0}^{k} Q_m^2 - \frac{k(k+1)^2}{4} \right] \tag{4.8}$$

onde D é o número de conjuntos de dados observados, k é o número de métodos comparados e Q_m é a média entre as posições no *ranking* do m-ésimo método.

Em seguida, no caso de rejeição da hipótese nula, foi executado o teste *post-hoc* de Bonferroni-Dunn para comparar par-a-par o modelo proposto e os demais (BONFERRONI, 1936; DUNN, 1961). O teste tem como objetivo verificar se um método alvo possui diferenças estatisticamente significativas com cada um dos demais métodos, o que é confirmado se a diferença entre suas posições médias no *ranking* é maior do que um intervalo de confiança, dado pela Equação 4.9:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6D}} \tag{4.9}$$

na qual q_α corresponde ao valor crítico para um intervalo de confiança α calculado seguindo a distribuição t de Student dividida por $\sqrt{2}$.

Maiores detalhes sobre os testes estatísticos podem ser encontrados no trabalho de Demšar (2006).

4.2.2.2 Métodos de recomendação e implementação

Para executar o experimento e avaliar o Interact2Vec (INT2V) em comparação com outros modelos, foram selecionados cinco métodos populares na literatura, abrangendo diferentes estratégias de recomendação: um método de vizinhança clássico, baseado em similaridade de item (KNN) (DESROSIERS; KARYPIS, 2011); a fatoração de matriz para *feedback* implícito de Hu, Koren e Volinsky (2008) (SVD); o *Bayesian Personalized Ranking*, de Rendle et al. (2009), também baseado em fatoração de matriz, mas otimizando

o ranqueamento (BPR); e os dois métodos de RVDs por rede neural Item2Vec (BARKAN; KOENIGSTEIN, 2016) (ITM2V) e User2Vec (GRBOVIC et al., 2015) (USR2V).

Todos os métodos foram implementados em linguagem de programação `Python3`. O KNN foi implementado através da biblioteca `Turicreate` (APPLE, 2019). O SVD e o BPR através da biblioteca `Implicit` (FREDERICKSON, 2017). A implementação do ITM2V e o USR2V foi feito através da adaptação da biblioteca para geração de RVDs textuais `Gensim` (ŘEHŮŘEK; SOJKA, 2010). Finalmente, o Interact2Vec foi implementado utilizando a biblioteca `Keras` (CHOLLET et al., 2015), baseada em `Tensorflow` (ABADI et al., 2015). Adicionalmente, toda a manipulação de dados e cálculo das métricas foi realizado através do uso das bibliotecas `Numpy` (HARRIS et al., 2020), `Scipy` (VIRTANEN et al., 2020) e `Scikit-learn` (PEDREGOSA et al., 2011).

4.2.2.3 Ajuste de parâmetros

Com o objetivo de encontrar a melhor combinação de valores para os hiperparâmetros de cada método, foi executada busca em grade. Para cada possível combinação de parâmetros, os modelos foram ajustados sobre a base de treinamento, e a combinação que alcançou o maior NDGC@15 foi utilizada no momento de gerar a recomendação final.

Para o KNN, não é necessário realizar ajuste de parâmetros no cenário de recomendação top-N, visto que o método apenas encontra os N vizinhos mais próximos e recomenda-os. Já para os métodos baseados em fatoração de matriz – SVD e BPR – variou-se o número de fatores latentes f descobertos pelo modelo (o tamanho da representação vetorial gerada) em $\{50, 100, 200\}$, e a taxa de regularização em $\{10^{-6}, 10^{-4}, 10^{-2}\}$.

Para o Item2Vec e o User2Vec, foram testados diferentes valores para os parâmetros apontados como mais impactantes por Caselles-Duprés, Lesaint e Royo-Letelier (2018). Em suas pesquisas, os autores realizaram uma série exaustiva de testes para descobrir os parâmetros mais importantes de serem ajustados em modelos de RVD para recomendação. Assim, variou-se o número C de iterações da rede em $\{50, 100, 200\}$, a taxa ρ de subamostragem de itens frequentes em $\{10^{-5}, 10^{-4}, 10^{-3}\}$ e o expoente γ da seleção negativa em $\{-1,\ -0,5,\ 0,5,\ 1\}$. Os demais parâmetros foram mantidos com seus valores padrões, comumente empregados na literatura (BARKAN; KOENIGSTEIN, 2016) e em bibliotecas de geração de RVDs (ŘEHŮŘEK; SOJKA, 2010), como recomendado por Caselles-Duprés, Lesaint e Royo-Letelier (2018) (Tabela 4). Os valores testados para os parâmetros mais influentes também foram selecionados com base nas conclusões dos pesquisadores.

Ao contrário dos demais métodos de RVD, não havia conhecimento sobre quais parâmetros exercem maior influência sobre o resultado do Interact2Vec. Adicionalmente, fazer um busca extensiva completa para ajustar todos os parâmetros para cada uma das bases seria computacionalmente inviável. Assim, foram selecionadas duas bases de dados para a realização de um estudo de impacto da seleção de parâmetros, que encontra-se

Parâmetro	**Símbolo**	**Valor padrão**
Taxa de aprendizado	α	0,25
Tamanho das RVDs	M	100
Quantidade de amostras negativas	$\|G\|$	5

Tabela 4 – Valores padrões utilizados na literatura para os parâmetros não ajustados dos modelos Item2Vec e User2Vec

presente no Apêndice A.

Neste estudo, constatou-se que para os parâmetros taxa de aprendizado (α), quantidade de iterações da rede (C), taxa de subamostragem de itens frequentes (ρ) e fator de regularização (λ), existem valores que tendem a apresentar melhores resultados. Já para o tamanho das RVDs (M), quantidade de amostras negativas ($|G|$) e, principalmente, expoente da distribuição negativa (γ), não há um valor comum para alcançar resultados melhores, variando significativamente de acordo com a base de dados. Desta forma, variou-se M em $\{50, 100, 150\}$, $|G|$ em $\{5, 10, 15\}$ e γ em $\{-1,\ -0,5,\ 0,5,\ 1\}$ Para os demais métodos, foram associados os valores ideais encontrados, apresentados na Tabela 5.

Parâmetro	**Símbolo**	**Valor padrão**
Taxa de aprendizado	α	0,25
Quantidade de iterações da rede	C	50
Taxa de subamostragem de itens frequentes	ρ	10^{-6}
Fator de regularização	λ	0,1

Tabela 5 – Valores ideais para os parâmetros do Interact2Vec

Em adição aos parâmetros do modelo de geração das RVDs, é necessário também a seleção e ajuste de parâmetros do método de recomendação, ou seja, o algoritmo que consumirá as RVDs aprendidas para selecionar os N itens que irão compôr a recomendação.

As técnicas utilizadas para o Item2Vec e o User2Vec, respectivamente, similaridade item-item (Seção 3.3.1.2) e similaridade usuário-item (Seção 3.3.1.1), não possuem parâmetros ajustáveis, evitando assim a necessidade de otimização de parâmetros para esta etapa. Para o Interact2Vec, foram executados todos os métodos de recomendação apresentados na Seção 3.3, comportando-se como mais um parâmetro a ser escolhido sobre a base de validação. Como os métodos de similaridades ponderadas, combinação de RVDs e comitê de métodos possuem parâmetros ajustáveis, foi realizada uma busca em grade avaliando todas as combinações dos valores apresentados na Tabela 6.

Dessa forma, com o término da etapa de ajuste de parâmetros, foi selecionado, para cada base de dados, um conjunto de valores para os parâmetros do modelo neural do Interact2Vec, um método de recomendação e – se necessário – um conjunto de valores para os parâmetros do método de recomendação.

Método de recomendação	**Valores avaliados**
Similaridades ponderadas	$\beta = \{0,1,\ 0,25,\ 0,5,\ 0,75,\ 0,9\}$ $\mu = \{0,1,\ 0,25,\ 0,5,\ 0,75,\ 0,9\}$
Combinação de RVDs	Combinação = {média, concatenação} $K = \{1, 5, 10, 15, \lvert U_i \rvert\}$
Comitê de métodos	uso de $w_m = \{\text{sim},\ \text{não}\}$ uso de $w_r = \{\text{sim},\ \text{não}\}$ $L = \{15,\ 30,\ 45\}$

Tabela 6 – Valores avaliados para os parâmetros dos métodos de recomendação utilizados para as RVDs do Interact2Vec

Os melhores valores para cada parâmetro, de cada método, por conjunto de dados, encontram-se disponibilizados no Apêndice B.

4.2.2.4 Resultados

As Tabelas 7 e 8 apresentam, respectivamente, a F-Medida e o Ganho Cumulativo Descontado Normalizado alcançados por cada método em cada base de dados, para um cenário de recomendação top-15. As duas métricas variam entre 0 e 1, com valores maiores representando desempenhos melhores. Em ambas as tabelas, as células com os resultados encontram-se em escala de cinza, sendo que células com tons mais escuros representam resultados superiores. A coloração dos resultados de cada conjunto de dados é independente e não está relacionada com as demais.

Conjuntos de Dados	**KNN**	**SVD**	**BPR**	**ITM2V**	**USR2V**	**INT2V**
Anime	0,1326	0,1338	0,0897	0,0877	0,0072	0,0752
BestBuy	0,0336	0,0138	0,0161	0,0170	0,0049	0,0127
Book-Crossing	0,0074	0,0097	0,0056	0,0078	0,0012	0,0043
CiaoDVD	0,0110	0,0152	0,0119	0,0120	0,0031	0,0192
DeliciousBookmarks	0,0603	0,0230	0,0195	0,0393	0,0129	0,0544
Filmtrust	0,2680	0,1103	0,2411	0,2779	0,1270	0,2697
Last.FM	0,1203	0,0939	0,0784	0,1047	0,0157	0,1040
MovieLens	0,1597	0,1751	0,1095	0,0975	0,0003	0,0593
NetflixPrize	0,0566	0,0420	0,0314	0,0537	0,0007	0,0291
RetailRocket	0,0224	0,0225	0,0020	0,0184	0,0062	0,0169

Tabela 7 – Valores de F1@15 obtidos por cada método em cada base de dados

Observando as duas tabelas em conjunto, é possível perceber que os resultados alcançados são bastante similares quando os métodos são comparados entre si, isto é, embora o valor absoluto das métricas seja diferente, métodos que apresentaram bons resultados para a F-Medida fizeram o mesmo para o NDCG. O resultado é esperado, visto que o NDCG é uma medida avaliativa que afere não apenas a qualidade do ordenamento,

Conjuntos de Dados	KNN	SVD	BPR	ITM2V	USR2V	INT2V
Anime	0,2123	0,2374	0,1736	0,1275	0,0084	0,1267
BestBuy	0,1416	0,0633	0,0746	0,0557	0,0160	0,0435
Book-Crossing	0,0150	0,0173	0,0098	0,0131	0,0014	0,0076
CiaoDVD	0,0308	0,0408	0,0295	0,0277	0,0061	0,0457
DeliciousBookmarks	0,1594	0,0548	0,0467	0,0969	0,0235	0,1615
Filmtrust	0,5588	0,2385	0,5574	0,5721	0,1948	0,5663
Last.FM	0,2216	0,1864	0,1597	0,1894	0,0229	0,1737
MovieLens	0,2157	0,2727	0,1683	0,1211	0,0004	0,0683
NetflixPrize	0,0690	0,0578	0,0434	0,0645	0,0014	0,0357
RetailRocket	0,1100	0,1230	0,0025	0,0829	0,0133	0,0663

Tabela 8 – Valores de NDCG@15 obtidos por cada método em cada base de dados

mas também a qualidade da seleção, objetivo da F-Medida. Por esta semelhança entre os resultados, e particularidades de avaliação adicionais do NDCG, ele foi a métrica principal utilizada para analisar os métodos.

Há dois pontos de fácil observação em relação aos resultados alcançados: a superioridade do KNN e a inferioridade do User2Vec. De acordo com o NDCG@15, o KNN figura como o melhor ou segundo melhor método em oito das dez bases de dados, sendo o melhor em três delas. O resultado pode ser explicado pelo fato de o KNN ser o único que não realiza nenhuma técnica de redução de dimensionalidade, o que pode gerar perda de informação. Ao calcular as similaridades com as representações completas, isto é, os vetores de interações dos itens ($\vec{r_i}$), o método tem conhecimento de todas as interações realizadas, construindo uma vizinhança bastante confiável para cada item. Entretanto, exatamente por utilizar a representação completa, o KNN é um método que pode se tornar bastante ineficiente em bases muito volumosas, exigindo um elevado poder computacional para cálculo das similaridades. Como a tendência das bases de dados em sistemas de recomendação é sempre crescer, constantemente incluindo novos itens e usuários, os demais métodos baseados em redução de dimensionalidade tornam-se bastante atrativos.

Entre eles, é possível observar uma leve superioridade do SVD, alcançando F-Medida e NDCG maiores em uma boa parte das bases selecionadas. Ainda assim, o método apresentou resultados baixos para certos conjuntos de dados, como o DeliciousBookmarks e o Filmtrust. Superando o SVD em 40% das bases de dados, o Item2Vec apresentou desempenho constante, sendo o segundo pior método apenas em uma base, CiaoDVD. Nas demais, o modelo de RVD atingiu resultados competitivos com os demais métodos avaliados. O User2Vec, como já mencionado, apresentou resultados aquém do esperado. O modelo foi o que obteve a menor F-Medida e NDCG em 9 dos 10 conjuntos de dados, sendo melhor apenas que o BPR na base RetailRocket. Por aplicar, no momento do aprendizado, uma agregação através da média entre RVDs de itens e usuários, é possível que muita

informação seja perdida pelo modelo, o que explicaria seu desempenho.

O Interact2Vec, modelo proposto nesta dissertação, apresentou resultados bastante próximos aos demais métodos comparados. Embora tenha obtido F-Medida e NDCG baixos para alguns casos, como BestBuy e MovieLens, o modelo foi o primeiro ou segundo melhor em 30% das bases. Essa diferença de comportamento dificulta a comparação do Interact2Vec com os demais métodos. Desta forma, foi conduzida uma análise estatística com o objetivo de verificar se há superioridade do Interact2Vec em relação a outros modelos.

Inicialmente, foi executado o teste não-paramétrico de Friedman para verificar se há diferenças significativas entre os métodos. Os resultados indicaram, com 95% de confiança, que os modelos são estatisticamente diferentes entre si ($X_r^2 = 28,51$). Dessa forma, foi executado o teste *post-hoc* de Bonferroni-Dunn para realizar uma comparação par-a-par entre o Interact2Vec e os demais métodos. O resultado encontra-se na Figura 9. Nela, cada método é representado por um círculo preto – com exceção do Interact2Vec, representado como um quadrado azul – posicionado no eixo y de acordo com sua posição média em um *ranking* construído com os resultados da Tabela 8. A diferença crítica calculada pelo teste ($CD = 2,15$) encontra-se representada como barras de erro vermelhas acima e abaixo de cada método.

É possível afirmar que há diferença estatisticamente significativa entre o Interact2Vec e outro método se a diferença entre seus *ranks* médios é superior à diferença crítica. Dessa forma, tem-se que o Interact2Vec é estatisticamente superior ao User2Vec, e não se pode afirmar que há diferença significativa entre ele e os demais métodos. Ainda assim, sua posição média no *ranking* foi superior a do método BPR, indicando uma possível superioridade.

Em seguida, o mesmo procedimento foi realizado apenas com os métodos de RVD neurais. Novamente, o teste não-paramétrico de Friedman indicou uma diferença estatisticamente relevante entre os métodos ($X_r^2 = 16,80$), permitindo a aplicação do teste de Bonferroni-Dunn, cujo resultado encontra-se na Figura 10.

A aplicação dos testes apenas sobre os modelos de RVD não gerou resultados diferentes dos obtidos com a comparação entre todos os modelos. É possível afirmar que o Interact2Vec é estatisticamente superior ao User2Vec, com uma diferença crítica de $1,00$, e que não há indícios estatísticos de superioridade do Interact2Vec em relação ao Item2Vec.

Embora o método proposto não apresente o melhor F1@15 ou NDCG@15 para todas as bases, alcançando a quarta posição no *ranking* médio do NDCG, não há evidência estatística que seus resultados são inferiores aos demais métodos. Em adição, o Interact2Vec é um dos métodos mais computacionalmente eficientes entre os comparados, possuindo uma complexidade computacional linear. O único outro método que possui esta característica

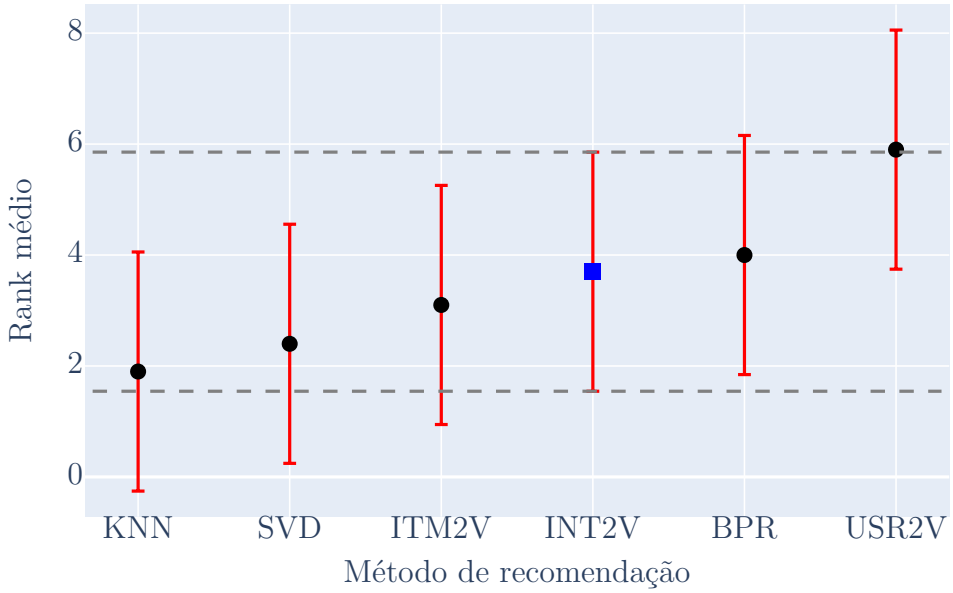

Figura 9 – Representação gráfica do teste *post-hoc* de Bonferroni-Dunn para ranqueamento top-N, comparando todos os métodos

é o User2Vec, cujos resultados se mostraram inferiores aos do modelo proposto. Tem-se então um método de baixa complexidade computacional, capaz de aprender RVDs de itens e usuários simultaneamente durante o treinamento, e que não sofre perdas significativas na qualidade da recomendação final. Portanto, o método proposto é uma boa opção, principalmente em cenários onde existam restrições computacionais (como escassez de recursos ou necessidade de agilidade no treinamento) e que produz diferentes tipos de representações, podendo ser utilizadas em outras tarefas.

4.2.3 Previsão de nota

A tarefa de previsão de nota foi uma das primeiras a ser abordada dentro da área da recomendação (SARWAR et al., 2001), ganhando uma enorme popularidade dentro da literatura após ser o método avaliativo adotado para o Netflix Prize (BENNETT; LANNING, 2007). Nela, os métodos de recomendação devem aprender com um histórico de *feedback* explícito entre usuários e itens, ou seja, notas atribuídas a itens por usuários, para que consiga prever as notas ainda não existentes. Dessa forma, o sistema deve ser capaz de, dado um par usuário-item, estimar que avaliação que o usuário alvo atribuiria ao item.

Para avaliar o problema de previsão de nota, são reservadas interações já existentes para que não sejam consumidas pelo modelo durante a etapa de aprendizado. Com o modelo treinado, é requisitado que o mesmo preveja estas interações, de forma que seja

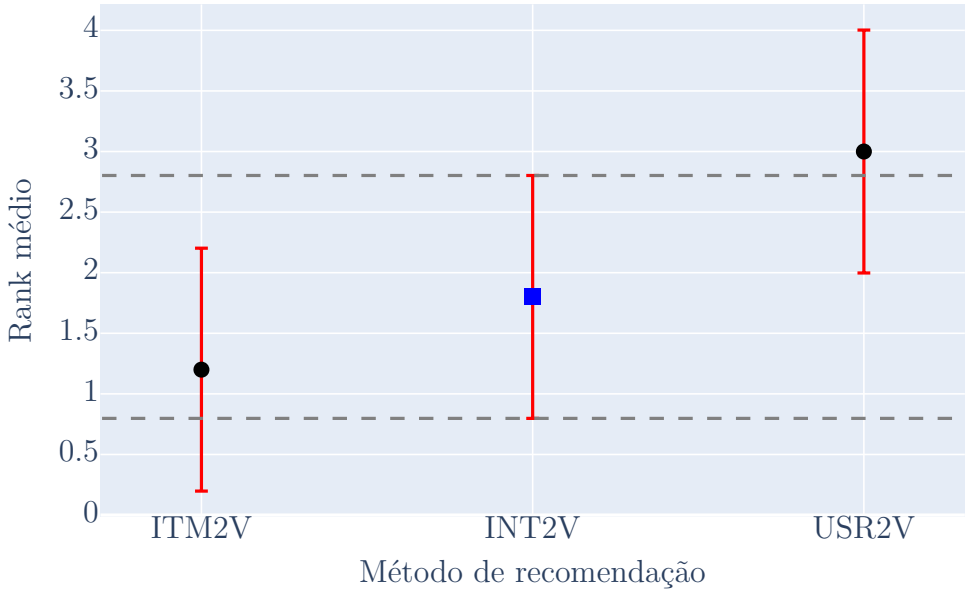

Figura 10 – Representação gráfica do teste *post-hoc* de Bonferroni-Dunn para ranqueamento top-N, comparando apenas os métodos de RVD

possível comparar a nota real atribuída pelo usuário com a estimada pelo sistema. Através de medidas de erro, a qualidade dos métodos pode ser mensurada, como mostrado na seção seguinte.

Os métodos foram treinados sobre a base de treinamento, composta por interações explícitas entre usuários e itens, para que pudesse prever as interações reservadas nas bases de validação e teste. Nas próximas seções, são apresentadas as medidas de avaliação, os métodos comparados, os detalhes para ajuste de parâmetros e os resultados finais obtidos.

4.2.3.1 Métricas e estratégias de avaliação

Para avaliar a tarefa de previsão de nota, são tradicionalmente adotadas medidas de erro (SHANI; GUNAWARDANA, 2011). Nelas, o valor previsto é comparado com o valor real, ou seja, a nota gerada pelo sistema é comparada com a nota atribuída pelo usuário, e a métrica é calculada de acordo com a diferença entre elas. Medidas de erro normalmente apresentam um valor que varia no intervalo $[0, \infty]$, com 0 sendo o melhor resultado possível, indicando que não há diferença entre o valor real e o previsto.

A avaliação das previsões geradas pelos métodos foi feita através de duas métricas de erro: o Erro Médio Absoluto (MAE) e a Raiz do Erro Quadrático Médio (RMSE). A primeira caracteriza-se como a média das diferenças absolutas entre cada par de valor previsto e real, calculado pela Equação 4.10. Já a segunda é calculada através da raiz quadrada da média das diferenças entre os valores reais e previstos, elevadas ao quadrado,

como é mostrado na Equação 4.11. Antes da aplicação de ambas as métricas, as notas reais e previstas foram transformadas por meio de normalização *min-max*, utilizando a menor e a maior nota possível como valores extremos ($R_{\min}$ e $R_{\max}$). Dessa forma, todas as métricas, independente da base de dados, passaram a ter a mesma escala, permitindo a comparação entre bases.

$$\text{MAE} = \frac{1}{|R^*|} \sum_{u,i \in R^*} |\hat{r}_{ui} - r_{ui}| \tag{4.10}$$

$$\text{RMSE} = \sqrt{\sum_{u,i \in R^*} \frac{(\hat{r}_{ui} - r_{ui})^2}{|R^*|}} \tag{4.11}$$

Nas equações, R^* representa o conjunto de pares usuário-item (u, i) que devem receber uma predição de nota, $\hat{r}_{ui}$ contém a nota prevista e r_{ui} a real.

Após o cálculo das métricas, foi aplicado o teste estatístico não-paramétrico de Friedman para verificar se há diferenças estatisticamente significativas entre os métodos de recomendação, seguido pelo teste *post-hoc* de Bonferroni-Dunn para comparação do modelo proposto com os demais, da mesma maneira que realizado no experimento de ranqueamento top-N, apresentado na Seção 4.2.2.1.

4.2.3.2 Métodos de recomendação e implementação

Em adição ao Interact2Vec (INT2V), foram executados cinco métodos de previsão de nota para fins de comparação e análise. A seleção dos métodos se deu com base na popularidade na literatura e disponibilidade. São eles: o K-vizinhos mais próximos orientado a item (KNN) (HERLOCKER et al., 2004); um método tradicional de fatoração de matriz utilizando decomposição de valores singulares (SVD) (FUNK, 2016; SALAKHUTDINOV; MNIH, 2007); o método de fatoração estado-da-arte, Factorization Machines (FM) (RENDLE, 2010); e os dois métodos de RVD Item2Vec (ITM2V) (BARKAN; KOENIGSTEIN, 2016) e User2Vec (USR2V) (GRBOVIC et al., 2015). Para os três métodos de RVD, não destinados para recomendação com *feedback* explícito, foi utilizada a técnica de K-vizinhos mais próximos para recomendação (Seção 3.3.2.1), consumindo apenas a representação vetorial aprendida para os itens.

Assim como na tarefa de ranqueamento top-N, todos os métodos e procedimentos foram implementados em `Python3`. O KNN foi executado através de implementação própria, utilizando as bibliotecas `Numpy` (HARRIS et al., 2020) e `Scipy` (VIRTANEN et al., 2020). O SVD foi implementado pela biblioteca `Surprise` (HUG, 2017). A implementação do FM foi realizada através da biblioteca `Turicreate` (APPLE, 2019). Finalmente, para os três métodos de RVD, ITM2V, USR2V e INT2V, as implementações foram realizadas utilizando

as mesmas bibliotecas, ou seja, `Gensim` (ŘEHŮŘEK; SOJKA, 2010) e `Keras` (CHOLLET et al., 2015), respectivamente.

4.2.3.3 Ajuste de parâmetros

Para encontrar a combinação de valores mais apropriada para os parâmetros de cada método, foi executada uma etapa de ajuste de parâmetros por meio de busca em grade. Para cada método, foram avaliadas diferentes combinações, sendo escolhida aquela que obteve o maior RMSE.

Para o KNN, foram testados diferentes valores para K (quantidade de itens selecionados para estimar a nota que o usuário atribuiria ao item alvo), variando-o em $\{20, 40, 60, 80, 100\}$. Para os métodos de fatoração de matriz, SVD e FM, variou-se o parâmetro de regularização λ em $\{10^{-6}, 10^{-4}, 10^{-2}\}$ e o número de fatores latentes f em $\{50, 100, 200\}$.

Os métodos de RVD tiveram seus parâmetros ajustados da mesma maneira que realizado nos experimentos de ranqueamento top-N, como descrito na Seção 4.2.2.3. Entretanto, diferente da outra tarefa, foi executado apenas um algoritmo para consumir as representações vetoriais e gerar a previsão: o KNN, explicado na Seção 3.3.2.1. Assim, para cada modelo de RVD, otimizou-se o valor de K, variando em $\{20, 40, 60, 80, 100\}$.

Os melhores valores para cada parâmetro, de cada método, por conjunto de dados, encontram-se disponibilizados no Apêndice B.

4.2.3.4 Resultados

Para aferir a qualidade dos métodos em prever notas para pares usuário-item, foram calculadas as métricas MAE e RMSE. Os resultados encontram-se, respectivamente, nas Tabelas 9 e 10. As células encontram-se em escala de cinza, de forma que resultados superiores (isto é, que apresentam um erro menor) estejam coloridos com tonalidades mais escuras.

Conjunto	KNN	SVD	FM	ITM2V	USR2V	INT2V
Anime	0,1042	0,1023	0,0949	0,1077	0,1338	0,1282
Book-Crossing	0,1508	0,1426	0,1361	0,1506	0,1543	0,1618
CiaoDVD	0,1957	0,1850	0,1854	0,1964	0,1959	0,1920
Filmtrust	0,1785	0,1762	0,1755	0,1806	0,1786	0,1813
MovieLens	0,1444	0,1457	0,1488	0,1529	0,1657	0,1948
NetflixPrize	0,1938	0,1832	0,1900	0,2043	0,2039	0,2066

Tabela 9 – Valores de MAE obtidos por cada método em cada base de dados

Ambas as métricas calculadas apresentaram resultados semelhantes para comparação entre métodos. Tanto para o MAE quanto para o RMSE, o método de factorization

Conjunto	KNN	SVD	FM	ITM2V	USR2V	INT2V
Anime	0,1400	0,1350	0,1260	0,1437	0,1771	0,1680
Book-Crossing	0,2005	0,1838	0,1778	0,1998	0,2019	0,2061
CiaoDVD	0,2673	0,2400	0,2390	0,2666	0,2630	0,2608
Filmtrust	0,2362	0,2265	0,2244	0,2360	0,2324	0,2369
MovieLens	0,1951	0,1909	0,1945	0,2015	0,2224	0,2612
NetflixPrize	0,2613	0,2337	0,2390	0,2716	0,2616	0,2590

Tabela 10 – Valores de RMSE obtidos por cada método em cada base de dados

machines (FM) apresentou os melhores resultados, sendo o melhor método em metade das bases para o MAE e 66% das bases para o RMSE. O SVD foi outro método que apresentou bons resultados, ficando em primeiro lugar em todas as bases cuja a posição não foi ocupada pelo FM (com exeção do MAE para a base MovieLens, na qual o método KNN apresentou o menor erro). Isto reforça a classificação dos métodos FM e SVD como estado-da-arte para a tarefa de previsão de notas.

Os três métodos de RVD apresentaram resultados inferiores aos demais métodos, para ambas as medidas. Em todos os casos, os modelos de RVD obtiveram desempenho pior que os métodos de fatoração de matriz. Apenas em comparação ao KNN que os modelos atingiram erros mais competitivos, como o Item2Vec na base Book-Crossing, o Interact2Vec na base CiaoDVD, o User2Vec na base Filmtrust, entre outros exemplos.

É interessante observar como os métodos apresentaram comportamentos diferentes em relação aos resultados apresentados na tarefa de ranqueamento top-N. Nela, o Interact2Vec, por exemplo, apresentou um NDCG@15 baixo na base NetflixPrize, obtendo resultados melhores para as bases CiaoDVD e Filmtrust. Para a previsão de notas, o modelo só manteve o mesmo padrão de resultados para a base CiaoDVD, sendo o terceiro método com menor erro para a NetflixPrize e o método com maior erro para a Filmtrust. Isso evidencia como o ranqueamento top-N e a previsão de notas são duas tarefas bastante distintas, e métodos com bom desempenho em uma não necessariamente apresentarão o mesmo comportamento na outra.

Como esperado, os três métodos de RVD apresentaram resultados inferiores aos métodos de vizinhança e fatoração matricial. Por não consumirem o *feedback* explícito dos usuários no momento do aprendizado, as representações vetoriais geradas não carregam significado em relação às suas preferências ou rejeições, possuindo assim um conhecimento menor sobre o cenário do problema.

Para comparar os métodos e verific ar se há realmente diferença estatisticamente significativa entre os desempenhos obtidos, foi conduzido um teste não-paramétrico de Friedman com a hipótese nula de que os métodos não apresentam diferença significativa. O resultado do teste indicou que há diferença estatisticamente significativa ($X_r^2 = 21,14$).

Desta forma, foi aplicado o teste *post-hoc* de Bonferroni-Dunn para comparar o modelo proposto com os demais (Figura 11).

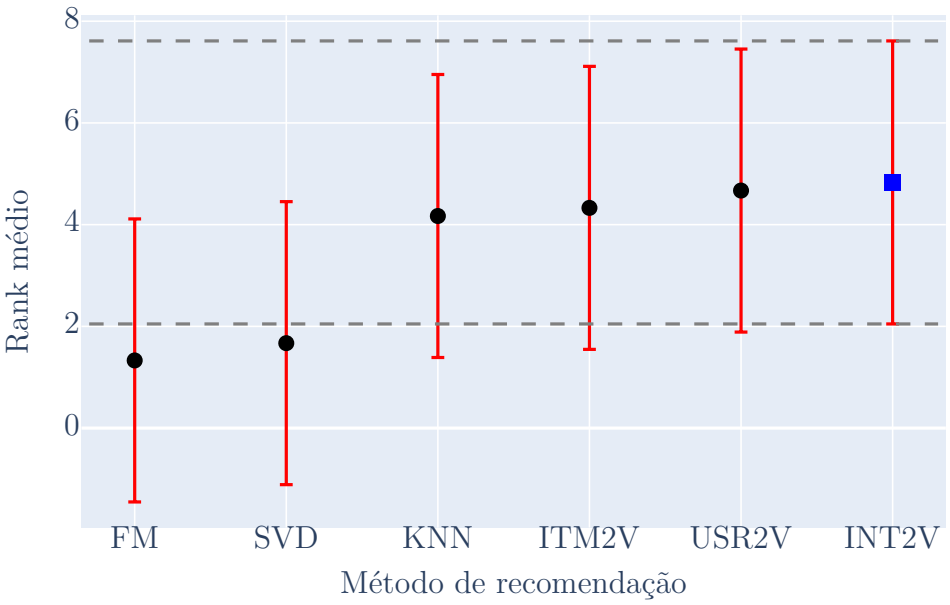

Figura 11 – Representação gráfica do teste *post-hoc* de Bonferroni-Dunn para previsão de notas, comparando todos os métodos

O Interact2Vec foi o modelo que apresentou o pior *rank* médio, com um valor bastante próximo ao User2Vec e o Item2Vec. Com uma confiança de 95% e diferença crítica de $2,78$, pode-se dizer que o modelo é estatisticamente inferior ao FM e SVD, e não apresenta superioridade estatística a nenhum dos outros modelos. Desta forma, é conclusivo que o Interact2Vec, assim como os demais métodos de RVD, não são recomendados para a tarefa de previsão de notas, havendo na literatura métodos mais apropriados para o problema.

Para comparar apenas os métodos de RVD, foi realizado o teste de Friedman nos resultados do Item2Vec, User2Vec e Interact2Vec. Entretanto, a hipótese nula não foi rejeitada, indicando que não há diferença estatisticamente significativa entre os três modelos de representação vetorial ($X_r^2 = 0,33$). Neste cenário, o Interact2Vec e o User2Vec apresentam a vantagem de possuírem menor complexidade computacional que o Item2Vec.

4.3 Avaliação intrínseca

Uma das características mais interessantes do uso de RVDs na área de processamento de linguagem natural, é a vantagem que representações vetoriais distribuídas apresentam de carregar significado semântico. Assim, palavras ou frases que possuem significados semelhantes tendem a ser representadas de maneira próxima no espaço vetorial (MIKOLOV

et al., 2013). Tal característica popularizou o uso de RVDs em diversas tarefas da área que podem se aproveitar diretamente do significado semântico das representações, como tradução (MIKOLOV; LE; SUTSKEVER, 2013) e geração de texto (RIZOS; HEMKER; SCHULLER, 2019).

De forma mais abrangente, é possível dizer que RVDs são representações que carregam significado intrínseco. Na área da recomendação, isto pode se referir às características de itens e usuários, como metadados e informações demográficas, e muito pode ser realizado com representações de baixa dimensionalidade que carreguem este significado. Tarefas como a previsão automática de rótulos descritivos para itens, o agrupamento de itens ou usuários de acordo com seu conteúdo e a filtragem de itens com determinados atributos (LU et al., 2015), são exemplos de aplicações que podem ser enriquecidas com o uso de RVDs com valor intrínseco.

Ainda que existam muitas vantagens e oportunidades de pesquisa sobre o assunto, não é uma prática comum em pesquisas sobre RVDs para recomendação conduzir uma análise ampla sobre os modelos de representação propostos. Tradicionalmente, a avaliação das representações é feita apenas sobre a recomendação final, através de uma análise extrínseca. Entretanto, um bom desempenho no problema da recomendação não necessariamente garante um bom conteúdo intrínseco para as RVDs (SCHNABEL et al., 2000).

Para avaliar a capacidade dos métodos de representação em carregar conteúdo intrínseco, foi conduzido um protocolo de avaliação intrínseca inédito sobre os modelos executados na tarefa de ranqueamento top-N, incluindo o modelo proposto. Para uma comparação justa com o Interact2Vec e os demais métodos de RVD, os modelos executados na tarefa de previsão de nota, que consomem *feedback* explícito, não foram selecionados. O KNN também não foi incluído na avaliação, visto que é um método que não aprende representações vetoriais para itens e usuários.

Foram executados quatro esquemas de avaliação diferentes: tabelas de similaridade, detecção de intruso, descoberta de rótulos e através de uma nova métrica proposta, que quantifica a similaridade das RVDs com a representação baseada em conteúdo. Os esquemas foram divididos de acordo com a maneira como são realizados. Os dois primeiros baseiam-se em opiniões humanas, enquanto os demais são calculados de forma automática. A explicação de cada formato de avaliação, assim como os resultados, encontram-se nas seções seguintes.

4.3.1 Métricas baseadas em opiniões humanas

Uma das formas mais confiáveis de se medir a qualidade intrínseca da representação vetorial é através de *feedback* humano. Muitas vezes, a opinião humana tem conhecimento adicional em relação aos itens representados, podendo assim avaliar de forma mais realista

a qualidade da representação.

Embora possua certas deficiências, discutidas em seções mais adiante, o uso de métricas que dependem da opinião humana pode ser empregado para entender a qualidade da representação vetorial aprendida. Desta forma, foram conduzidos dois procedimentos avaliativos com o intuito de aferir a qualidade das RVDs com base na opinião humana. A primeira, bastante utilizada na área de recomendação, consiste em construir tabelas de similaridade, ou seja, dado um item alvo, verifica-se quais os itens mais próximos a ele no espaço vetorial aprendido, permitindo uma análise subjetiva sobre os resultados. A segunda, menos popular no contexto de recomendação, consiste em gerar um problema a ser respondido por seres humanos, e calcular um valor quantitativo para aferir a qualidade das representações. Cada uma das tarefas, assim como seu resultado, encontram-se apresentadas nas seções seguintes.

4.3.1.1 Tabelas de similaridade

Tabelas de similaridade são a estratégia de avaliação intrínseca mais utilizadas em pesquisas sobre RVDs para recomendação (BARKAN; KOENIGSTEIN, 2016; FU et al., 2017). A técnica caracteriza-se como a construção de uma tabela que evidencia ao leitor os itens mais semelhantes a um item alvo previamente selecionado, para cada um dos modelos de representação avaliados.

Para conduzir uma avaliação intrínseca do Interact2Vec e compará-lo com os demais métodos de representação vetorial, foram selecionados, de forma arbitrária, cinco itens popularmente conhecidos dentro das bases Last.FM e MovieLens. A escolha dos conjuntos de dados se deu por possuírem itens de domínios populares (música e filmes, respectivamente). Para cada um dos cinco itens selecionados, foram recuperados os três itens mais próximos a eles no espaço vetorial gerado por cada um dos métodos de recomendação. Os resultados para a base Last.FM encontram-se na Tabela 11, e para a base MovieLens na Tabela 12. Para cada artista ou filme selecionado, foram indicados seus gêneros musicais ou cinematográficos, possibilitando um maior conhecimento sobre os itens.

Embora a análise das tabelas seja altamente enviesada por opiniões humanas, pode-se dizer que, para a base Last.FM (Tabela 11), todos os modelos encontraram itens com gêneros similares aos alvos, com exceção apenas do SVD para a banda *The Beatles*. É difícil entretanto avaliar se há algum modelo que se sobressai aos demais, visto que, mesmo havendo bastante diferença entre os itens relacionados, todos possuem algum tipo de relação com o item alvo.

Com exceção do SVD, o User2Vec foi o modelo que mais recomendou artistas pouco conhecidos, tais como*Flyleaf* ou *VersaEmerge* para *Pink Floyd*, e *Heaven & Hell* ou *Metal Church* para *AC/DC*. Ainda assim, há uma concordância em relação ao gênero musical entre os artistas recomendados e o alvo, dificultando a comparação da qualidade

Alvo	**Modelo de representação**				
	SVD	**BPR**	**ITM2V**	**USR2V**	**INT2V**
Linkin Park *alt rock, rock*	medusa' Scream *emo*	30 Seconds to Mars *rock, alt rock*	Breaking Benjamin *rock, alt rock*	30 Seconds to Mars *rock, alt rock*	Marilyn Manson *metal, indust.*
	Grey Daze *alt rock, rock*	The Rasmus *rock, alt*	30 Seconds to Mars *rock, alt rock*	Flyleaf *alt rock, rock*	Nickelback *rock, hard rock*
	Dead by Sunrise *alt rock, rock*	Breaking Benjamin *rock, alt rock*	Evanescence *rock, female*	VersaEmerge *rock, female*	30 Seconds to Mars *rock, alt rock*
Shakira *pop, female*	Juanes *latin, pop*	Beyoncé *rnb, pop*	Rihanna *pop, rnb*	Katy Perry *pop, female*	Jennifer Lopez *pop, dance*
	Fanny Lu *latin pop*	Marilyn Monroe *jazz, female*	Beyoncé *rnb, pop*	Mariah Carey *rnb, pop*	Miley Cyrus *pop, female*
	Thalía *female, latin*	Rihanna *pop, rnb*	Britney Spears *pop, dance*	Beyoncé *rnb, pop*	Mariah Carey *rnb, pop*
AC/DC *classic rock* *hard rock*	Ozzy Osbourne *heavy metal*	Velvet Revolver *hard rock, alt*	Van Halen *hard rock, 80s*	Heaven & Hell *heavy metal*	Aerosmith *hard rock* *classic rock*
	Led Zeppelin *classic rock* *hard rock*	Guns N' Roses *hard rock, rock*	Metallica *metal* *thrash metal*	Metal Church *thrash metal*	Iron Maiden *heavy metal* *metal*
	Guns N' Roses *hard rock, rock*	Tenacious D *rock, hard rock*	Guns N' Roses *hard rock, rock*	Iommi *heavy metal* *metal*	Motörhead *heavy metal* *hard rock*
Eminem *rap, hip-hop*	Ice Cube *hip-hop, rap*	Jay-Z *hip-hop, rap*	Ke$ha *pop, dance*	Akon *hip-hop, rap*	Drake *hip-hop, rnb*
	Bizarre *hip-hop, rap*	50 Cent *rap, hip-hop*	P!nk *pop, female*	Nelly *rap, hip-hop*	Jay-Z *hip-hop, rap*
	Xzibit *hip-hop, rap*	Kanye West *hip-hop, rap*	Jay-Z *hip-hop, rap*	Jason Derulo *pop, rnb*	Black Eyed Peas *hip-hop, pop*
The Beatles *rock* *classic rock*	Ricky Nelson *rock, classic rock*	Beach Boys *60s* *classic rock*	David Bowie *rock* *classic rock*	The Kinks *60s* *classic rock*	Radiohead *alt, rock*
	Souad Massi *female, arabic*	John Lennon *classic rock* *rock*	Radiohead *alt, rock*	The Rolling Stones *classic rock*	Pink Floyd *prog rock* *classic rock*
	Andrés Segovia *baroque*	Ringo Starr *classic rock* *rock*	Led Zeppelin *classic rock* *hard rock*	The Velvet Underground *psychedelic*	Led Zeppelin *classic rock* *hard rock*

Tabela 11 – Tabela de similaridade para 5 artistas populares na base Last.FM

de sua representação. O SVD foi outro método que recomendou artistas com menor popularidade, gerando uma situação curiosa para a artista *Shakira*: assim como o alvo, todos os demais métodos recomendaram cantoras *pop* mundialmente famosas, como *Beyoncé* e *Mariah Carey*, possivelmente indicando uma melhor representação. Entretanto, todas as artistas recomendadas pelo SVD são cantoras *pop* de nacionalidade latina, possuindo uma característica adicional relacionada ao alvo que não é compartilhada pelas demais recomendações. Portanto é uma tarefa complexa avaliar a qualidade das representações, sendo extremamente atrelada aos conhecimentos de domínio do avaliador.

Para a base de filmes (Tabela 12), os resultados entre os métodos apresentaram maior discrepância. Os itens recomendados pelo Interact2Vec mostraram-se bastante diferentes em relação ao item alvo, como a escolha de filmes de ação para *Grease* (um musical romântico), ou produções de origem indiana para *Friday the 13th* (um terror estadunidense). Considerando o baixo desempenho do método para a base MovieLens na tarefa de ranqueamento top-N, o comportamento apresentado é esperado, e aumenta a evidência de que o modelo proposto não foi capaz de aprender uma boa representação para a base cinematográfica.

Observando os demais métodos, novamente torna-se difícil selecionar algum que tenha apresentado resultados superiores. Em muitos casos, a similaridade entre filmes não se dá com base em seus gêneros, mas sim por serem clássicos do cinema (como *Titanic* e *Men in Black* ou *Saving Private Ryan*). Assim, há argumentos favoráveis para quase todas as escolhas feitas pelos métodos. Outro ponto interessante de análise é como determinados métodos apresentaram resultados extremamente precisos para alguns filmes, e resultados questionáveis para outros, como é o caso do BPR: o método foi o único que recomendou três musicais para *Grease*, porém também recomendou um filme de terror (*Jack-O*) para *Toy Story* (animação infantil).

Embora seja uma das técnicas de análise intrínseca mais utilizadas na literatura, tabelas de similaridade podem gerar diversos problemas e dificuldade de interpretação, como mostrado nestes exemplos. Faz-se então interessante o uso de abordagens que gerem um valor quantitativo de avaliação, minimizando assim a interferência humana. Uma opção para isto é a aplicação da técnica de detecção de intruso, apresentada na seção seguinte.

4.3.1.2 Detecção de intruso

A detecção de intruso é um procedimento avaliativo proposto para a área de PLN por Schnabel et al. (2000). A técnica nunca foi utilizada no contexto de recomendação, embora seja de fácil aplicação e capaz de gerar resultados interessantes.

O procedimento é construído da seguinte maneira: para cada base de dados, é selecionada uma quantidade de itens-alvo, e para cada modelo de representação avaliado, é encontrado um conjunto de itens similares ao alvo no espaço vetorial, compondo assim um

Alvo	**Modelo de representação**				
	SVD	**BPR**	**ITM2V**	**USR2V**	**INT2V**
Toy Story *Children, Comedy*	Average Italian *Comedy*	Muppet Treasure Island *Children*	Braveheart *Drama, War*	Lion King *Children, Musical*	Star Wars IV *Adventure, Sci-Fi*
	The Pride and Passion *War*	Babe *Children, Drama*	12 Monkeys *Sci-Fi, Thriller*	Toy Story 2 *Children, Comedy*	LOTR II *Adventure, Fantasy*
	Barbie *Animation, Children*	Jack-O *Horror*	The Usual Suspects *Crime*	Men in Black *Action, Sci-Fi*	Sixth Sense *Drama, Horror*
Grease *Musical, Romance*	Dirty Dancing *Musical*	The Sound of Music *Musical*	Batman Returns *Action, Crime*	Top Gun *Action, Romance*	First Knight *Action Romance*
	Big *Comedy, Romance*	The Little Princess *Children*	Scream *Horror, Mystery*	Gremlins *Comedy Horror*	Deep Impact *Drama, Sci-Fi*
	The Little Mermaid *Musical*	Oliver! *Drama, Musical*	Air Force 1 *Action, Thriller*	Mary Poppins *Musical*	Sneakers *Action, Crime*
X-Men *Action, Sci-Fi*	Star Wars II *Adventure, Sci-Fi*	Monsters Inc. *Adventure, Children*	Toy Story 2 *Children, Comedy*	Star Wars II *Adventure, Sci-Fi*	Men in Black *Action, Sci-Fi*
	The Matrix 2 *Action, Sci-Fi*	X2: X-Men United *Action, Sci-Fi*	Gladiator *Action, Adventure*	X2: X-Men United *Action, Sci-Fi*	Avatar *Adventure Sci-Fi*
	Star Wars I *Adventure, Sci-Fi*	Spider-Man *Adventure, Sci-Fi*	Memento *Mystery, Thriller*	Gladiator *Action, Adventure*	Speed *Action, Thriller*
Titanic *Drama, Romance*	Groundhog Day *Comedy*	The Truman Show *Comedy*	Good Will Hunting *Drama*	Jurassic Park *Action, Sci-Fi*	The Usual Suspects *Crime*
	The Truman Show *Comedy*	Catch Me If You Can *Crime, Drama*	Men in Black *Action, Sci-Fi*	The Truman Show *Comedy*	Blade Runner *Action, Sci-Fi*
	Christmas Do-Over *Comedy*	Best Friends Wedding *Comedy*	Saving Private Ryan *Drama, War*	Men in Black *Action, Sci-Fi*	Indiana Jones I *Adventure*
Friday the 13th *Horror*	A View to a Kill *Action*	Nightmare on Elm Street 4 *Horror*	Gremlins 2 *Comedy Horror*	Friday the 13th Part 2 *Horror*	Jaal: The Trap -
	Child's Play *Horror, Thriller*	Friday the 13th Part 3 *Horror*	Texas Chainsaw Mass. *Horror*	Halloween II *Horror*	Calcutta Mail *Thriller*
	Pet Sematary *Horror*	Children of the Corn *Horror*	Halloween *Horror*	Child's Play *Horror, Thriller*	Saaya -

Tabela 12 – Tabela de similaridade para 5 filmes populares na base MovieLens

grupo de itens. Em seguida, são selecionados itens aleatórios ("intrusos") e inseridos em cada grupo. É realizada então uma consulta com indivíduos externos, para que os mesmos busquem acertar qual é o item "intruso". Com as respostas fornecidas, é possível mensurar a qualidade das representações, utilizando métricas como a acurácia, por exemplo.

A técnica foi aplicada sobre três conjuntos de dados: Anime, MovieLens e Last.FM, por serem domínios conhecidos ou bases com bastante informação de conteúdo. Para cada base, foram selecionados quinze itens-alvo, dos quais dez são popularmente conhecidos e cinco foram extraídos de forma aleatória. Em seguida, foram construídos cinco questionários diferentes, cada um com 15 grupos de itens, alternando entre os modelos de representação. Foi solicitado para que dez indivíduos tentassem solucionar a tarefa, informando qual item era um possível "intruso". Ao todo, cada modelo de representação obteve trinta votos. A acurácia para cada modelo foi calculada, e encontra-se nas Tabelas 13, 14 e 15, respectivamente: a acurácia geral, a acurácia apenas para os dez itens populares e apenas para os cinco itens aleatórios (e, provavelmente, desconhecidos). Todos os resultados encontram-se em escala de cinza, com tons mais escuros indicando maior acurácia.

Conjunto	SVD	BPR	ITM2V	USR2V	INT2V
Anime	43,33%	63,33%	60,0%	90,0%	56,67%
MovieLens	33,33%	66,67%	46,67%	43,33%	46,67%
Last.FM	66,67%	90,0%	86,67%	66,67%	60,0%

Tabela 13 – Acurácia para a tarefa de detecção de intruso (todos os itens)

Conjunto	SVD	BPR	ITM2V	USR2V	INT2V
Anime	44,44%	66,67%	55,56%	83,33%	83,33%
MovieLens	38,89%	61,11%	44,44%	44,44%	55,56%
Last.FM	72,22%	94,44%	88,89%	77,78%	72,22%

Tabela 14 – Acurácia para a tarefa de detecção de intruso (apenas itens populares)

Conjunto	SVD	BPR	ITM2V	USR2V	INT2V
Anime	41,67%	58,33%	66,67%	100,0%	16,67%
MovieLens	25,0%	75,0%	50,0%	41,67%	33,33%
Last.FM	58,33%	83,33%	83,33%	50,0%	41,67%

Tabela 15 – Acurácia para a tarefa de detecção de intruso (itens aleatórios)

Embora não haja um consenso entre todas as bases referente a um melhor método, o BPR foi aquele que apresentou melhores resultados quando considerado todos os itens-alvo (Tabela 13), sendo o mais preciso para a base MovieLens e Last.FM, e alcançando o segundo melhor para a base Anime, na qual o User2Vec apresentou os melhores resultados. O User2Vec não apenas apresentou bons resultados para a base Anime, como atingiu

100% de acurácia no cenário onde itens foram selecionados aleatoriamente (Tabela 15), mostrando que foi capaz de gerar uma vizinhança relevante mesmo em casos onde há pouco conhecimento sobre o item.

O Interact2Vec apresentou resultados medianos na Tabela 13, aproximando-se do Item2Vec e alcançando acurácias superiores ao SVD (com exceção da base Last.FM). Entretanto, os resultados do modelo proposto são interessantes quando compara-se as Tabelas 14 e 15. Na primeira, na qual apenas itens populares foram considerados, o método apresentou valores altos de acurácia, empatando com o User2Vec na base Anime e tendo um aumento de 10% de acurácia na base MovieLens (o maior entre os métodos). Todavia, o modelo teve uma grande queda de qualidade na segunda tabela, cenário no qual apenas itens aleatórios e pouco conhecidos foram considerados. Isso indica que o Interact2Vec possui um alto poder de aprender informação relevante sobre itens bastante consumidos, porém não mantém o mesmo comportamento para itens menos populares. O resultado pode ser explicado devido à maneira como é realizado seu aprendizado. Por gerar as recomendações com bases em pares usuário-item, o modelo beneficia-se de casos onde um mesmo item foi consumido por diversos usuários, aparecendo assim mais vezes como entrada para a rede neural.

Em relação a tarefa de detecção de intruso, há fortes evidências de que o BPR é o melhor método para aprender informações intrínsecas. Entretanto, este procedimento avaliativo, embora gere uma métrica quantitativa, ainda é bastante influenciado pela opinião humana. Foi reportado pelos indivíduos consultados que, em muitos casos onde não houve consenso sobre um item "intruso", a escolha foi feita de forma aleatória. Isto pode distorcer os resultados, evidenciando a necessidade do uso de técnicas automáticas de avaliação.

4.3.2 Métricas automáticas

Abordagens dependentes de *feedback* humano são importantes para o entendimento da qualidade intrínseca das representações, visto que refletem da maneira mais próxima a opinião que um usuário real do sistema possui. Entretanto, tais abordagens possuem pelo menos três desvantagens: *(i)* são fortemente influenciadas por viés humano, de forma que os resultados para um usuário podem ser discordados por outro; *(ii)* demandam bastante trabalho e tempo, visto que é necessário que usuários reajam às representações geradas, incluindo um fator externo à avaliação; e *(iii)* podem não ser totalmente confiáveis, dado que é praticamente impossível que uma pessoa gere *feedback* sobre uma larga quantidade de itens representados, fazendo com que todas as análises sejam feitas sobre resultados que pouco representam a totalidade do sistema.

Para aliviar estes problemas, é pertinente o uso de métricas de avaliação automáticas, ou seja, medidas calculadas sem interferência humana. Foram então conduzidos dois

procedimentos automáticos de avaliação: a classificação de categorias para os itens; e uma nova métrica proposta, calculada através da comparação entre a representação aprendida e uma representação baseada em conteúdo. A explicação de cada técnica e seus resultados encontram-se nas seções seguintes.

4.3.2.1 Classificação de categorias e *tags*

A tarefa de descobrir automaticamente atributos de itens, como categorias ou *tags*, é um problema específico dentro da área de sistemas de recomendação, com muitos métodos propostos exclusivamente para isso (SONG; ZHANG; GILES, 2011). Também conhecida como *auto-tagging*, uma das maneiras mais fáceis de ser aplicada consiste no seguinte: para cada item no catálogo (ou um subconjunto deles), é selecionado K vizinhos mais próximos de acordo com a representação avaliada. Em seguida, observando os vizinhos, os atributos do item-alvo são descobertos por meio de alguma estratégia de classificação, como uma simples votação. Com isso, é possível calcular métricas de avaliação, que comparam as categorias previstas em relação às reais (BARKAN; KOENIGSTEIN, 2016; FU et al., 2017).

Embora predizer atributos de itens possa não ser totalmente confiável, dado que poderá haver informações falsas ou ruidosas (especialmente em cenários onde tais atributos são informados por humanos), na maioria dos casos, essa será a única informação disponível para descrever o conteúdo intrínseco do item.

Foi então realizada uma tarefa de *auto-tagging* utilizando os modelos de representação comparados. Para cada item pertencente à base, foram selecionados os cinco itens mais próximos ($K = 5$), de acordo com a representação vetorial avaliada, e uma categoria, gênero ou *tag* foi prevista através do voto majoritário entre os vizinhos. Sobre a previsão, foi calculada a Precisão, Revocação e F-Medida, apresentada na Tabela 16.

Conjunto	**SVD**	**BPR**	**ITM2V**	**USR2V**	**INT2V**
Anime	0,3600	0,4405	0,4968	0,4903	0,2583
BestBuy	0,1057	0,1390	0,3099	0,3468	0,0205
DeliciousBookmarks	0,1357	0,1245	0,1308	0,1342	0,0658
Last.FM	0,4444	0,3656	0,3851	0,4367	0,2679
MovieLens	0,3863	0,4297	0,4398	0,3858	0,1562

Tabela 16 – F-Medida para a tarefa de classificação automática de categorias e *tags*

Os resultados foram bastante desfavoráveis para o Interact2Vec, o que é curioso, dado que o método alcançou bons resultados na tarefa de recomendação. Adicionalmente, o User2Vec, que obteve o pior desempenho para a recomendação, figurou como um dos melhores modelos para a tarefa de predição automática de *tags*, sendo o primeiro ou segundo melhor em 80% dos conjuntos de dados.

Pode-se concluir sobre os resultados obtidos que, assim como evidenciado por Schnabel et al. (2000) para a área de PLN, modelos com bom desempenho na aplicação final (recomendação) não irão necessariamente carregar conteúdo intrínseco. Dos modelos de representação comparados, o Item2Vec e o User2Vec são aqueles com a maior capacidade de aprender informações intrínsecas em relação ao conteúdo dos itens que representam. O Interact2Vec, embora alcance resultados favoráveis na tarefa de recomendação, especialmente considerando suas vantagens em relação aos demais métodos, não é recomendado para tarefas que utilizem diretamente de conteúdo implícito, como as apresentadas por Lu et al. (2015).

4.3.2.2 CB-NDCG

A classificação automática de categorias e *tags* possui a desvantagem de ser influenciada por valores parametrizáveis. Diferentes valores de K e estratégias de votação podem influenciar consideravelmente o resultado, aumentando a complexidade de aplicação da técnica avaliativa.

Para contornar este problema, é proposta uma nova métrica: a aplicação da já conhecida medida avaliativa Ganho Cumulativo Descontado Normalizado (NDCG), comparando a representação vetorial aprendida com a gerada baseada no conteúdo do item. Por esse motivo, a métrica é chamada de Ganho Cumulativo Descontado Normalizado Baseado em Conteúdo (CB-NDCG, do inglês *Content-based Normalized Discounted Cumulative Gain*).

É assumido que há uma ordem correta para a vizinhança de um dado item, gerada através da similaridade entre os atributos do item alvo e dos demais itens do catálogo, como feito em recomendadores baseado em conteúdo tradicionais (PAZZANI; BILLSUS, 2007). Com isso, é possível comparar o ordenamento gerado pela representação vetorial com a vizinhança construída pelo conteúdo intrínseco, aplicando o NDCG para mensurar a semelhança entre as ordenações por meio de um valor quantitativo.

Em sistemas de recomendação baseado em conteúdos tradicionais, um item $i \in I$ pode ser representado como um vetor $\vec{f}$ de atributos, sendo seu conteúdo expresso pela Equação 4.12:

$$\vec{f}_{i,a} = \begin{cases} 1, & \text{se o item } i \text{ possui o atributo } a \\ 0, & \text{caso contrário} \end{cases} \tag{4.12}$$

Através desta representação, é possível construir uma matriz de similaridade $\mathcal{C}$ entre itens. De maneira semelhante, é possível construir uma matriz $\mathcal{E}$, de mesmas dimensões, contendo a semelhança entre itens de acordo com a representação vetorial avaliada. Por

fim, para cada matriz, é possível construir uma vizinhança de K itens similaridades para cada item i: $N_{\mathcal{C}_i}$ para a matriz $\mathcal{C}$ e $N_{\mathcal{E}_i}$ para a matriz $\mathcal{E}$.

O CB-NDCG pode ser calculado como explicado na Seção 4.2.2.1, sendo o DCG ideal (IDCG) gerado de acordo com a vizinhança $N_{\mathcal{C}}$, como mostrado pelas Equações 4.13, 4.14, 4.15 e 4.16.

$$\text{CB-NDCG} = \frac{1}{|I|} \sum_{i \in I} (\text{CB-NDCG}_i) \tag{4.13}$$

$$\text{CB-NDCG}_i = \frac{\text{CB-DCG}_i}{\text{CB-IDCG}_i} \tag{4.14}$$

$$\text{CB-DCG}_i = \sum_{n=1}^{K} \frac{\mathcal{C}_{i,N_{\mathcal{E}_{in}}}}{\log_2(n+1)} \tag{4.15}$$

$$\text{CB-IDCG}_i = \sum_{n=1}^{K} \frac{\mathcal{C}_{i,N_{\mathcal{C}_{in}}}}{\log_2(n+1)} \tag{4.16}$$

A métrica possui a vantagem de ser calculada de forma automática, eliminando interferência humana e subjetividade. Também apresenta a qualidade de avaliar o ordenamento gerado para a vizinhança de cada item. Como desvantagem, a medida depende que a representação baseada em conteúdo seja confiável para descrever o item, o que pode não ser verdade em todos os cenários. Ainda assim, é uma estratégia bastante adotada em recomendadores baseados em conteúdo, e pode fornecer uma ideia inicial da qualidade intrínseca das representações.

Para avaliar os modelos, foi calculado o CB-NDCG de cada representação gerada, considerando $K = 15$ (CB-NDCG@15). Os resultados encontram-se na Tabela 17.

Conjunto	SVD	BPR	ITM2V	USR2V	INT2V
Anime	0,3657	0,4173	0,4579	0,4412	0,268
BestBuy	0,1307	0,1558	0,3086	0,3407	0,0405
DeliciousBookmarks	0,185	0,178	0,184	0,1868	0,1271
Last.FM	0,4362	0,3852	0,412	0,4226	0,2962
MovieLens	0,3478	0,3873	0,3949	0,356	0,1991

Tabela 17 – Resultados para o Ganho Cumulativo Descontado Normalizado Baseado em Conteúdo (CB-NDCG)

Os resultados obtidos pelo CB-NDCG se mostraram bastante semelhantes aos obtidos pela classificação automática de texto (Tabela 16), evidenciando a superioridade dos modelos Item2Vec e User2Vec. A similaridade de desempenho entre essas diferentes tarefas evidencia como o CB-NDCG pode ser utilizado para estimar o conteúdo intrínseco

das representações vetoriais sem demandar ajustes de parâmetros, como feito na tarefa da seção anterior.

Por apresentarem conteúdo semelhante, as conclusões geradas sobre a classificação automática de categorias e *tags* pode ser estendida para o CB-NDCG. O Interact2Vec se mostrou um método eficiente, alcançando resultados interessantes para a tarefa de recomendação top-N. Entretanto, para problemas que utilizam conteúdo intrínseco de forma direta, o método não obteve bons resultados, havendo outras técnicas de RVDs de mesma complexidade computacional, como o User2Vec, e que possuem maior capacidade de aprender conteúdo intrínseco.

4.4 Considerações finais

Neste capítulo, foram apresentados diferentes experimentos realizados com o objetivo de comparar o modelo proposto com outros métodos consolidados na área de recomendação. Inicialmente, foi descrito o protocolo experimental utilizado para condução dos experimentos, apresentando os conjuntos de dados utilizados e as etapas de manipulação dos dados realizadas antes do treinamento dos modelos e geração da recomendação.

Foram conduzidos dois tipos de experimentos diferentes: extrínseco e intrínseco. O primeiro busca avaliar a qualidade dos métodos e modelos de representação para a tarefa da recomendação em si. O segundo, para aferir a capacidade das representações vetoriais aprendidas de carregar informações referentes ao conteúdo do objeto representado. Nestes dois tipos de experimentos, mais de um cenário avaliativo foi executado, de forma a construir uma visão ampla sobre o desempenho de cada modelo, especialmente o proposto.

Os resultados dos experimentos extrínsecos mostraram a capacidade do Interact2Vec de gerar recomendações em um cenário de ranqueamento top-N, sendo um modelo mais computacionalmente eficiente que os demais e atingindo resultados similares. Entretanto, o modelo, assim como os demais métodos de RVD, não é recomendado para a tarefa de previsão de nota. Por não consumir interações explícitas, muitas informações sobre o problema não são assimiladas pelas representações geradas, fazendo com que os métodos de RVD apresentem um alto erro na predição para a maioria das bases de dados.

Os resultados dos experimentos intrínsecos evidenciaram mais propriedades importantes sobre o Interact2Vec. Inicialmente, através da técnica de detecção de intruso, percebeu-se como o método possui maior facilidade em aprender conteúdo intrínseco para itens populares, isto é, que possuam muitas interações dentro da base de dados, apresentando ganhos não compartilhados por nenhum outro método. Entretanto, as tarefas de detecção automática de *tags* e a nova métrica proposta, CB-NDCG, mostraram que o Interact2Vec não é um modelo apropriado para aprender significado intrínseco referente aos metadados dos itens. Tal fator possivelmente ocorre devido a sua já mencionada falta

de habilidade em aprender informações de conteúdo sobre itens pouco consumidos, o que constitui a maior parcela dos conjuntos de dados para recomendação.

Desta forma, com base nos resultados, é possível concluir que o Interact2Vec pode ser um excelente modelo em cenários onde se deseja gerar recomendações top-N, e há baixo poder computacional ou dados de entrada limitados. Para cenários de previsão de nota, ou tarefas que demandem conteúdo intrínseco, há outros métodos mais adequados.

Conclusões

Nesta dissertação, foi proposto, desenvolvido e avaliado um novo modelo neural de RVD em sistemas de recomendação de filtragem colaborativa, chamado Interact2Vec. O modelo tem como objetivo ser capaz de gerar uma representação vetorial distribuída de itens e usuários concomitantemente, de forma computacionalmente eficiente e sem demandar dados de conteúdo durante o aprendizado. Essas características possibilitam sua aplicação em cenários onde há escassez de recursos computacionais ou presença apenas de *feedback* implícito entre usuários e itens.

Inicialmente, foi conduzida uma apresentação sobre a área de sistemas de recomendação, abordando sua origem e cenário atual. Foram introduzidos conceitos básicos, como classificações de métodos de recomendação e tipos de dados de entrada, seguido por uma revisão bibliográfica de sistemas de recomendação baseados em filtragem colaborativa, dividindo-os de acordo com seu funcionamento interno e estratégia de aprendizado.

Algumas das limitações da área foram expostas, como os problemas de alta esparsidade e dimensionalidade enfrentados por certos modelos de representação. Uma das mais recentes estratégias sugeridas para aliviar estes problemas é o uso de métodos de RVD: arquiteturas de redes neurais artificiais responsáveis por gerar representações vetoriais distribuídas de itens ou usuários. Dessa forma, foi conduzida uma minuciosa revisão da literatura de modelos baseados em RVD, separando-os de acordo com suas estratégias de representação, apontando as tendências na área e descrevendo o funcionamento de dois métodos pioneiros: o Item2Vec e o User2Vec.

Com a literatura atual para métodos de RVD devidamente contextualizada, foi introduzido o Interact2Vec, método para geração concomitante de RVDs de itens e usuários, inspirado na arquitetura do Item2Vec e com objetivo semelhante ao User2Vec. Foi apresentada sua arquitetura neural, em conjunto com estratégias de aprendizado e de recomendação que podem ser aplicadas sobre o modelo. Em seguida, foi oferecida uma análise sobre sua complexidade computacional, comparando-o com outros métodos da área. Por fim, o modelo foi avaliado em três cenários diferentes do contexto de recomendação: ranqueamento top-N, previsão de notas, e uma inédita avaliação intrínseca.

Em um cenário de avaliação por ranqueamento top-N, o Interact2Vec obteve resultados competitivos com os métodos clássicos, como o KNN, o SVD e o Item2Vec, pois uma análise não-paramétrica mostrou que os resultados obtidos foram estatisticamente equivalentes. Além disso, o método proposto apresenta diversas vantagens, como a geração de uma representação de baixa dimensionalidade de maneira computacionalmente eficiente. Seus resultados foram superiores aos métodos Bayesian Personalized Ranking e User2Vec,

tendo este último sido comprovado por meio de testes estatísticos.

Para um cenário de previsão de nota, o modelo apresentou resultados insatisfatórios, embora tal resultado já fosse esperado, visto que – ao contrário de métodos estado-da-arte na tarefa – o Interact2Vec não consome *feedback* explícito durante o aprendizado. Demais métodos de RVD também apresentaram resultados pouco expressivos. Desta forma, recomenda-se utilizar o método proposto em cenários de recomendação top-N, especialmente naqueles onde há baixo poder computacional disponível e não há dados adicionais sobre os itens e usuários do sistema.

Por fim, foi realizada uma avaliação intrínseca, com o objetivo de analisar a capacidade dos modelos de representação em aprender informações sobre o conteúdo dos objetos que representam, como, por exemplo, os metadados dos itens. O modelo proposto não apresentou bons resultados nesta avaliação, mostrando-se capaz de aprender informações de conteúdo apenas para itens frequentemente consumidos. Desta forma, outros métodos de RVD, como o Item2Vec e o User2Vec, figuram como mais indicados para problemas que demandem conteúdo intrínseco.

Trabalhos futuros

Existem diversos possíveis pontos de melhoria para o modelo de RVD proposto, que não puderam ser explorados no escopo desta dissertação. Em adição ao Interact2Vec, também foi observado possibilidades de avanço na métrica CB-NDCG, utilizada para avaliação intrínseca de representações vetoriais no contexto de sistemas de recomendação. A seguir, são listadas algumas sugestões de pesquisas futuras:

- **Adaptar o Interact2Vec para a tarefa de previsão de nota.** Através dos resultados dos experimentos extrínsecos, foi verificado como o Interact2Vec, assim como outros métodos de RVD, possui um desempenho baixo para a tarefa de previsão de nota. Isso acontece devido ao funcionamento do modelo, que consome apenas *feedback* implícito durante a etapa de aprendizado, não sendo capaz de gerar uma representação vetorial distribuída que carregue preferências e recusas por parte do usuário. Sugere-se então uma alteração na arquitetura ou função-objetivo do modelo, para que considere as avaliações atribuídas pelos usuários aos itens. Outra opção sugerida é o uso de novos métodos de recomendação, que ponderem a similaridade entre RVDs de acordo com as avaliações do usuário previamente a etapa de construção da vizinhança;

- **Enriquecer o Interact2Vec por meio de metadados de itens e informações demográficas de usuário.** Embora tal estratégia descaracterize uma das principais características do Interact2Vec, sua facilidade de aplicação por depender de poucos

dados, já está consolidado na literatura que o uso de métodos híbridos, especialmente a combinação de métodos de filtragem colaborativa com baseado em conteúdo, pode gerar resultados superiores (BOBADILLA et al., 2013). Adicionalmente, pesquisas evidenciam que consumir metadados durante a geração das RVDs pode aumentar sua qualidade (VASILE; SMIRNOVA; CONNEAU, 2016; FU et al., 2017). Mesmo que passe a ser de mais difícil aplicação, é sugerido expandir o método para que resultados melhores sejam alcançados, possivelmente superando métodos mais complexos de RVD;

- **Aplicar técnicas complexas para gerar a recomendação.** Todos os métodos de recomendação por meio de RVD apresentados nessa dissertação são baseados em métodos de vizinhança. Embora tenham a vantagem de ser simples, não adicionando complexidade ao modelo, estratégias mais elaboradas de recomendação, como redes neurais convolucionais ou recorrentes que consumam as RVDs do Interact2Vec, podem gerar resultados mais competitivos com os demais métodos da literatura (TAN; XU; LIU, 2016; GREENSTEIN-MESSICA; ROKACH; FRIEDMAN, 2017; TANG; WANG, 2018).

- **Inserir o fator "tempo" durante o aprendizado.** Tanto o Interact2Vec quanto os demais métodos avaliados nesta pesquisa são estáticos, ou seja, consideram que todas as interações no histórico são igualmente importantes, e que ocorreram de forma simultânea. Pesquisas recentes dentro da área mostram como esta abordagem pode ser penalizada em comparação com técnicas que consideram o uso do tempo para gerar as recomendações (KOREN, 2008; BEEL, 2017). Assim, acredita-se que é possível melhorar os resultados do Interact2Vec em aplicações reais através de estratégias que façam com que o momento da interação seja considerado pela rede. Algumas sugestões são o uso de janelas de esquecimento, funções de decaimento ou interações ponderadas pelo tempo (SHI; LARSON; HANJALIC, 2014).

- **Gerar comparações com representações mais complexas para o cálculo do CB-NDCG.** Para calcular o CB-NDCG, métrica de avaliação intrínseca proposta nesta dissertação, foi sugerido o uso de uma representação tradicional baseada em conteúdo, de acordo com os atributos dos itens. Avanços na área de filtragem baseada em conteúdo mostram que representações mais complexas e específicas de domínio, podem gerar resultados melhores na recomendação final, tais como uso de dados visuais não estruturados para filmes (FILHO; WEHRMANN; BARROS, 2017) ou *embeddings* baseadas em grafos de contexto para músicas (WANG; XU; DENG, 2017). O uso dessas representações para construir a vizinhança ideal para o CB-NDCG poderá tornar a medida mais confiável e robusta, capaz de aferir com maior qualidade o significado intrínseco das RVDs aprendidas.

Referências

ABADI, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Disponível em: <https://www.tensorflow.org/>. Citado na página 84.

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 17, n. 6, p. 734–749, 2005. Citado 3 vezes nas páginas 27, 31 e 32.

AGGARWAL, C. C. et al. Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In: *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 1999. (KDD '99"), p. 201—212. Citado na página 43.

AHUJA, R.; SOLANKI, A.; NAYYAR, A. Movie recommender system using k-means clustering AND k-nearest neighbor. In: *Proceedings of the 9th International Conference on Confluence The Next Generation Information Technology Summit*. New York, NY, USA: IEEE, 2019. (Confluence). Citado na página 41.

AMATRIAIN, X.; PUJOL, J. M. Data mining methods for recommender systems. In: RICCI, F.; ROKACH, L.; SHAPIRA, B. (Ed.). *Recommender Systems Handbook*. Berlin/Heidelberg, Germany: Springer Science+Business Media, 2015. cap. 6, p. 227–262. ISBN 978-1-4899-7636-9. Citado na página 44.

APPLE. *Turi Create*. 2019. <https://github.com/apple/turicreate>. Citado 2 vezes nas páginas 84 e 91.

ARORA, S. *Recommendation Engines: How Amazon and Netflix Are Winning the Personalization Battle*. 2016. Disponível em: <https://www.martechadvisor.com/articles/customer-experience-2/recommendation-engines-how-amazon-and-netflix-are-winning-the-personalization-battle/>. Citado na página 33.

BAEZA-YATES, R.; RIBEIRO-NETO, B. *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley, 1999. ISBN 978-0201398298. Citado na página 34.

BALABANOVIĆ, M.; SHOHAM, Y. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, Association for Computing Machinery, New York, NY, USA, v. 40, n. 3, p. 66–72, 1997. Citado na página 36.

BARKAN, O.; KOENIGSTEIN, N. Item2Vec: Neural item embedding for collaborative filtering. In: *IEEE 26th International Workshop on Machine Learning for Signal Processing*. Piscataway, NJ, USA: IEEE, 2016. (MLSP 2016), p. 1–6. Citado 13 vezes nas páginas 28, 44, 48, 49, 53, 55, 59, 62, 84, 91, 96, 102 e 127.

BASU, C.; HIRSH, H.; COHEN, W. Recommendation as classification: Using social and content-based information in recommendation. In: *Proceedings of the 15th National/10th Conference on Artificial intelligence/Innovative Applications of Artificial Intelligence*.

Menlo Park, CA, USA: American Association for Artificial Intelligence, 1998. (AAAI '98/IAAI '98), p. 714—-720. Citado na página 36.

BEEL, J. *It's Time to Consider 'Time' when Evaluating Recommender-System Algorithms*. [S.l.], 2017. Citado na página 109.

BEEL, J. et al. Introducing mr. dlib, a machine-readable digital library. In: *Proceedings of the 11th ACM/IEEE Joint Conference on Digital Libraries*. New York, NY, USA: Association for Computing Machinery, 2011. (JCDL '11), p. 463–464. Citado na página 51.

BEHERA, B. et al. *Distributed Vector Representation Of Shopping Items, The Customer And Shopping Cart To Build A Three Fold Recommendation System*. 2017. Citado na página 52.

BELL, R. M.; KOREN, Y. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In: *Proceedings of the 7th IEEE International Conference on Data Mining*. New York, NY, USA: IEEE, 2007. (ICDM 2007), p. 1–10. Citado na página 42.

BENGIO, Y. et al. A neural probabilistic language model. *The Journal of Machine Learning Research*, JMLR.org, v. 3, p. 1137–1155, 2003. Citado 2 vezes nas páginas 28 e 47.

BENNETT, J.; LANNING, S. The Netflix Prize. In: *Proceedings of KDD Cup and Workshop 2007*. New York, NY, USA: Association for Computing Machinery, 2007. (KDDCup '07). Citado 3 vezes nas páginas 32, 37 e 89.

BILLSUS, D.; PAZZANI, M. J. Learning collaborative information filters. In: *Proceedings of the 15th International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998. (ICML '98), p. 46–54. Citado na página 45.

BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent dirichlet allocation. *The Journal of Machine Learning Research*, JMLR.org, v. 3, p. 993–1022, 2003. Citado na página 47.

BOBADILLA, J. et al. Recommender systems clustering using bayesian non negative matrix factorization. *IEEE Access*, IEEE, New York, NY, USA, v. 6, p. 3549–3564, 2017. Citado na página 43.

BOBADILLA, J. et al. Recommender systems survey. *Knowledge-Based Systems*, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 46, p. 109–132, 2013. Citado 9 vezes nas páginas 27, 28, 31, 33, 34, 35, 39, 43 e 109.

BONFERRONI, C. E. Teoria statistica delle classi e calcolo delle probabilitá. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, Firenze, Italy, 1936. Citado na página 83.

BREESE, J. S.; HECKERMAN, D.; KADIE, C. Empirical analysis of predictive algorithms for collaborative filtering. In: *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998. (UAI '98), p. 43—-52. Citado na página 40.

BURKE, R. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, Kluwer Academic Publishers, Dordrecht, Netherlands, v. 12, n. 4, p. 331–370, 2002. Citado na página 36.

BURKE, R.; FELFERNIG, A.; GÖKER, M. H. Recommender systems: An overview. *AI Magazine*, Association for the Advancement of Artificial Intelligence, Menlo Park, CA, USA, v. 32, n. 3, p. 13–18, 2011. Citado na página 36.

CAMACHO-COLLADOS, J.; PILEHVAR, M. T. From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research*, AI Access Foundation, El Segundo, CA, USA, v. 63, n. 1, p. 743–788, 2018. Citado na página 48.

CANDILLIER, L.; MEYER, F.; BOULLé, M. Comparing state-of-the-art collaborative filtering systems. In: *Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition*. Berlin/Heidelberg, Germany: Springer Science+Business Media, 2007. (MLDM 2007, Lecture Notes in Computer Science 4571), p. 548–562. Citado na página 33.

CASELLES-DUPRéS, H.; LESAINT, F.; ROYO-LETELIER, J. Word2vec applied to recommendation: hyperparameters matter. In: *Proceedings of the 12th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2018. (RecSys '18), p. 352–356. Citado 3 vezes nas páginas 62, 63 e 84.

CENIKJ, G.; GIEVSKA, S. Boosting recommender systems with advanced embedding models. In: *Companion Proceedings of the Web Conference 2020*. New York, NY, USA: Association for Computing Machinery, 2020. (WWW '20), p. 385—-389. Citado na página 43.

CHAMBERLAIN, B. P. et al. Customer lifetime value prediction using embeddings. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2017. (KDD '17), p. 1753—-1762. Citado na página 47.

CHEN, Q. et al. BioConceptVec: Creating and evaluating literature-based biomedical concept embeddings on a large scale. *PLOS Computational Biology*, Public Library of Science, San Francisco, CA, USA, v. 16, n. 4, p. 1–18, 2020. Citado na página 47.

CHENG, H.-T. et al. Wide & deep learning for recommender systems. In: *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2016. (DLRS 2016), p. 7—-10. Citado na página 50.

CHO, K. et al. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2014. (EMNLP 2014), p. 1532–1543. Citado na página 50.

CHOLLET, F. et al. *Keras*. GitHub, 2015. Disponível em: <https://github.com/fchollet/keras>. Citado 2 vezes nas páginas 84 e 92.

CHRISTAKOU, C.; STAFYLOPATIS, A. A hybrid movie recommender system based on neural networks. In: *Proceedings of the 2005 5th International Conference on Intelligent Systems Design and Applications*. Washington, D.C., USA: IEEE Computer Society, 2005. (ISDA '05), p. 1–6. Citado na página 44.

CLAYPOOL, M. et al. Combining content-based and collaborative filters in an online newspaper. In: *Proceedings of the ACM SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*. New York, NY, USA: Association for Computing Machinery, 1999. (SIGIR '99), p. 1–8. Citado na página 36.

COLLINS, A.; BEEL, J. Document embeddings vs. keyphrases vs. terms for recommender systems: A large-scale online evaluation. In: *Proceedings of the 2019 ACM/IEEE Joint Conference on Digital Libraries*. New York, NY, USA: IEEE, 2019. (JCDL 2019), p. 130–133. Citado na página 51.

COLLOBERT, R.; WESTON, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In: *Proceedings of the 25th International Conference on Machine learning*. Helsinki, Finland: [s.n.], 2008. (ICML 08), p. 160–167. Citado na página 48.

COVINGTON, P.; ADAMS, J.; SARGIN, E. Deep neural networks for youtube recommendations. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2016. (RecSys '16), p. 191–198. Citado na página 33.

CREMONESI, P.; KOREN, Y.; TURRIN, R. Performance of recommender algorithms on top-n recommendation tasks. In: *Proceedings of the 4th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2010. (RecSys '10), p. 39–46. Citado 3 vezes nas páginas 36, 37 e 42.

CRICHTON, G. et al. Neural networks for link prediction in realistic biomedical graphs: a multi-dimensional evaluation of graph embedding-based approaches. *BMC Bioinformatics*, BioMed Central, London, UK, v. 19, p. 176:1–176:11, 2018. Citado na página 43.

DEERWESTER, S. et al. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, Wiley-Blackwell, Hoboken, NJ, USA, v. 41, n. 6, p. 391–407, 1990. Citado 2 vezes nas páginas 41 e 47.

DEGEMMIS, M.; LOPS, P.; SEMERARO, G. A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. *User Modeling and User-Adapted Interaction*, Springer US, New York, NY, USA, v. 17, p. 217—255, 2007. Citado na página 40.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, JMLR.org, v. 7, p. 1–30, 2006. Citado na página 83.

DESHPANDE, M.; KARYPIS, G. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, Association for Computing Machinery, New York, NY, USA, v. 22, n. 1, p. 143–177, 2004. Citado 5 vezes nas páginas 37, 40, 51, 66 e 80.

DESROSIERS, C.; KARYPIS, G. A comprehensive survey of neighborhood-based recommendation methods. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*.

New York, NY, USA: Springer US, 2011. cap. 4, p. 107–144. ISBN 978-0-387-85819-7. Citado 4 vezes nas páginas 41, 43, 66 e 83.

DRACHSLER, H.; HUMMEL, H. G. K.; KOPER, R. Personal recommender systems for learners in lifelong learning networks: the requirements, techniques and model. Inderscience Publishers, Geneva, Switzerland, v. 3, n. 4, p. 404—-423, 2008. Citado na página 45.

DUNN, O. J. Multiple comparisons among means. *Journal of the American Statistical Association*, Taylor & Francis, Oxfordshire, UK, v. 56, n. 293, p. 52–64, 1961. Citado na página 83.

EKSTRAND, M. D.; KONSTAN, J. A. Recommender systems notation: Proposed common notation for teaching and research. *Boise State University Computer Science Faculty Publications and Presentations*, Boise, ID, USA, v. 177, p. 1–7, 2019. Citado na página 38.

FILHO, R. J. R.; WEHRMANN, J.; BARROS, R. C. Leveraging deep visual features for content-based movie recommender systems. In: *Proceedings of the 2017 International Joint Conference on Neural Networks*. New York, NY, USA: IEEE, 2017. (IJCNN 2017), p. 604–611. Citado na página 109.

FOROUZANDEH, S. et al. Recommender system for users of internet of things (iot). *International Journal of Computer Science and Network Security*, IJCSNS, South Korea, v. 17, n. 8, p. 46–51, 2017. Citado na página 33.

FOUSS, F. et al. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, New York, NY, USA, v. 19, n. 3, p. 355–369, 2007. Citado na página 43.

FREDERICKSON, B. *Implicit: Fast Python Collaborative Filtering for Implicit Datasets*. 2017. <https://github.com/benfred/implicit>. Citado na página 84.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, Taylor & Francis, Oxfordshire, UK, v. 32, n. 200, p. 675–701, 1937. Citado na página 83.

FU, P. et al. Attr2vec: a neural network based item embedding method. In: *Proceedings of the 2nd International Conference on Computer, Mechatronics and Electronic Engineering*. Lancaster, PA, USA: DEStech Publications, 2017. (CMEE 2017), p. 300–307. Citado 4 vezes nas páginas 49, 96, 102 e 109.

FUNK, S. *Netflix Update: Try This at Home*. 2016. Disponível em: <https://sifter.org/~simon/journal/20061211.html>. Citado 2 vezes nas páginas 42 e 91.

GOLDBERG, D. et al. Using collaborative filtering to weave and information tapestry. *Communications of the ACM*, Association for Computing Machinery, New York, NY, USA, v. 35, n. 12, p. 61–70, 1992. Citado 4 vezes nas páginas 27, 32, 35 e 39.

GOLDBERG, Y. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, AAAI Press, Palo Alto, CA, USA, v. 57, n. 1, p. 345–420, 2016. Citado na página 47.

GOLDBERG, Y.; HIRST, G. *Neural Network Methods in Natural Language Processing*. San Rafael, CA, USA: Morgan & Claypool Publishers, 2017. ISBN 978-1-62705-298-6. Citado na página 47.

GOMEZ-URIBE, C. A.; HUNT, N. The Netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems*, Association for Computing Machinery, New York, NY, USA, v. 6, n. 4, p. 13:1–13:19, 2015. Citado 4 vezes nas páginas 27, 33, 35 e 36.

GORI, M.; PUCCI, A. ItemRank: A random-walk based scoring algorithm for recommender engines. In: *Proceedings of the 20th International Joint Conference on Artifical Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007. (IJCAI '07), p. 2766—-2771. Citado na página 43.

GRBOVIC, M.; CHENG, H. Real-time personalization using embeddings for search ranking at Airbnb. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2018. (KDD '18), p. 311–320. Citado 5 vezes nas páginas 27, 33, 35, 51 e 52.

GRBOVIC, M. et al. E-commerce in your inbox: Product recommendations at scale. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2015. (KDD '15), p. 1809–1818. Citado 9 vezes nas páginas 33, 44, 48, 52, 53, 59, 74, 84 e 91.

GREENSTEIN-MESSICA, A.; ROKACH, L.; FRIEDMAN, M. Session-based recommendations using item embedding. In: *Proceedings of the 22nd International Conference on Intelligent User Interfaces*. New York, NY, USA: Association for Computing Machinery, 2017. (IUI '17), p. 629–633. Citado 3 vezes nas páginas 48, 50 e 109.

GUY, I. Social recommender systems. In: RICCI, F.; ROKACH, L.; SHAPIRA, B. (Ed.). *Recommender Systems Handbook*. Berlin/Heidelberg, Germany: Springer Science+Business Media, 2015. cap. 15, p. 511–543. ISBN 978-1-4899-7636-9. Citado na página 33.

HALLINAN, B.; STRIPHAS, T. Recommended for you: The Netflix Prize and the production of algorithmic culture. *New Media and Society*, SAGE Publishing, New York, NY, USA, v. 18, n. 1, p. 117–137, 2016. Citado na página 32.

HARRIS, C. R. et al. Array programming with NumPy. *Nature*, Springer Science+Business Media, v. 585, n. 7825, p. 357–362, 2020. Citado 2 vezes nas páginas 84 e 91.

HASANZADEH, S.; FAKHRAHMAD, S. M.; TAHERI, M. Review-based recommender systems: A proposed rating prediction scheme using word embedding representation of reviews. *The Computer Journal*, The British Computer Society, London, UK, p. 1–10, 2020. Citado na página 52.

HE, X. et al. Neural collaborative filtering. In: *Proceedings of the 26th International Conference on World Wide Web*. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2017. (WWW '17), p. 173—-182. Citado na página 53.

HERLOCKER, J.; KONSTAN, J. A.; RIEDL, J. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, Springer

Science+Business Media, Berlin/Heidelberg, Germany, v. 5, p. 287–310, 2002. Citado 2 vezes nas páginas 27 e 39.

HERLOCKER, J. L. et al. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, Association for Computing Machinery, New York, NY, USA, v. 22, n. 1, p. 5–53, 2004. Citado 3 vezes nas páginas 36, 37 e 91.

HERNANDO, A.; BOBADILLA, J.; ORTEGA, F. A non negative matrix factorization for collaborative filtering recommender systems based on a bayesian probabilistic model. *Knowledge-Based Systems*, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 97, n. C, p. 188—-202, 2016. Citado na página 43.

HIDASI, B. et al. Session-based recommendations with recurrent neural networks. In: *Proceedings of the International Conference on Learning Representations*. Amherst, MA, USA: OpenReview, 2016. (ICLR 2016), p. 1–10. Citado na página 50.

HILL, W. et al. Recommending and evaluating choices in a virtual community of use. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM SIGCHI, 1995. (CHI '95), p. 194–201. Citado na página 32.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, MIT Press, Cambridge, MA, USA, v. 9, n. 8, p. 1735–1780, 1997. Citado na página 51.

HU, Y.; KOREN, Y.; VOLINSKY, C. Collaborative filtering for implicit feedback datasets. In: *Proceedings of the 8th IEEE International Conference on Data Mining*. Washington, D.C., USA: IEEE Computer Society, 2008. (ICDM '08), p. 263–272. ISBN 9780769535029. Disponível em: <https://doi.org/10.1109/ICDM.2008.22>. Citado 5 vezes nas páginas 36, 37, 42, 72 e 83.

HUANG, Z.; CHEN, H.; ZENG, D. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems*, Association for Computing Machinery, New York, NY, USA, v. 22, n. 1, p. 116–142, 2004. Citado 2 vezes nas páginas 43 e 45.

HUANG, Z.; CHUNG, W.; CHEN, H. A graph model for e-commerce recommender systems. *Journal of the American Society for Information Science and Technology*, John Wiley & Sons, Inc., Hoboken, NJ, USA, v. 55, n. 3, p. 259–274, 2004. Citado na página 43.

HUG, N. *Surprise, a Python library for recommender systems*. 2017. <http://surpriselib.com>. Citado na página 91.

JACKSON, D. *The Netflix Prize: How a $1 Million Contest Changed Binge-Watching Forever*. 2017. Disponível em: <https://www.thrillist.com/entertainment/nation/the-netflix-prize>. Citado na página 33.

JANNACH, D. et al. *Recommender Systems: An Introduction*. Cambridge, UK: Cambridge University Press, 2010. ISBN 9780521493369. Citado 3 vezes nas páginas 27, 35 e 39.

JIN, R.; CHAI, J. Y.; SI, L. An automatic weighting scheme for collaborative filtering. In: *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY, USA: Association for Computing Machinery, 2004. (SIGIR '04), p. 337—-344. Citado na página 40.

JUN, H. J. et al. “SeoulHouse2Vec”: An embedding-based collaborative filtering housing recommender system for analyzing housing preference. *Sustainability*, MDPI, Basel, Switzerland, v. 12, n. 17, p. 6964:1–6964:23, 2020. Citado na página 53.

JÄRVELIN, K.; KEKÄLÄINEN, J. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, Association for Computing Machinery, New York, NY, USA, v. 20, n. 4, p. 422—-446, 2002. Citado na página 82.

KARYPIS, G. Evaluation of item-based top-n recommendation algorithms. In: *Proceedings of the 10th International Conference on Information and Knowledge Management*. [S.l.: s.n.], 2001. (CIKM ‘01), p. 247–254. Citado na página 40.

KHSURO, S.; ALI, Z.; ULLAH, I. Recommender systems: Issues, challenges, and research opportunities. In: *Proceedings of the 7th International Conference on Information Science and Applications*. Berlin/Heidelberg, Germany: Springer Science+Business Media, 2016. (ICISA 2016), p. 1179–1189. Citado 2 vezes nas páginas 44 e 45.

KIM, D.; YUM, B.-J. Collaborative filtering based on iterative principal component analysis. *Expert Systems with Applications*, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 28, n. 4, p. 823–830, 2005. Citado na página 42.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. In: *Proceedings of the 3rd International Conference for Learning Representations (Poster)*. Amherst, MA, USA: OpenReview, 2014. (ICLR 2014). Citado na página 128.

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. (KDD ‘08), p. 426–434. Citado 3 vezes nas páginas 33, 42 e 109.

KOREN, Y. The BellKor solution to the Netflix Grand Prize. 2009. Citado na página 33.

KOREN, Y. Collaborative filtering with temporal dynamics. In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2009. (KDD ‘09), p. 447–456. Citado na página 42.

KOREN, Y.; BELL, R. Advances in collaborative filtering. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. New York, NY, USA: Springer US, 2011. cap. 5, p. 145–186. ISBN 978-0-387-85819-7. Citado na página 41.

KOREN, Y.; BELL, R.; VOLINSKY, C. Matrix factorization techniques for recommender systems. *Computer*, v. 42, n. 8, p. 30–37, 2009. Citado 4 vezes nas páginas 28, 37, 40 e 42.

KRULWICH, B. Lifestyle Finder: Intelligent user profiling using large-scale demographic data. *AI Magazine*, Association for the Advancement of Artificial Intelligence, Menlo Park, CA, EUA, v. 18, n. 2, p. 37–46, 1997. Citado na página 34.

LAM, X. N. et al. Addressing cold-start problem in recommendation systems. In: *Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication*. New York, NY, USA: Association for Computing Machinery, 2008. (ICUIMC ‘08), p. 208—-211. Citado na página 45.

LARA-CABRERA, R.; GONZÁLEZ-PRIETO Ángel; ORTEGA, F. Deep matrix factorization approach for collaborative filtering recommender systems. *Applied Sciences*, MDPI, Basel, Switzerland, v. 10, n. 14, p. 1–14, 2020. Citado na página 43.

LE, Q.; MIKOLOV, T. Distributed representations of sentences and documents. In: *Proceedings of the 31st International Conference on Machine Learning*. [S.l.]: JMLR.org, 2014. (ICML 2014), p. 1188–1196. Citado 2 vezes nas páginas 51 e 52.

LEE, S. et al. PathRank: A novel node ranking measure on a heterogeneous graph for recommender systems. In: . New York, NY, USA: Association for Computing Machinery, 2012. (CIKM '12), p. 1637—-1641. Citado na página 43.

LEMIRE, D. Scale and translation invariant collaborative filtering systems. *Information Retrieval*, Springer Science+Business Media, Berlin/Heidelberg, Germany, v. 8, p. 129—150, 2005. Citado na página 40.

LI, G.; ZHANG, J. Music personalized recommendation system based on improved KNN algorithm. In: *Proceedings of the 3rd Advanced Information Technology, Electronic and Automation Control Conference*. New York, NY, USA: IEEE, 2018. (IAEAC), p. 777–781. Citado na página 41.

LI, H. et al. MSGD: A novel matrix factorization approach for large-scale collaborative filtering recommender systems on GPUs. *IEEE Transactions on Parallel and Distributed Systems*, IEEE, New York, NY, USA, v. 29, n. 7, p. 1530–1544, 2018. Citado na página 43.

LILIEN, G.; KOTLER, P.; MOORTHY, K. S. *Marketing Models*. London, UK: Prentice Hall, 1992. Citado na página 31.

LIN, C. H.; KAMAR, E.; HORVITZ, E. Signals in the silence: models of implicit feedback in a recommendation system for crowdsourcing. In: *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. Cambridge, MA, USA: AAAI Press, 2014. (AAAI '14), p. 908—-914. Citado na página 36.

LINDEN, G.; SMTITH, B.; YORK, J. Amazon.com Recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, IEEE, New York, NY, USA, v. 7, n. 1, p. 76–80, 2003. Citado 6 vezes nas páginas 27, 33, 35, 40, 45 e 51.

LIU, L.; MEHANDJIEV, N.; XU, D.-L. Multi-criteria service recommendation based on user criteria preferences. In: *Proceedings of the 5th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2011. (RecSys '11), p. 77–84. Citado na página 33.

LOCHTER, J. V. et al. Evaluating the impact of corpora used to train distributed text representation models for noisy and short texts. In: *Proceedings of the 2018 International Joint Conference on Neural Networks*. [S.l.: s.n.], 2018. (IJCNN 08), p. 1–8. Citado 4 vezes nas páginas 13, 47, 54 e 55.

LOPS, P.; GEMMIS, M. de; SEMERARO, G. Content-based recommender systems: State of the art and trends. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. New York, NY, USA: Springer US, 2011. cap. 3, p. 73–105. ISBN 978-0-387-85819-7. Citado na página 34.

LU, J. et al. Recommender system application developments: A survey. *Decision Support Systems*, v. 74, p. 12–32, 2015. Citado 4 vezes nas páginas 33, 35, 95 e 103.

LUDEWIG, M.; JANNACH, D. Evaluation of session-based recommendation algorithms. *User Modeling and User-Adapted Interaction*, Kluwer Academic Publishers, Dordrecht, Netherlands, v. 28, n. 4, p. 331—-390, 2018. Citado na página 50.

LUO, X. et al. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. *IEEE Transactions on Industrial Informatics*, IEEE, New York, NY, USA, v. 10, n. 2, p. 1273–1284, 2014. Citado na página 43.

MA, H. et al. Recommender systems with social regularization. In: *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2011. (WSDM '11), p. 287—296. Citado na página 43.

MATUSZYK, P.; SPILIOPOULOU, M. Selective forgetting for incremental matrix factorization in recommender systems. *Discovery Science*, v. 8777, p. 204–215, 2014. Citado na página 43.

MATUSZYK, P. et al. Forgetting methods for incremental matrix factorization in recommender systems. In: *Proceedings of the 30th Annual ACM Symposium on Applied Computing*. New York, NY, USA: Association for Computing Machinery, 2015. (SAC '15), p. 947–953. Citado na página 43.

MCALONE, N. *Why Netflix thinks its personalised recommendation engine is worth $1 billion per year*. 2016. Disponível em: <https://www.businessinsider.com.au/netflix-recommendation-engine-worth-1-billion-per-year-2016-6>. Citado na página 33.

MELVILLE, P.; MOONEY, R. J.; NAGARAJAN, R. Content-boosted collaborative filtering for improved recommendations. In: *18th National Conference on Artificial Intelligence*. USA: American Association for Artificial Intelligence, 2002. p. 187–192. Citado na página 40.

METEREN, R. V.; SOMEREN, M. van. Using content-based filtering for recommendation. In: *Proceedings of the 11th European Conference on Machine Learning. Workshop: Maching Learning in Information Age*. Berlin/Heidelberg, Germany: Springer Science+Business Media, 2000. (ECML 2000), p. 1–10. Citado na página 34.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. In: *Proceedings of the International Conference on Learning Representations (Workshop Poster)*. Amherst, MA, USA: OpenReview, 2013. (ICLR 2013). Citado 5 vezes nas páginas 48, 49, 50, 51 e 95.

MIKOLOV, T.; LE, Q. V.; SUTSKEVER, I. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168, 2013. Citado na página 95.

MIKOLOV, T. et al. Distributed representations of words and phrases and their compositionality. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2013. (NIPS 2013), p. 3111–3119. Citado 3 vezes nas páginas 48, 62 e 63.

MIKOLOV, T.; YIH, W. tau; ZWEIG, G. Linguistic regularities in continuous space word representations. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2013. (NAACL 2013), p. 746–751. Citado 2 vezes nas páginas 28 e 47.

MIRZA, B. J.; KELLER, B. J.; RAMAKRISHNAN, N. Studying recommendation algorithms by graph analysis. *Journal of Intelligent Information Systems*, Springer Science+Business Media, Berlin/Heidelberg, Germany, v. 20, p. 131—160, 2003. Citado na página 43.

MURTHI, B. P. S.; SARKAR, S. The role of the management sciences in research on personalization. *Management Science*, INFORMS, Catonsville, MD, USA, v. 49, n. 10, p. 1344–1362, 2003. Citado na página 31.

MUSTAFA, G.; FROMMHOLZ, I. Performance comparison of top n recommendation algorithms. In: *Proceedings of the 4th International Conference on Future Generation Communication Technology*. New York, NY, USA: IEEE, 2015. (FGCT 2015), p. 100–105. Citado 2 vezes nas páginas 37 e 80.

PARK, D. H. et al. A literature review and classification of recommender system research. *Expert Systems with Applications*, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 39, n. 11, p. 10059–10072, 2012. Citado 2 vezes nas páginas 32 e 40.

PATEREK, A. Improving regularized singular value decomposition for collaborative filtering. In: *Proceedings of KDD Cup and Workshop 2007*. New York, NY, USA: Association for Computing Machinery, 2007. (KDDCup '07), p. 39–42. Citado na página 42.

PAZZANI, M. J. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, Springer Science+Business Media, Berlin/Heidelberg, Germany, v. 13, p. 393–408, 1999. Citado 3 vezes nas páginas 32, 34 e 35.

PAZZANI, M. J.; BILLSUS, D. Content-based recommendation systems. *The Adaptive Web*, Springer Science+Business Media, Berlin/Heidelberg, Germany, Lecture Notes in Computer Science, vol 4321, p. 325–341, 2007. Citado 2 vezes nas páginas 34 e 103.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 84.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. GloVe: Global vectors for word representation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2014. (EMNLP 2014), p. 1532–1543. Citado na página 50.

PENNOCK, D. M. et al. Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. In: *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000. (UAI '00), p. 473—-480. Citado na página 40.

ŘEHŮŘEK, R.; SOJKA, P. Software framework for topic modelling with large corpora. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*.

Paris, France: European Language Resources Association (ELRA), 2010. (LREC 2010), p. 45–50. Citado 3 vezes nas páginas 84, 92 e 127.

RENDLE, S. Factorization machines. In: *Proceedings of the 10th IEEE International Conference on Data Mining*. New York, NY, USA: IEEE, 2010. (ICDM 2010), p. 14–17. Citado 3 vezes nas páginas 37, 42 e 91.

RENDLE, S. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology*, Association for Computing Machinery, New York, NY, USA, v. 3, n. 3, p. 57:1–57:22, 2012. Citado na página 42.

RENDLE, S. et al. BPR: Bayesian personalized ranking from implicit feedback. In: *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*. Arlington, VA, USA: AUAI Press, 2009. (UAI '09), p. 452–461. Citado 3 vezes nas páginas 42, 73 e 83.

RESNICK, P. et al. GroupLens: An open architecture for collaborative filtering of netnews. In: *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*. New York, NY, USA: Association for Computing Machinery, 1994. (CSCW '94), p. 175–186. Citado 2 vezes nas páginas 32 e 39.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. New York, NY, USA: Springer US, 2011. cap. 1, p. 1–38. ISBN 978-0-387-85819-7. Citado 2 vezes nas páginas 36 e 44.

RICH, E. User modeling via stereotypes. *Cognitive Science*, John Wiley & Sons, Hoboken, NJ, USA, v. 3, n. 4, p. 329–354, 1979. Citado na página 32.

RIJSBERGEN, C. J. V. *Information Retrieval*. Newton, MA, USA: Butterworth-Heinemann, 1979. ISBN 0408709294. Citado na página 81.

RIZOS, G.; HEMKER, K.; SCHULLER, B. Augment to prevent: Short-text data augmentation in deep learning for hate-speech classification. In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. New York, NY, USA: Association for Computing Machinery, 2019. (CIKM '19), p. 991—-1000. Citado na página 95.

ROH, T. H.; OH, K. J.; HAN, I. The collaborative filtering recommendation based on SOM cluster-indexing CBR. *Expert Systems with Applications*, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 25, n. 3, p. 413–423, 2003. Citado na página 44.

RUDOLPH, M. et al. Exponential family embeddings. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2016. (NIPS 2016), p. 478–486. Citado na página 49.

SALAKHUTDINOV, R.; MNIH, A. Probabilistic matrix factorization. In: *Proceedings of the 20th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2007. (NIPS'07), p. 1257–1264. Citado 2 vezes nas páginas 42 e 91.

SARWAR, B. M. et al. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In: *Proceedings of The 5th International Conference on Computer and Information Technology*. Dhaka, Bangladesh: [s.n.], 2002. (ICCIT 2002), p. 1–6. Citado 2 vezes nas páginas 28 e 45.

SARWAR, B. M. et al. Analysis of recommendation algorithms for e-commerce. In: *Proceedings of the 2nd ACM conference on Electronic commerce*. New York, NY, USA: Association for Computing Machinery, 2000. (EC '00), p. 158–167. Citado 2 vezes nas páginas 40 e 81.

SARWAR, B. M. et al. Application of dimensionality reduction in recommender system - a case study. In: *Proceedings of the 9th WebKDD Workshop on Web Mining for e-commerce*. New York, NY, USA: Association for Computing Machinery, 2000. (WebKDD '00), p. 1–12. Citado na página 28.

SARWAR, B. M. et al. Item-based collaborative filtering recommendation algorithms. In: *Proceedings of the 10th International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2001. (WWW '01), p. 285–295. Citado 7 vezes nas páginas 27, 37, 40, 42, 51, 74 e 89.

SARWAR, B. M. et al. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In: *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work*. New York, NY, USA: Association for Computing Machinery, 1998. (CSCW '98), p. 345–354. Citado na página 40.

SCHAFER, J. B. et al. Collaborative filtering recommender systems. In: _____. *The Adaptive Web: Methods and Strategies of Web Personalization*. Berlin/Heidelberg, germany: Springer Science+Business Media, 2007. p. 291–324. ISBN 9783540720782. Citado 2 vezes nas páginas 27 e 45.

SCHEIN, A. I. et al. 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. (SIGIR '02), p. 253—-260. Citado 2 vezes nas páginas 36 e 44.

SCHNABEL, T. et al. Evaluation methods for unsupervised word embeddings. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2000. (EMNLP 2015), p. 298–307. Citado 3 vezes nas páginas 95, 98 e 103.

SCHÜTZE, H. Word space. In: *Proceedings of the 5th International Conference on Neural Information Processing Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. (NIPS 1992), p. 895–902. Citado na página 47.

SEDHAIN, S. et al. AutoRec: Autoencoders meet collaborative filtering. In: *Proceedings of the 24th International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2015. (WWW '15 Companion), p. 111—-112. Citado na página 44.

SHAN, Y. et al. Deep crossing: Web-scale modeling without manually crafted combinatorial features. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2016. (KDD '16), p. 255—-262. Citado na página 44.

SHANI, G.; GUNAWARDANA, A. Evaluating recommendation systems. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. New York, NY, USA: Springer US, 2011. cap. 8, p. 257–259. ISBN 978-0-387-85819-7. Citado 3 vezes nas páginas 37, 80 e 90.

SHARDANAND, U.; MAES, P. Social information filtering: Algorithms for automating "word of mouth". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM SIGCHI, 1995. (CHI '95), p. 210–217. Citado 2 vezes nas páginas 32 e 39.

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 Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 47, n. 1, p. 3:1–3:45, 2014. Citado na página 109.

SIDANA, S. et al. User preference and embedding learning with implicit feedback for recommender systems. *Data Mining and Knowledge Discovery*, Springer Science+Business Media, v. 35, p. 568–592, 2021. Citado 2 vezes nas páginas 28 e 53.

SISWANTO, A. V.; TJONG, L.; SAPUTRA, Y. Simple vector representations of e-commerce products. In: *2018 International Conference on Asian Language Processing*. New York, NY, USA: IEEE, 2018. (IALP 2018), p. 368–372. Citado na página 51.

SONG, Y.; ZHANG, L.; GILES, C. L. Automatic tag recommendation algorithms for social recommender systems. *ACM Transactions on the Web*, Association for Computing Machinery, New York, NY, USA, v. 4, n. 1, p. 4:1–4:31, 2011. Citado na página 102.

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. (RecSys '13), p. 213–220. Citado na página 37.

STRASSEN, V. Gaussian elimination is not optimal. *Numerische Mathematik*, Springer Science+Business Media, Berlin/Heidelberg, Germany, v. 13, p. 354—356, 1969. Citado na página 70.

SU, X.; KHOSHGOFTAAR, T. M. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, Hindawi Limited, London, UK, v. 2009, p. 4:1–4:19, 2009. Citado 3 vezes nas páginas 39, 41 e 45.

SU, X.; KHOSHGOFTAAR, T. M.; GREINER, R. A mixture imputation-boosted collaborative filter. In: *Proceedings of the 21st International Florida Artificial Intelligence Research Society Conference*. Palo Alto, CA, USA: AAAI Press, 2008. p. 312–317. Citado na página 40.

SUBRAMANIYASWAMY, V.; LOGESH, R. Adaptive knn based recommender system through mining of user preferences. *Wireless Personal Communications*, Springer Science+Business Media, Berlin/Heidelberg, Germany, v. 97, p. 2229–2247, 2017. Citado na página 41.

TAN, Y. K.; XU, X.; LIU, Y. Improved recurrent neural networks for session-based recommendations. In: *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2016. (DLRS 2016), p. 17—-22. Citado 2 vezes nas páginas 50 e 109.

TANG, J.; WANG, K. Personalized top-n sequential recommendation via convolutional sequence embedding. In: *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2018. (WSDM '18), p. 565–573. Citado 2 vezes nas páginas 50 e 109.

THORAT, P. B.; GOUDAR, R. M.; BARVE, S. Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications*, Foundations of Computer Science, New York, NY, USA, v. 110, n. 4, p. 31–36, 2015. Citado na página 34.

TÖSCHER, A.; JAHRER, M. The BigChaos solution to the Netflix Grand Prize. 2009. Citado na página 33.

UNGAR, L. H.; FOSTER, D. P. Clustering methods for collaborative filtering. In: *Workshop on Recommender Systems at the 15th National Conference on Artificial Intelligence*. [S.l.: s.n.], 1998. (AAAI-98), p. 114–129. Citado na página 40.

VALCARCE, D. et al. Collaborative filtering embeddings for memory-based recommender systems. *Engineering Applications of Artificial Intelligence*, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 85, p. 347–356, 2019. Citado 2 vezes nas páginas 48 e 51.

VASILE, F.; SMIRNOVA, E.; CONNEAU, A. Meta-prod2vec: Product embeddings using side-information for recommendation. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2016. (RecSys '16), p. 225–232. Citado 4 vezes nas páginas 28, 49, 62 e 109.

VERMA, J. et al. Heterogeneous edge embeddings for friend recommendation. In: *Proceedings of the 41st European Conference on Information Retrieval*. Berlin/Heidelberg, Germany: Springer Science+Business Media, 2019. (ECIR 2019), p. 172–179. Citado na página 43.

VIRTANEN, P. et al. SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, v. 17, p. 261–272, 2020. Citado 2 vezes nas páginas 84 e 91.

WANG, B.; LIAO, Q.; ZHANG, C. Weight based knn recommender system. In: *Proceedings of the 5th International Conference on Intelligent Human-Machine Systems and Cybernetics*. New York, NY, USA: IEEE, 2013. (IHMSC 2013), p. 449–452. Citado na página 40.

WANG, D.; XU, G.; DENG, S. Music recommendation via heterogeneous information graph embedding. In: *Proceedings of the 2017 International Joint Conference on Neural Networks*. New York, NY, USA: IEEE, 2017. (IJCNN 2017), p. 596–603. Citado na página 109.

WANG, H.; WANG, N.; YEUNG, D.-Y. Collaborative deep learning for recommender systems. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2015. (KDD '15), p. 1235—-1244. Citado na página 50.

XUE, H.-J. et al. Deep matrix factorization models for recommender systems. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. Palo Alto, CA, USA: AAAI Press, 2017. (IJCAI '17), p. 3203—-3209. Citado na página 53.

YING, R. et al. Graph convolutional neural networks for web-scale recommender systems. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. New York, NY, USA: Association for Computing Machinery, 2018. (KDD '18), p. 974—983. Citado na página 43.

YING, X. An overview of overfitting and its solutions. *Journal of Physics Conference Series*, IOP Publishing, Bristol, UK, v. 1168, n. 2, p. 1–6, 2019. Citado na página 130.

YOON, Y. C.; LEE, J. W. Movie recommendation using metadata based word2vec algorithm. In: *2018 International Conference on Platform Technology and Service*. New York, NY, USA: IEEE, 2018. (PlatCon 2018), p. 1–6. Citado na página 52.

YU, H.-F. et al. Parallel matrix factorization for recommender systems. *Knowledge and Information Systems*, Springer Science+Business Media, Berlin/Heidelberg, Germany, v. 41, p. 793—-819, 2014. Citado na página 43.

ZARZOUR, H.; AL-SHARIF, Z. A.; JARARWEH, Y. RecDNNing: a recommender system using deep neural network with user and item embeddings. In: *Proceedings of the 10th International Conference on Information and Communication Systems*. New York, NY, USA: IEEE, 2019. (ICICS 2019). Citado 3 vezes nas páginas 28, 48 e 53.

ZENIL, H.; KIANI, N. A.; TEGNÉR, J. Quantifying loss of information in network-based dimensionality reduction techniques. *Journal of Complex Networks*, Oxford University Press, Oxford, UK, v. 4, n. 3, p. 342—362, 2016. Citado na página 129.

ZHANG, F. et al. Collaborative knowledge base embedding for recommender systems. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2016. (KDD '16), p. 353–362. Citado 2 vezes nas páginas 28 e 50.

ZHANG, S. et al. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, v. 6, 2019. Citado na página 43.

ZHANG, S. et al. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 52, n. 1, p. 5:1–5:35, 2019. Citado 2 vezes nas páginas 33 e 44.

ZHANG, W.; DU, T.; WANG, J. Deep learning over multi-field categorical data - a case study on user response prediction. In: *Proceedings of the 38th European Conference on Information Retrieval*. Berlin/Heidelberg, Germany: Springer Science+Business Media, 2016. p. 1–12. Citado na página 44.

ZHANG, Y.; LU, J.; SHAI, O. Improve network embeddings with regularization. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. New York, NY, USA: Association for Computing Machinery, 2018. (CIKM '18), p. 1643—-1646. Citado na página 64.

ZHAO, X. et al. Learning item-interaction embeddings for user recommendations. In: *Proceedings of the DAPA 2019 WSDM Workshop on Deep Matching in Practical Applications*. New York, NY, USA: Association for Computing Machinery, 2019. (DAPA '19), p. 4:1–4:6. Citado 4 vezes nas páginas 28, 37, 48 e 51.

ŹOŁNA, K.; ROMAńSKI, B. user2vec: user modeling using lstm networks. In: *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. Palo Alto, CA, USA: AAAI, 2017. (AAAI-17), p. 5025–5026. Citado na página 51.

APÊNDICE A – Experimentos para detectar a influência dos parâmetros do Interact2Vec

Assim como outros modelos baseados em RVD, o Interact2Vec possui uma quantidade considerável de hiper-parâmetros que podem ser ajustados anteriormente ao aprendizado. Isso faz com que técnicas de otimização de parâmetros que executem testes de forma exaustiva, como a busca em grade, possam ser inaplicáveis, gerando uma elevada quantidade de combinações de valores.

Para aliviar este problema, foi conduzido um estudo de impacto para os parâmetros do Interact2Vec. Para tal, foram construídos e avaliados uma série de modelos com diferentes combinações de parâmetros. O treinamento e avaliação foram realizados sobre as bases de dados CiaoDVD e Last.FM, empregadas para recomendação de filmes e artistas musicais, respectivamente. A seleção se deu por serem bases e domínios tradicionais na literatura, e por não possuírem um elevado volume de dados, possibilitando assim a execução de diversos modelos em um breve espaço de tempo.

Para avaliação do impacto no ajuste de parâmetros, foi conduzido o seguinte experimento: para cada parâmetro, foi definido um valor padrão, normalmente utilizado por métodos (BARKAN; KOENIGSTEIN, 2016) ou bibliotecas de geração de RVDs (ŘEHŮŘEK; SOJKA, 2010) (Tabela 18). Em seguida, os conjuntos de dados foram separados em treino, validação e teste, considerando uma distribuição 8:1:1, e diversas variações do modelo proposto foram ajustadas sobre o treino e avaliadas na base de validação. Para cada parâmetro analisado, seu conteúdo foi variado dentro de uma extensa lista de valores, com todos os demais parâmetros fixados nos valores padrões. Para avaliar cada modelo, foi utilizado o Ganho Descontado Ganho Cumulativo Descontado Normalizado, para uma recomendação top-15 (NDCG@15), como explicado na Seção 4.2.2.1.

Parâmetro	**Símbolo**	**Valor padrão**
Taxa de aprendizado	α	0,25
Tamanho das RVDs	M	100
Quantidade de iterações da rede	C	5
Taxa de subamostragem de itens frequentes	ρ	10^{-3}
Quantidade de amostras negativas	$\|G\|$	5
Expoente da distribuição negativa	γ	0,75
Parâmetro de regularização	λ	0

Tabela 18 – Valores comumente utilizados na literatura para parâmetros de modelos das RVDs.

Nas seções seguintes, são apresentadas os resultados observados para cada parâmetro em relação a seu impacto para a recomendação final.

A.1 Algoritmo de otimização

Para ajustar os pesos do Interact2Vec, foi selecionado o algoritmo de otimização Adam (KINGMA; BA, 2014), por ser um algoritmo bastante popular na literatura de redes neurais artificiais. O Adam possui uma taxa de aprendizado α, responsável por controlar a intensidade de atualização do vetor gradiente, que pode ser ajustada. Variou-se então α em $\{0,0025,\ 0,0075,\ 0,025,\ 0,075,\ 0,25,\ 0,75\}$. Os resultados podem ser visualizados na Figura 12.

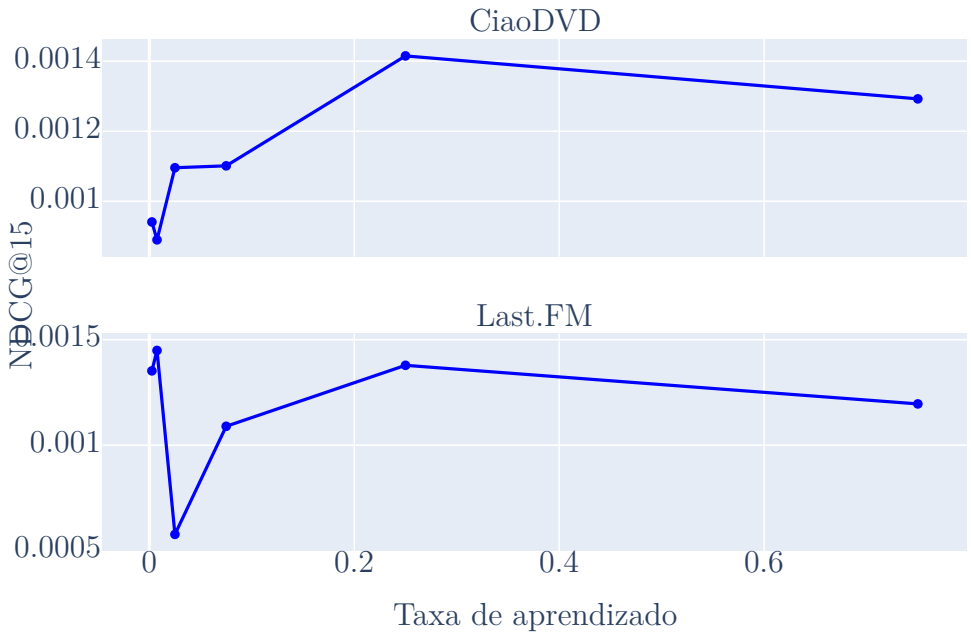

Figura 12 – NDCG@15 para diferentes valores de taxa de aprendizado (α).

Como é possível ver, em ambas as bases de dados o modelo normalmente apresenta NDCG inferior para valores baixos de α, apresentando melhoras nos dois cenários quando $\alpha \geq 0,25$. De forma a visualizar se os ganhos na métrica são significativos, uma sumarização dos resultados alcançados encontra-se na Tabela 19.

Como evidenciado nos dados, variar a taxa de aprendizado pode impactar no desempenho final do modelo, melhorando o NDCG em uma média aproximada de 25%. Para ambas as bases de dados, o comportamento dos resultados foram semelhantes, com exceção de uma expressiva melhora na base Last.FM para valores extremamente baixos de

Conjunto	Min	Q1	Mediana	Q3	Max	Q3:Q1	Max:Med
CiaoDVD	0,0009	0,0010	0,0011	0,0012	0,0014	27,05%	28,82%
Last.FM	0,0006	0,0011	0,0013	0,0014	0,0014	22,94%	13,74%

Tabela 19 – Sumário dos valores de NDCG@15 obtidos com o ajuste da taxa de aprendizado (α).

Min e **Max** contém, respectivamente, o maior e menor NDCG obtido, **Q1**, **Mediana** e **Q3** representam o primeiro, segundo e terceiro quartil dos resultados, **Q3:Q1** a porcentagem de ganho de Q3 para Q1, e **Max:Med** a porcentagem de ganho de Max para Mediana.

α. Ainda assim, verificando as variações na Tabela 19 e o resultado na Figura 12, pode-se concluir que adotar o valor padrão de $0,25$ para a taxa de aprendizado tenderá a atingir resultados satisfatórios.

A.2 Tamanho das representações vetoriais

O principal objetivo de métodos de das RVD é reduzir o tamanho em memória da representação vetorial comumente usada por sistemas de recomendação. Assim, é possível definir a dimensão final das RVDs geradas através do parâmetro M, que controla as dimensões das matrizes de peso $\mathcal{W}$ e $\mathcal{W}'$ e, consequentemente, o número de neurônios na camada intermediária $\mathcal{H}$.

Em muitos cenários, a escolha de M pode estar suscetível ao poder computacional disponível e o tamanho do conjunto de dados, sendo este o único parâmetro que exerce efeito sobre o volume em memória ocupado pela representação final. Ainda assim, é interessante analisar se há valores de M que apresentam melhores resultados, visto que, teoricamente, quanto menor o tamanho da representação vetorial, maior a parcela de informação perdida (ZENIL; KIANI; TEGNÉR, 2016).

Para analisar a importância do parâmetro, M foi variado no intervalo $[25, 300]$, com passo de 25. Os resultados para cada valor podem ser visualizados na Figura 13, e um resumo deles na Tabela 20.

Conjunto	Min	Q1	Mediana	Q3	Max	Q3:Q1	Max:Med
CiaoDVD	0,0003	0,0008	0,0010	0,0013	0,0021	68,57%	108,99%
Last.FM	0,0005	0,0009	0,0011	0,0015	0,0018	57,01%	58,5%

Tabela 20 – Sumário dos valores de NDCG@15 obtidos com o ajuste do tamanho das RVDs (M).

Colunas de mesmo significado que na Tabela 19.

A grande variação no NDCG presente na Figura 13, especialmente para a base Last.FM, é bastante interessante para a análise do impacto do parâmetro. Tal resultado

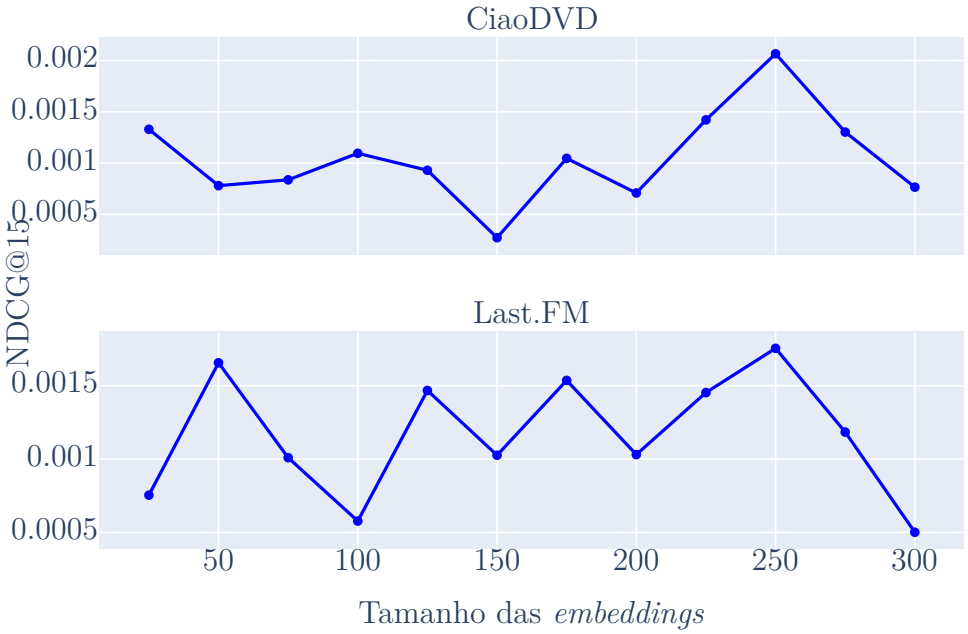

Figura 13 – NDCG@15 para diferentes valores de tamanho das RVDs (M)

evidencia como valores diferentes para M podem alterar a qualidade da representação, mas sem existir um padrão bem definido sobre valores superiores.

A importância da escolha de M fica ainda mais nítida ao observar os resultados da Tabela 20. Selecionar um valor adequado para o tamanho das RVDs pode melhorar o resultado em 60% ou até mais, como visto para a base CiaoDVD.

A.3 Quantidade de iterações da rede

O ajuste do Interact2Vec envolve C iterações sobre sua rede. Uma iteração consiste em processar todos os pares usuário-item presentes no conjunto de dados e atualizar o conteúdo das RVDs de forma a otimizar a função objetivo. Entretanto, um número alto de iterações pode resultar em problemas de superajustamento (YING, 2019), sendo assim necessário encontrar um valor de C que contribua para um bom treinamento do modelo.

Para avaliar a influência do número de iterações nos resultados do Interact2Vec, o parâmetro foi variado no intervalo $[5, 200]$, com intervalo de 25. O resultado pode ser visualizado na Figura 14.

Como mostrado no gráfico, houve um comportamento padrão para as duas bases: experimentos com poucas interações apresentaram um NDCG@15 inferior a experimentos

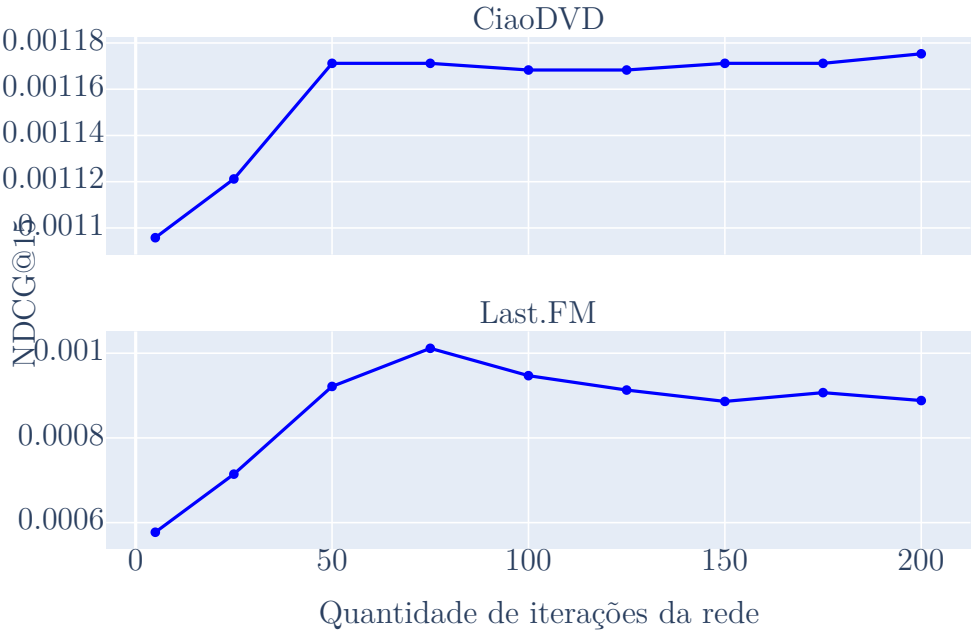

Figura 14 – NDCG@15 para diferentes valores de quantidade de iterações da rede (C).

onde $C >= 50$. Por outro lado, a métrica não sofreu grandes mudanças para valores maiores que 50, apresentando apenas uma leve queda para a base Last.FM, mas resultado constante para a base CiaoDVD.

Aliado a isto, tem-se os resultados sumarizados presentes na Tabela 21, que mostram como a variação no valor de C pouco influencia no resultado do modelo, apresentando um aumento médio máximo de $11,54\%$, na base Last.FM, número bastante inferior aos alcançados pelos outros parâmetros avaliados (taxa de aprendizado e tamanho das RVDs).

Conjunto	Min	Q1	Mediana	Q3	Max	Q3:Q1	Max:Med
CiaoDVD	0,0011	0,0012	0,0012	0,0012	0,0012	0,25%	0,35%
Last.FM	0,0006	0,0009	0,0009	0,0009	0,0010	3,99%	11,54%

Tabela 21 – Sumário dos valores de NDCG@15 obtidos com o ajuste da quantidade de iterações da rede (C).

Colunas de mesmo significado que na Tabela 19.

Com os resultados atingidos, conclui-se que não há necessidade de ajustar o valor de E. Todavia, o valor normalmente utilizado por bibliotecas ($E = 5$) tende a apresentar um resultado inferior a modelos com maior número de interações. Assim, no caso de não ajustamento do parâmetro, faz-se interessante utilizar como padrão um número de iterações entre 50 e 100, sem exceder a isso. Embora um aumento em E não pareça

impactar na qualidade das representações, o número de iterações é um dos parâmetros que mais influencia o tempo necessário para treinamento do modelo, o que pode comprometer sua aplicação prática.

A.3.1 Subamostragem de itens frequentes

Para aumentar o poder de generalização da representação e acelerar o treinamento, uma estratégia comumente adotada por métodos de das RVD é subamostragem de itens frequentes, como explicado na Seção 3.2.1. Nela, a taxa de itens descartados pode ser controlado pelo parâmetro ρ.

Com o objetivo de visualizar o impacto de ρ, o parâmetro foi variado em $[10^{-6}, 10^{-1}]$ com passo $\times 10$. Os NDCG@15 obtido para cada valor testado encontram-se na Figura 15, e evidenciam uma característica interessante: em ambas as base, a adoção de valores baixos para ρ melhora os resultados de forma expressiva, sobressaindo o NDCG alcançado pelo uso do valor padrão do parâmetro (10^{-3}). Tal resultado pode ser explicado devido a características específicas do domínio da recomendação, como a não existência de itens muito mais frequentes do que outros.

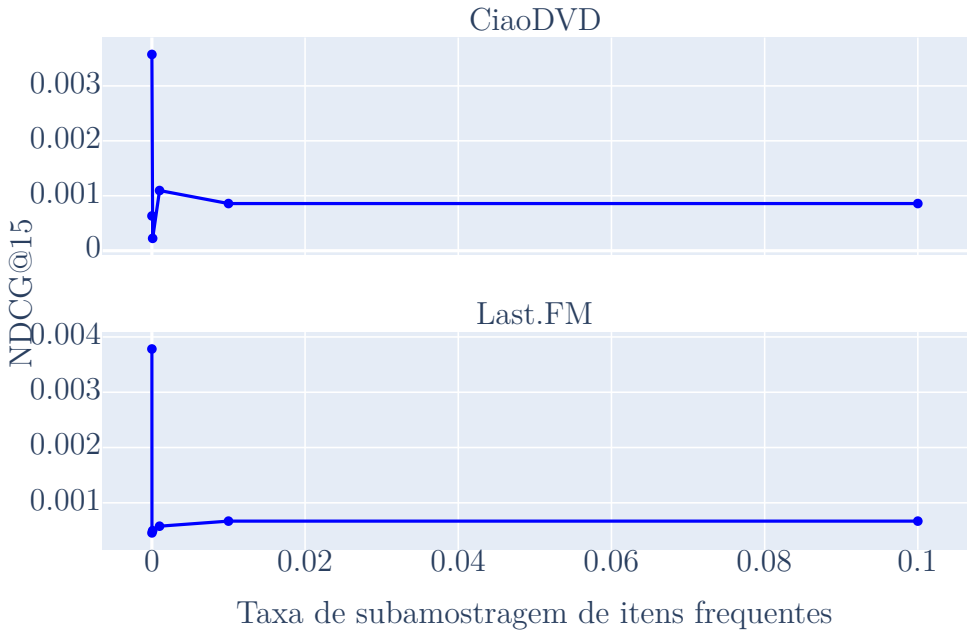

Figura 15 – NDCG@15 para diferentes valores de taxa de subamostragem de itens frequentes (ρ)

Os dados sumarizados, presentes na Tabela 22, deixam claro como a seleção de valores para ρ é extremamente importante. Para os dois conjuntos de dados, o ganho em

NDCG comparando o valor mediano com o valor máximo foi superior a 300%, atingindo até mesmo 500% para a base Last.FM. Uma escolha ruim de ρ pode comprometer significativamente a qualidade das RVDs do Interact2Vec. Adicionalmente, é melhor o uso de valores menores para o parâmetro, mantendo assim mais interações na base que alimentará a rede neural.

Conjunto	Min	Q1	Mediana	Q3	Max	Q3:Q1	Max:Med
CiaoDVD	0,0002	0,0007	0,0009	0,0010	0,0036	50,32%	316,21%
Last.FM	0,0005	0,0005	0,0006	0,0007	0,0038	29,43%	506,88%

Tabela 22 – Sumário dos valores de NDCG@15 obtidos com o ajuste da taxa de subamostragem de itens frequentes (ρ).

Colunas de mesmo significado que na Tabela 19.

A.4 Amostragem negativa

Assim como na subamostragem de itens frequentes, a amostragem negativa é uma técnica utilizada para aumentar a capacidade de generalização do modelo e torná-lo viável de ser executado. Há dois parâmetros que controlam a amostragem negativa: o número de amostras selecionadas $|G|$ e o expoente para controle da probabilidade de seleção γ, ambos explicados na Seção 3.2.2.

Ambos os parâmetros foram então avaliados. Para $|G|$, variou-se em $[3, 30]$, com passo variando entre 2 e 3. Já γ, recebeu como valores $[-1,0,\ 1,0]$, com passo de $0,25$. Os resultados para cada parâmetro encontram-se, respectivamente, nas Figuras 16 e 17.

Conjunto	Min	Q1	Mediana	Q3	Max	Q3:Q1	Max:Med
CiaoDVD	0,0009	0,0009	0,0010	0,0011	0,0012	17,51%	13,66%
Last.FM	0,0006	0,0010	0,0012	0,0015	0,0018	51,89%	43,19%

Tabela 23 – Sumário dos valores de NDCG@15 obtidos com o ajuste da quantidade de amostras negativas ($|G|$).

Colunas de mesmo significado que na Tabela 19.

Para a quantidade de amostras negativas, foi verificada uma variação significativa nos valores de NDCG apresentados, não havendo um consenso sobre um valor geral mais adequado. Para a base Last.FM, por exemplo, diferentes valores para $|G|$ podem alterar o resultado em mais do que 50%, como apresentado na Tabela 23, tornando-se assim um parâmetro interessante de ser ajustado.

O parâmetro γ, responsável por controlar a probabilidade de seleção das amostras negativas, foi o que obteve maior variação entre todos os parâmetros avaliados no

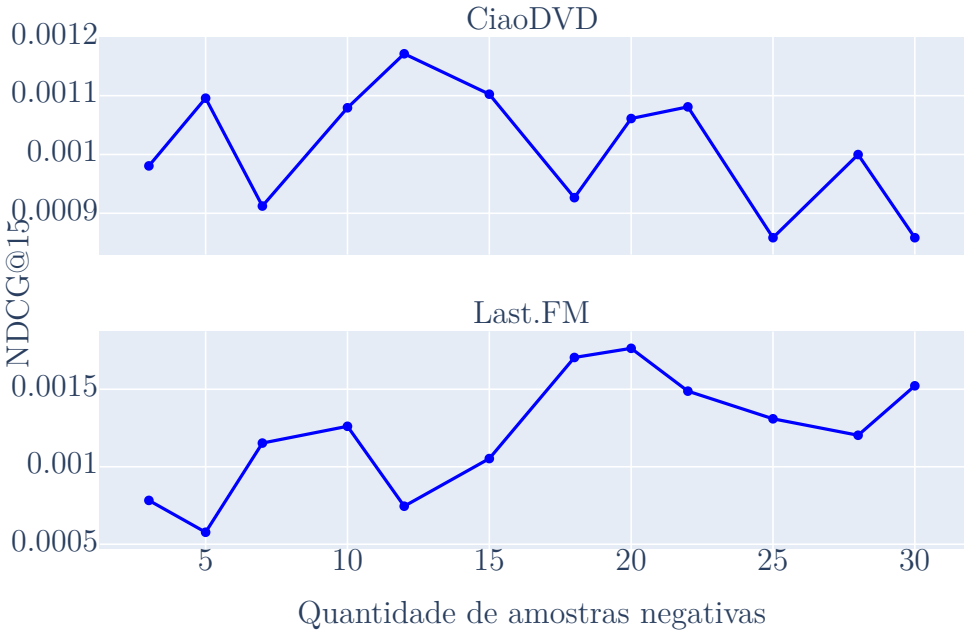

Figura 16 – NDCG@15 para diferentes valores de quantidade de amostras negativas ($|G|$)

experimento. Como apresentado na Tabela 24, para a base Last.FM diferentes valores para o parâmetro podem resultar em uma melhora de 5849,19%. Embora haja indícios que valores negativos comportam-se melhor, como ocorrido nas duas bases, uma variação de resultados com esta intensidade faz com que o expoente da seleção negativa seja um parâmetro de ajuste obrigatório para cada base de dados.

Conjunto	Min	Q1	Mediana	Q3	Max	Q3:Q1	Max:Med
CiaoDVD	0,0011	0,0011	0,0011	0,0014	0,0030	25,72%	166,59%
Last.FM	0,0004	0,0007	0,0070	0,0358	0,0558	5384,19%	700,15%

Tabela 24 – Sumário dos valores de NDCG@15 obtidos com o ajuste do expoente da distribuição negativa (γ).

Colunas de mesmo significado que na Tabela 19.

A.5 Fator de regularização

Por fim, como explicado na Seção 3.2.3, o fator de regularização λ é responsável por alterar a função de custo da rede neural com o objetivo de evitar o superajustamento. O parâmetro teve seu valor variado no intervalo $[10^{-6}, 10^{-1}]$, com paso de $\times 10$. Foi também

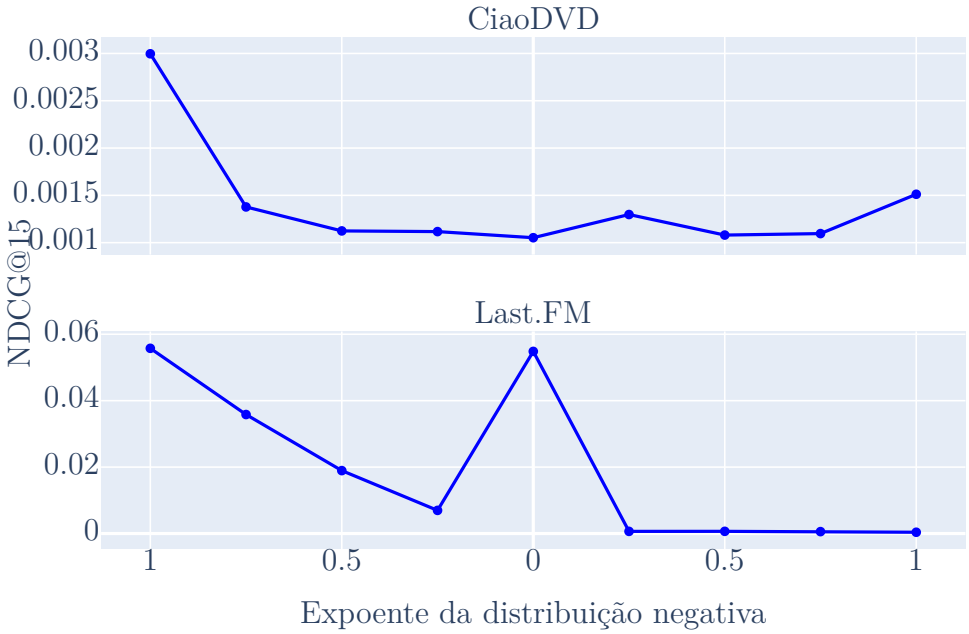

Figura 17 – NDCG@15 para diferentes valores de expoente da distribuição negativa (γ).

testado valor 0, ou seja, uma versão do modelo sem a regularização. Os resultados podem ser vistos na Figura 18

Os valores de NDCG obtidos indicam que a qualidade da representação pode ser impactada pela escolha de λ, gerando variações de até 56% no resultado final, como apresentado na Tabela 25. Entretanto, para as duas bases de dados selecionadas, o melhor resultado foi encontrado quando $\lambda = 10^{-1}$. Isso indica um comportamento interessante, e um valor que pode ser adotado como padrão em cenários onde não há a possibilidade de ajuste do parâmetro.

Conjunto	**Min**	**Q1**	**Mediana**	**Q3**	**Max**	**Q3:Q1**	**Max:Med**
CiaoDVD	0,0008	0,0010	0,0011	0,0012	0,0015	28,43%	40,51%
Last.FM	0,0006	0,0007	0,0010	0,0011	0,0011	56,19%	5,67%

Tabela 25 – Sumário dos valores de NDCG@15 obtidos com o ajuste do fator de regularização (λ).

Colunas de mesmo significado que na Tabela 19.

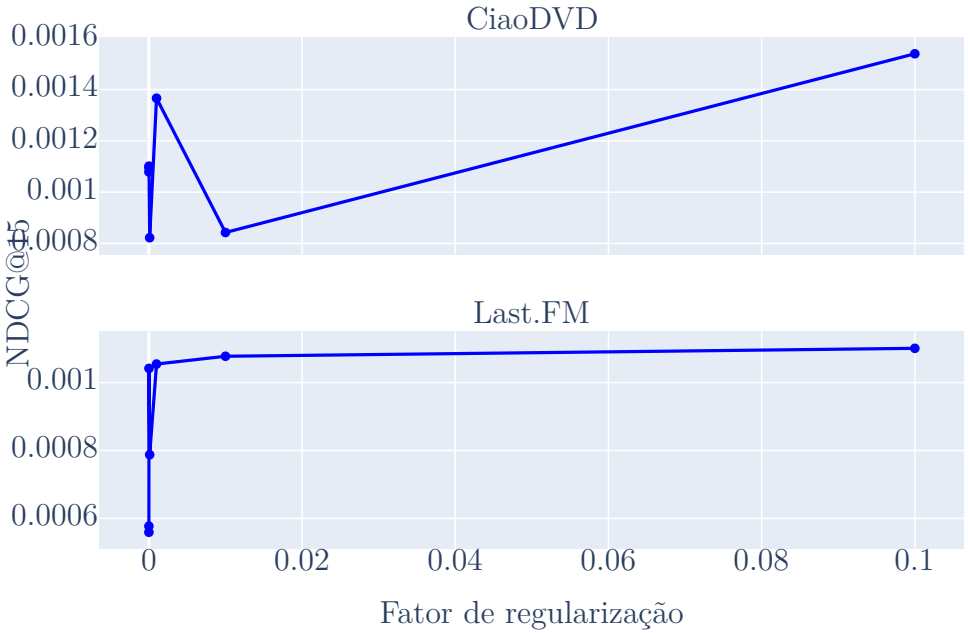

Figura 18 – NDCG@15 para diferentes valores de fator de regularização (λ).

A.6 Considerações finais

Com o objetivo de entender o impacto da seleção de hiper-parâmetros para o Interact2Vec, foi conduzida uma série de experimentos que avaliaram diferentes variações dos modelo para analisar empiricamente quais os parâmetros mais importantes de serem ajustados, e quais possuem valores ideais que podem ser tratados como padrão.

Ao término dos experimentos, é possível concluir que os parâmetros mais necessários de serem ajustados são: o expoente γ da seleção negativa, a quantidade $|G|$ de amostras negativas e a dimensionalidade M das RVDs. Para os demais parâmetros, embora alguns apresentem variações consideráveis nos resultados, existem valores específicos que se destacam em relação aos demais, podendo ser utilizados como valores padrões. São eles: $0,25$ para a taxa α de aprendizado; 50 para a quantidade M de iterações da rede; 10^{-6} para a taxa ρ de subamostragem de itens frequentes; e 10^{-1} para o fator λ de regularização.

APÊNDICE B – Melhores parâmetros para os métodos de recomendação obtidos por busca em grade

Para os experimentos finais, foi realizado, para cada base de dados, uma etapa de ajuste de parâmetros sobre cada método. Para a tarefa de ranqueamento top-N, a melhor combinação de parâmetro foi aquela que obteve o maior NDCG@15 sobre a base de validação. Para a tarefa de previsão de notas, a melhor combinação foi aquela que obteve o menor RMSE. Os melhores parâmetros por conjunto de dados, método de recomendação e tarefa encontram-se apresentados nas Tabelas 26 a 36.

Conjunto	f	λ
Anime	200	10^{-2}
BestBuy	200	10^{-2}
Book-Crossing	200	10^{-2}
CiaoDVD	50	10^{-6}
DeliciousBookmarks	200	10^{-6}
Filmtrust	50	10^{-4}
Last.FM	50	10^{-2}
MovieLens	50	10^{-6}
NetflixPrize	50	10^{-4}
RetailRocket	200	10^{-2}

Tabela 26 – Melhores parâmetros para o SVD no ranqueamento top-N.

Conjunto	f	λ
Anime	200	10^{-6}
BestBuy	100	10^{-4}
Book-Crossing	50	10^{-2}
CiaoDVD	50	10^{-2}
DeliciousBookmarks	50	10^{-4}
Filmtrust	200	10^{-2}
Last.FM	100	10^{-2}
MovieLens	50	10^{-2}
NetflixPrize	50	10^{-2}
RetailRocket	100	10^{-2}

Tabela 27 – Melhores parâmetros para o BPR no ranqueamento top-N.

Conjunto	C	ρ	γ
Anime	100	10^{-3}	0,5
BestBuy	50	10^{-5}	1,0
Book-Crossing	150	10^{-3}	1,0
CiaoDVD	100	10^{-4}	1,0
DeliciousBookmarks	150	10^{-5}	-0,5
Filmtrust	50	10^{-3}	-0,5
Last.FM	150	10^{-3}	1,0
MovieLens	150	10^{-3}	1,0
NetflixPrize	150	10^{-4}	1,0
RetailRocket	50	10^{-3}	-1,0

Tabela 28 – Melhores parâmetros para o Item2Vec no ranqueamento top-N.

Conjunto	C	ρ	γ
Anime	150	10^{-5} 0,5	
BestBuy	150	10^{-5}	0,5
Book-Crossing	150	10^{-4}	-0,5
CiaoDVD	100	10^{-4}	0,5
DeliciousBookmarks	150	10^{-3}	1,0
Filmtrust	100	10^{-4}	-0,5
Last.FM	100	10^{-5}	-0,5
MovieLens	50	10^{-5}	0,5
NetflixPrize	50	10^{-3}	1,0
RetailRocket	150	10^{-5}	-1,0

Tabela 29 – Melhores parâmetros para o User2Vec no ranqueamento top-N.

Conjunto	M	$\|G\|$	γ	**Método de rec.**	**Params. do método de rec.**
Anime	100	5	0,5	*Sims. ponderadas*	$\beta = 0,9$ e $\mu = 0,1$
BestBuy	100	5	1,0	*Sim. item-item*	-
Book-Crossing	100	5	1,0	*Comb. de RVDs*	Média com $K = \|U_i\|$
CiaoDVD	50	5	-0,5	*Sims. ponderadas*	$\beta = 0,9$ e $\mu = 0,1$
DeliciousBookmarks	100	5	-1,0	*Comb. de RVDs*	Média com $K = \|U_i\|$
Filmtrust	50	5	-0,5	*Sim. item-item*	-
Last.FM	100	5	-0,5	*Sims. ponderadas*	$\beta = 0,75$ e $\mu = 0,25$
MovieLens	100	5	-0,5	*Sims. ponderadas*	$\beta = 0,75$ e $\mu = 0,25$
NetflixPrize	50	5	-1,0	*Comb. de RVDs*	Média com $K = \|U_i\|$
RetailRocket	100	5	-1,0	*Sims. ponderadas*	$\beta = 0,1$ e $\mu = 0,9$

Tabela 30 – Melhores parâmetros para o Interact2Vec no ranqueamento top-N.

Conjunto	K
Anime	60
Book-Crossing	80
CiaoDVD	20
Filmtrust	20
MovieLens	20
NetflixPrize	20

Tabela 31 – Melhores parâmetros para o KNN na previsão de notas.

Conjunto	f	λ
Anime	50	10^{-2}
Book-Crossing	50	10^{-2}
CiaoDVD	50	10^{-2}
Filmtrust	50	10^{-2}
MovieLens	50	10^{-2}
NetflixPrize	50	10^{-2}

Tabela 32 – Melhores parâmetros para o SVD na previsão de notas.

Conjunto	f	λ
Anime	50	10^{-6}
Book-Crossing	100	10^{-4}
CiaoDVD	50	10^{-2}
Filmtrust	100	10^{-2}
MovieLens	50	10^{-4}
NetflixPrize	50	10^{-4}

Tabela 33 – Melhores parâmetros para o FM na previsão de notas.

Conjunto	C	ρ	γ	K
Anime	50	10^{-4}	-0,5	80
Book-Crossing	50	10^{-5}	-0,5	20
CiaoDVD	50	10^{-5}	-1,0	20
Filmtrust	50	10^{-4}	-0,5	100
MovieLens	150	10^{-3}	1,0	100
NetflixPrize	50	10^{-3}	1,0	20

Tabela 34 – Melhores parâmetros para o Item2Vec na previsão de notas.

Conjunto	C	ρ	γ	K
Anime	100	10^{-3}	0,5	20
Book-Crossing	50	10^{-4}	1,0	20
CiaoDVD	150	10^{-3}	1,0	20
Filmtrust	100	10^{-5}	-1,0	100
MovieLens	150	10^{-3}	-1,0	20
NetflixPrize	50	10^{-4}	-0,5	20

Tabela 35 – Melhores parâmetros para o User2Vec na previsão de notas.

Conjunto	M	$\|G\|$	γ	K
Anime	50	10	-0,5	20
Book-Crossing	150	15	-1,0	20
CiaoDVD	100	5	1,0	20
Filmtrust	100	5	1,0	100
MovieLens	100	5	-0,5	20
NetflixPrize	100	15	1,0	20

Tabela 36 – Melhores parâmetros para o Interact2Vec na previsão de notas.