UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

BRUNO SANTANA MASSENA DE LIMA

Análise de Aplicabilidade de Recomendações Baseadas em Grafos para Conteúdos Multimídia do Globoplay

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Orientador: Prof. Dr. Weverton Luís da Costa Cordeiro
Co-orientadora: Profa. Dra. Renata Galante

Porto Alegre
2022

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Reitor: Prof. Carlos André Bulhões
Vice-Reitora: Profª. Patricia Pranke
Pró-Reitora de Graduação: Profª. Cíntia Inês Boll
Diretora do Instituto de Informática: Profª. Carla Maria Dal Sasso Freitas
Coordenador do Curso de Ciência de Computação: Prof. Marcelo Walter
Bibliotecário-chefe do Instituto de Informática: Alexsander Borges Ribeiro

RESUMO

Com a Internet cada vez mais presente no dia-a-dia das pessoas, cresce cada vez mais a quantidade de informações disponíveis ao usuário. Com o objetivo de personalizar a experiência do usuário, são utilizados sistemas de recomendação para apresentar a ele apenas informações consideradas relevantes. Este trabalho tem como objetivo geral analisar a aplicabilidade de diferentes sistemas de recomendação baseados em grafos propostos na literatura sobre um conjunto de dados da plataforma de *streaming* digital Globoplay. Inicialmente, foi realizada uma análise da literatura para encontrar trabalhos com código fonte disponível e replicáveis. Em seguida, foi feita uma modelagem do conjunto de dados fornecido em um grafo de conhecimento. Por fim, os *frameworks* foram treinados com o grafo de conhecimento gerado e foi realizada uma análise individual dos resultados, assim como uma comparação geral entre os trabalhos. Os experimentos realizados demonstraram bons resultados, indicando grande potencial destes modelos quando aplicados ao conjunto de dados do Globoplay. Outro ponto observado foi a tendência que os sistemas de recomendação baseados em grafos têm de fornecer boas recomendações em um contexto de conteúdos de mídia audiovisual. Além disso, também foi observado que um aumento na quantidade de informações utilizadas para o treinamento destes modelos tem grande potencial para melhorar ainda mais os resultados obtidos.

Palavras-chave: Sistema de Recomendação. Grafo de Conhecimento.

Name: Applicability Analysis of Graph-Based Recommendations for Globoplay's Multimedia Content

ABSTRACT

With the internet increasingly present in people's daily lives, the amount of information available to the user grows more and more. In order to personalize the user experience, recommender systems are used to present only information that the user considers relevant. The general goal of this work is to analyze the applicability of different graph-based recommendation systems proposed in the literature on a dataset from the Globoplay digital streaming platform. Initially, a literature review was performed to find works with available source code and replicable. Then, the provided dataset was modeled into a knowledge graph. Finally, the *frameworks* were trained with the generated knowledge graph and an individual analysis of the results was performed, as well as a general comparison between the works.

The experiments performed showed good results, indicating great potential of these models when applied to the Globoplay dataset. Another point observed was the tendency of graph-based recommendation systems to provide good recommendations in a context of audiovisual media content. In addition, it was also observed that an increase in the amount of information used for training these models has great potential to further improve the results obtained.

Keywords: Recommender System. Knowledge Graph.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

SR	Sistema de Recomendação
GC	Grafo de Conhecimento
SRGC	Sistema de Recomendação baseado em Grafos de COnhecimento
KGE	*Knowledge Graph Embedding*
CSV	*Comma Sepparated Values*

SUMÁRIO

1 INTRODUÇÃO

Com a Internet cada vez mais presente no dia-a-dia das pessoas, a quantidade de informações disponíveis vem crescendo exponencialmente. Essa alta quantidade de informações impede que o usuário encontre algo de seu interesse com facilidade, ocasionando frustração ou até mesmo desistência. Para contornar este problema, é necessário personalizar a experiência de cada usuário individualmente, apresentando somente informações que sejam relevantes a ele no momento. Com isso, surge a necessidade de se utilizar sistemas de recomendação, que tem como objetivo gerar recomendações de itens de interesse do usuário.

Estas recomendações podem ser geradas com base em diversos fatores, como seu histórico de interações e semelhanças com outros usuários. Dentre os diversos métodos de representar conhecimento e informações secundárias, os grafos de conhecimento se destacam por serem uma abordagem prática para representar grandes quantidades de informações dentre múltiplos domínios (EHRLINGER; WÖSS, 2016). Utilizando os conceitos de sistemas de recomendação e de grafos de conhecimento, surgem os sistemas de recomendação baseados em grafos de conhecimento, que se utilizam do poder de representação de conhecimento dos grafos para treinar modelos que gerem recomendações relevantes ao usuário.

Este trabalho tem como objetivo geral analisar a aplicabilidade de diferentes sistemas de recomendação baseados em grafos propostos na literatura sobre um conjunto de dados da plataforma de *streaming* digital Globoplay. A análise realizada envolve as seguintes etapas:

- Explorar a literatura em busca de *frameworks* de recomendação com código fonte disponível e replicáveis.
- Modelar o conjunto de dados fornecido em um grafo de conhecimento.
- Analisar e comparar as execuções dos *frameworks* com o conjunto de dados fornecido.

Este trabalho está estruturado da seguinte forma. O Capítulo 2 apresenta os principais conceitos que baseiam este trabalho. O Capítulo 3 lista e compara trabalhos que se assemelham à este. O Capítulo 4 apresenta

a metodologia utilizada e as etapas necessárias para alcançar o objetivo aqui proposto. O Capítulo 5 apresenta os resultados obtidos ao aplicar a metodologia descrita anteriormente, assim como uma comparação entre estes resultados. Por fim, o Capítulo 6 apresenta conclusões gerais sobre este trabalho e direções futuras.

2 FUNDAMENTAÇÃO TEÓRICA

Neste Capítulo são abordados conceitos importantes utilizados para a elaboração deste trabalho e necessários para seu entendimento. A Seção 2.1 apresenta definições e conceitos relacionados a grafos de conhecimento. A Seção 2.2 apresenta definições e conceitos relacionados a sistemas de recomendação.

2.1 Grafos de conhecimento

Grafos de conhecimento (GC) são uma abordagem prática para representar um grande conjunto de relações entre diferentes tipos de entidades. Os nós são os elementos básicos de um grafo de conhecimento, utilizados para representar entidades. Estas entidades podem representar qualquer conceito definido na modelagem do GC, como objetos, pessoas, filmes, entre outros. Já as arestas de um GC, são utilizadas para representar relações entre estas entidades, como “assistiu”, “jogou”, “comprou”, entre outras.

Os GC são comumente utilizados seguindo as definições do padrão RDF (*Resource Definition Framework*) (KLYNE; CARROLL, 2004), no qual o grafo é composto por tuplas no formato (*subject,predicate,object*), sendo que *subject* e *object* referem-se a entidades e *predicate* refere-se à relação entre estas entidades. Um exemplo de tupla em um grafo de conhecimento seria (Yuri, comprou, iPhone). Nessa tupla, “Yuri” é o nó que representa uma entidade do tipo pessoa, sendo essa a entidade origem da tupla. “iPhone” é o nó que representa uma entidade do tipo produto, sendo essa a entidade destino da tupla. Por fim, “comprou” representa a relação entre as entidades origem e destino, indicando que Yuri comprou iPhone.

2.2 Sistemas de recomendação

Sistemas de recomendação (SR) são ferramentas que buscam sugerir itens que possuam a maior probabilidade de serem relevantes para

determinado usuário (CHEW et al., 2020). Com a existência de SR, usuários podem ter uma experiência totalmente customizada com base em informações extraídas de suas interações com o sistema, como avaliações, histórico de pesquisa, itens favoritos, entre outros. Ao utilizar um bom SR, uma empresa pode melhorar significativamente suas vendas, além de proporcionar uma melhor experiência para o usuário.

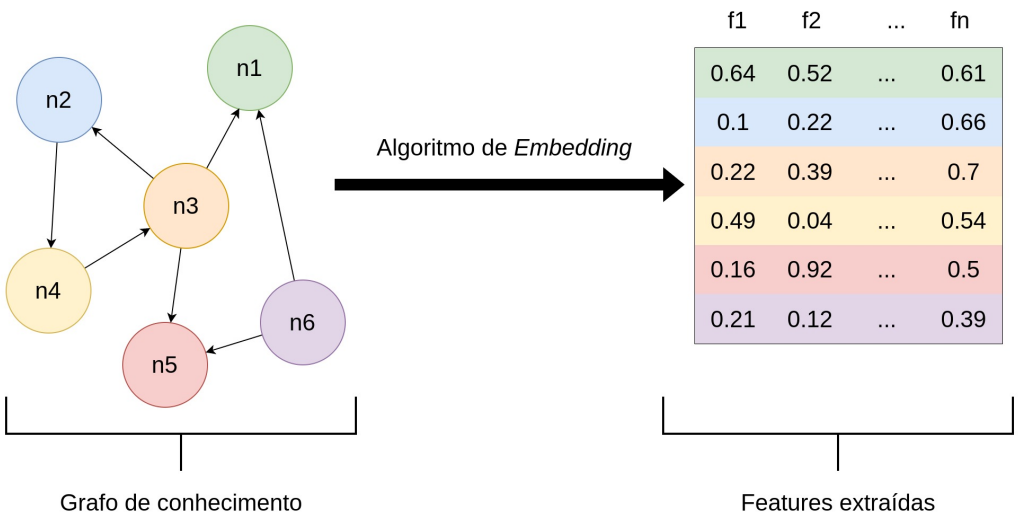
Os sistemas de recomendação podem ser divididos em 3 principais tipos, dependendo da técnica utilizada: SR baseados em filtragem colaborativa, SR baseados em conteúdo e SR híbridos. Os baseados em filtragem colaborativa historicamente são os mais utilizados, porém, com os avanços na área de *deep learning*, estes estão perdendo espaço para técnicas mais sofisticadas, como as híbridas.

Com o intuito de agregar outras informações diversas sobre os usuários, itens, suas relações e o contexto em que estão inseridos, podemos agregar o uso de grafos de conhecimento em conjunto com sistemas de recomendação. Além de possibilitar a modelagem de *feedbacks* dos usuários sobre os itens, é possível incluir entidades que não se encaixam em nenhum desses grupos, mas provém informações de suma importância quando procuram-se relações que possam enriquecer as recomendações geradas.

2.2.1 Sistemas de recomendação baseados em grafos de conhecimento baseados em *embeddings*

Uma possível abordagem para a utilização de grafos de conhecimento em sistemas de recomendação são os métodos baseados em *embeddings*. Estes métodos transformam cada nó do grafo de conhecimento em um vetor de números reais, chamado de *embedding*, de baixa dimensionalidade que codifica a informação estrutural do grafo. Portanto, são extraídas informações do grafo de conhecimento que ajudam a enriquecer a representação dos usuários e itens analisados (GUO et al., 2020). A Figura 2.1 exemplifica o funcionamento de um algoritmo de KGE (*Knowledge Graph Embedding*), recebendo um grafo de conhecimento como entrada e retornando um conjunto de vetores de tamanho n, sendo n uma quantidade arbitrária de *features* extraídas para as entidades modeladas.

Figura 2.1: Exemplo de algoritmo de *Embedding*



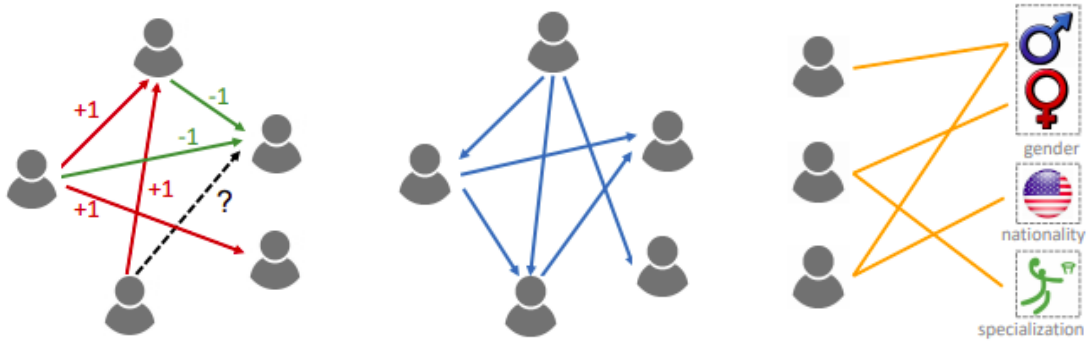
Fonte: Elaborada pelo autor.

Os algoritmos de KGE podem ser divididos em duas categorias, dependendo se os usuários estão inclusos no grafo de conhecimento ou não. A primeira categoria engloba métodos em que o grafo de conhecimento é construído com base nos itens e seus atributos extraídos do conjuntos de dados ou de fontes externas, e os usuários acabam ficando fora deste grafo. Este tipo de algoritmo de KGE tem como objetivo modelar as características dos itens analisados e adicionar informações extras sobre os mesmos, podendo ser feito o uso de bases de conhecimento externas. Como exemplo da utilização desse método temos o Collaborative Knowledge Base Embedding (CKE), proposto por ZHANG et al. (2016). Este método é baseado em *Collaborative Filtering* (CF) e utiliza diversos tipos de informações sobre os itens para enriquecer suas representações no grafo de conhecimento, podendo ser eles textuais, como uma sinopse, ou visuais, como um pôster de filme.

A outra categoria de algoritmos de KGE incluí métodos em que os usuários estão inclusos no grafo de conhecimento, em conjuntos com os itens e seus atributos. Neste cenário, ambas as relações de atributo e de usuários são modeladas como arestas no grafo de conhecimento e utilizadas para gerar *embeddings*. Como exemplo de aplicação deste método, temos o

SHINE, proposto por WANG et al. (2018a), que propõe a recomendação de celebridades para usuários em redes sociais com base em ligações sentimentais. Neste algoritmo, são construídos três grafos de conhecimento para representar todas as relações necessárias: o primeiro representa as ligações sentimentais entre usuários e celebridades (*sentimental network*), o segundo modela as relações sociais entre os usuários da rede (*social network*), e o terceiro inclui as informações extraídas dos perfis dos usuários analisados (*profile network*). Após a modelagem, é aplicado um algoritmo de *embedding* nestes três grafos utilizando uma técnica de *autoencoder* e as predições podem ser geradas com predição de *links*. Os *autoencoders* são uma técnica de aprendizado não supervisionado em que se utiliza redes neurais para realizar o aprendizado de representação (MICHELUCCI, 2022). As modelagens dos grafos de conhecimento de *sentimental network*, *social network* e *profile network* podem ser observadas na Figura 2.2.

Figura 2.2: Grafos de conhecimento modelados pelo SHINE



(a) *Sentimental network* (b) *Social network* (c) *Profile network*

Fonte: WANG et al. (2018a)

2.2.2 Sistemas de recomendação baseados em grafos de conhecimento baseados em caminhos

Outra possível abordagem para gerar recomendações baseadas em grafos de conhecimento são os métodos baseados em caminhos. Nestes métodos, assim como nos baseados em *embeddings*, são construídos grafos de conhecimento que modelam as relações entre itens, usuários e seus possíveis atributos. Porém, estes métodos analisam os padrões de

conectividade dos usuários e itens no grafo e os utilizam para prever *links* de acordo com suas similaridades e assim gerar recomendações. Diferente dos métodos de *embedding*, estes podem gerar recomendações facilmente explicáveis, utilizando como base os padrões de caminhos observados.

Para analisar a similaridade entre entidades, estes métodos utilizam uma modelagem de meta-caminhos explicitamente definidos com base no contexto dos dados analisados. Os meta-caminhos são utilizados para descrever relações semânticas de alto nível entre duas entidades. Por exemplo, um meta-caminho “usuário-filme-gênero-filme-usuário” modela uma relação semântica entre dois usuários que assistiram filmes do mesmo gênero, podendo indicar uma possível similaridade entre eles.

Como exemplo da aplicação de métodos baseados em caminhos temos o HeteRec, proposto por YU et al. (2013). Nele, é proposto que seja construído um *framework* de recomendação baseado em *feedback* implícito do usuário, extraindo *features* latentes para se aproveitar de múltiplas semânticas e fatores de recomendação. Para se beneficiar da heterogeneidade da informação, HeteRec explora a similaridade entre meta-caminhos dos nodos do grafo com objetivo de enriquecer a matriz de interação “usuário-item”.

Para tentar contornar o problema da modelagem manual de meta-caminhos, MA et al. (2019) propõe RuleRec para tentar extrair os meta-caminhos de modo automático. Para isso, é utilizado um módulo de extração de regras, que busca computar a probabilidade de encontrar caminhos que respeitem certas regras entre pares de itens com base em um algoritmo de passeio aleatório. Esta abordagem permite que as recomendações geradas sejam explicáveis com base nos meta-caminhos gerados e ainda mantém o *framework* de recomendação genérico, excluindo a necessidade de gerar regras de associação entre itens e usuários manualmente.

2.2.3 SRGCs unificados

Ambos os métodos descritos anteriormente trazem suas vantagens e desvantagens com base na metodologia utilizada. De um lado temos a

flexibilidade dos métodos baseados em *embeddings*, que permitem gerar recomendações com base em qualquer estrutura de rede de informação heterogênea, mas carecem de interpretabilidade nas suas recomendações. Do outro lado os métodos baseados em caminhos resolvem o problema da explicabilidade das recomendações, mas necessitam de uma modelagem manual dos meta-caminhos que serão utilizados. Com objetivo de unir os pontos fortes de ambas as abordagens, foram criados os métodos unificados.

Para se aproveitar tanto da representação semânticas de usuários, itens e suas relações quanto das informações estruturais de conectividade entre eles, os métodos unificados se baseiam na ideia de propagação de *embeddings* (também chamado de propagação de preferência, *preference propagation*). Assim como nos métodos baseados em *embeddings*, estes são gerados para cada entidade representada, e logo após é feita a propagação dessa informação para enriquecer a representação de entidades vizinhas no grafo de conhecimento, fazendo uso da informação estrutural do mesmo, como nos métodos baseados em caminhos.

Como exemplo da aplicação de métodos unificados temos o AKUPM, proposto por TANG et al. (2019). Nele o autor modela o usuário com seu histórico de cliques. Primeiro é aplicado um algoritmo de *embedding* às entidades, então essa informação é propagada através dos nodos com o objetivo de aprender a relação entre as entidades por meio de uma camada chamada de *self-attention*, que busca entender as preferências do usuário com base em atributos mais relevantes.

3 TRABALHOS RELACIONADOS

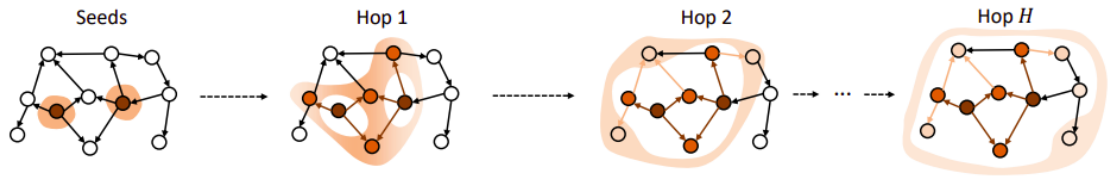
Neste Capítulo são apresentadas as principais contribuições sobre os trabalhos analisados experimentalmente neste trabalho.

3.1 *RippleNet: Propagando preferências do usuário no grafo de conhecimento para gerar recomendações*

Proposto por WANG et al. (2018b), RippleNet foi definido pelos autores como primeiro trabalho a combinar métodos baseados em *embeddings* e métodos baseados em caminhos para gerar recomendações, que posteriormente foram nomeados métodos unificados. Como descrito na Seção 2.2.3, essa combinação de métodos foi possível devido à uma técnica chamada de propagação de preferência (*preference propagation*), utilizada pela primeira vez no contexto de SRGCs pelo RippleNet. Esta técnica é descrita como sendo a ideia chave por trás do RippleNet, e tem como objetivo propagar as preferências do usuário iterativamente através das entidades do grafo de conhecimento por meio de suas conexões.

A Figura 3.1 exemplifica o funcionamento geral do algoritmo de propagação de preferência. Cada item com o qual o usuário já interagiu é chamado de semente e é utilizado como início da propagação no grafo de conhecimento. Cada iteração da propagação (tambem chamado de *hop*) gera um conjunto de entidades com base nas *seeds* chamado de *ripple set*. Estes *ripple sets* são utilizados para calcular o *embedding* do usuário com relação ao item em questão, e por fim são combinados os *embeddings* do usuário e item para gerar uma probabilidade do usuário interagir com este item.

Figura 3.1: Propagação de preferência no RippleNet. A propagação é iniciada a partir das *seeds* e os *embeddings* das entidades são atualizados iterativamente.



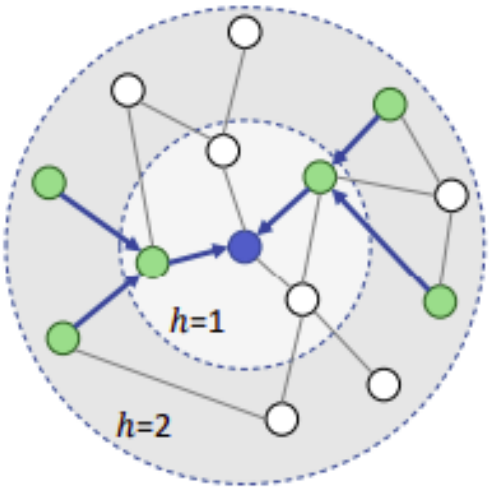
Fonte: WANG et al. (2018b)

Mesmo sendo o primeiro trabalho a aplicar o conceito de método unificado em sistemas de recomendação baseados em grafos, o RippleNet já demonstrou uma melhora expressiva nos resultados das recomendações obtidas em comparação com outros métodos considerados estado da arte. Foram analisados os conjuntos de dados *MovieLens-1M*, *Book-Crossing* e *Bing-News*, e, em todos estes, o RippleNet se mostrou superior em performance com relação à outros métodos.

3.2 *KGCN: Redes convolucionais com grafos de conhecimento para sistemas de recomendação*

Também chamado de KGCN, este trabalho proposto por WANG et al. (2019a) possui um mecanismo de propagação de preferência similar ao RippleNet. O *framework* proposto modela a representação de um item no grafo de conhecimento agregando os *embeddings* de itens vizinhos distantes para o próprio item em questão. A grande diferença com relação ao RippleNet é o sentido em que ocorre a propagação de preferência. Enquanto no RippleNet a propagação ocorre de dentro para fora, no KGCN esta ocorre de fora para dentro como pode ser observado na Figura 3.2.

Figura 3.2: Propagação de preferência no KGCN.



Fonte: WANG et al. (2019a)

Para conseguir captar ao mesmo tempo informações semânticas e estruturais do grafo, os autores extraem uma amostra de vizinhos para cada

entidade no grafo de conhecimento e os definem como seu campo receptivo, e então combinam as informações desses nodos vizinhos com viés para gerar a representação de dada entidade. Esses vizinhos selecionados não são necessariamente vizinhos diretos da entidade no grafo de conhecimento, o que significa que podem estar conectados através de um número arbitrário de *hops* no GC.

3.3 *DSKE: Destilando conhecimento estruturado em embeddings para gerar recomendações explicáveis e precisas*

Este trabalho combina as principais vantagens dos métodos baseados em *embeddings* e dos métodos baseados em *caminhos* de forma diferente. Nele, ZHANG et al. (2019) propõe que modelos baseados em *embeddings* previamente treinados possam ser acoplados à um *framework* baseado em caminhos com o objetivo de atingir o melhor dos dois mundos.

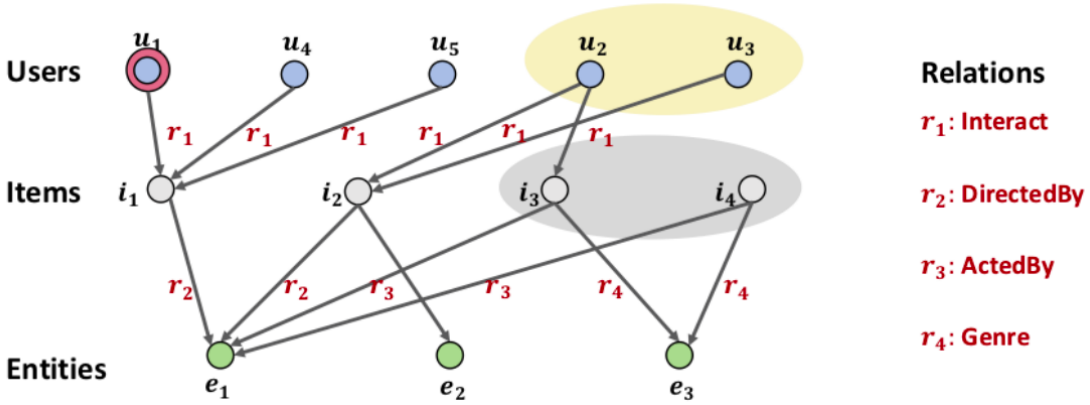
Os autores propõe um *framework* de aprendizado em conjunto que consegue acoplar um modelo que gera recomendações caixa-preta baseado em *embeddings* a um modelo baseado em caminhos que interpreta as recomendações geradas com base em um grafo de conhecimento e gera explicações para elas. Além disso, este modulo baseado em caminhos também regulariza o modelo caixa-preta com informações estruturais do grafo de conhecimento para aprimorar suas recomendações.

3.4 KGAT: *Recomendações com redes de atenção em grafos de conhecimento*

Outros trabalhos citados anteriormente como o RippleNet e o KGCN modelam um grafo de conhecimento com foco nos itens e seus atributos, excluindo os usuários da estrutura gerada. O KGAT, proposto por WANG et al. (2019b), utiliza uma estrutura híbrida denominada grafo de conhecimento colaborativo (CKG), que é uma associação entre o grafo de conhecimento dos itens e um grafo “usuário-item” que modela o histórico de interações do usuário.

Além disso, o *framework* proposto modela diretamente as relações de alta ordem entre usuários e itens, utilizando também o conceito de propagação de *embeddings* para enriquecer a representação destas entidades. Por meio da Figura 3.3, o autor do KGAT exemplifica como entidades importantes em um contexto de recomendação poderiam passar despercebidas caso não sejam modeladas as relações de alta ordem. Ao gerar recomendações para o usuário u_1, os usuários em amarelo u_2 e u_3 seriam relevantes, pois assistiram outros filmes dirigidos pela mesma pessoa e_1. Da mesma forma, os itens em cinza i_3 e i_4 seriam relevantes, pois compartilham outras relações em comum com e_1.

Figura 3.3: Entidades que passariam despercebidas caso não houvessem relações de alta ordem modeladas.



Fonte: WANG et al. (2019b)

3.5 *ECFKG: Aprendendo embeddings de bases de conhecimento heterogêneas para recomendações explicáveis*

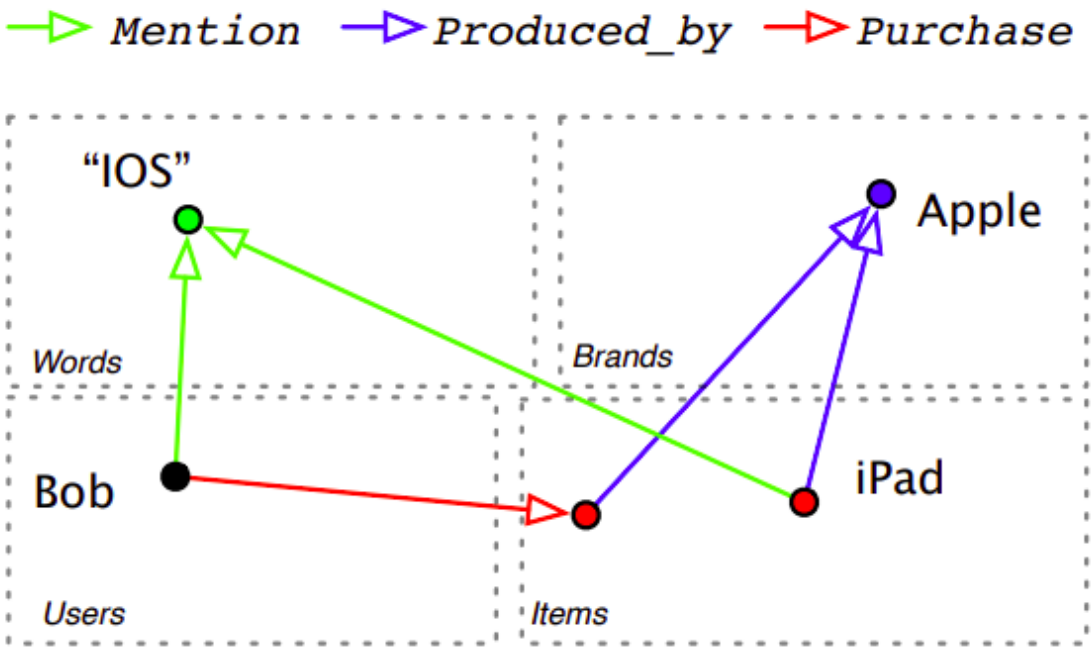
Também chamado de ECFKG (*Explainable Collaborative Filtering over Knowledge Graph*), este artigo proposto por AI et al. (2018) se diferencia dos outros citados anteriormente por utilizar um método de filtragem colaborativa para gerar os *embeddings* e consequentemente as recomendações. No ECFKG, assim como no KGAT, o grafo de conhecimento é gerado contendo usuários e itens, com foco em incorporar todo o conhecimento sobre os itens em conjunto com o histórico de comportamento dos usuários.

Para gerar boas recomendações se utilizando das informações

disponíveis no grafo de conhecimento, este trabalho propõe estender o uso do método de filtragem colaborativa tradicional para conseguir aprender com base em conhecimento heterogêneo, capturando as preferências do usuário de forma mais abrangente. Além disso, também é proposto um algoritmo para gerar explicações às recomendações geradas com base em caminhos no grafo de conhecimento. De forma simplificada, dado um usuário u e um item recomendado i, este algoritmo percorre o grafo de conhecimento e encontra potenciais caminhos ligando essas duas entidades. Logo após, é computada a probabilidade de cada caminho ser uma boa explicação para tal recomendação, e o melhor dentre eles é selecionado. Tendo em mãos o melhor caminho, este algoritmo gera uma explicação em linguagem natural para explicar coerentemente a recomendação gerada.

Para melhor ilustrar como estas explicações são geradas, temos a Figura 3.4, que exemplifica um possível cenário em que o item *iPad* foi recomendado ao usuário *Bob*. Observando as entidades e relações deste grafo, é possível extrair duas explicações para tal recomendação. A primeira seria por meio da palavra "IOS", que é mencionada em uma avaliação de outro produto comprado por *Bob* e também na descrição do produto recomendado. Outra possível explicação seria por meio da entidade *Apple*, que é a marca de um produto comprado anteriormente por *Bob* e também é a marca do produto *iPad* recomendado.

Figura 3.4: Exemplo de caminhos utilizados para explicar a recomendação do item *iPad* para o usuário *Bob*



Fonte: AI et al. (2018)

3.6 Resumo comparativo

A Tabela 3.1 apresenta um resumo comparativo dos trabalhos relacionados descritos nas seções anteriores.

Percebe-se que a grande maioria dos trabalhos analisados utiliza uma abordagem baseada em métodos unificados, esta que foi proposta e utilizada inicialmente pelo RippleNet (WANG et al.) e demonstrou um grande potencial na geração de recomendações. Além da indiscutível melhora de performance apresentada por esses métodos, eles também trazem a vantagem de gerar recomendações explicáveis, funcionalidade que está presente em todos os artigos analisados. O ECFKG, apesar de ser o único artigo analisado que utiliza apenas métodos baseados em *embeddings*, incorpora em seu *framework* um módulo de explicação de recomendações, funcionalidade amplamente utilizada em sistemas que necessitam de um módulo de recomendação, como serviços de *e-commerce*, *streaming* e redes sociais.

O tipo de modelagem utilizada no grafo de conhecimento também influencia no funcionamento destes *frameworks*. Enquanto uma modelagem

Tabela 3.1: Comparação entre os artigos

	Método	**Modelagem**	**Datasets**	**Métricas**
WANG et al. (2018b)	Unificado	Itens	MovieLens, Book-Crossing, Bing-News	Accuracy, AUC
WANG et al. (2019a)	Unificado	Itens	MovieLens, Book-Crossing, LastFM	Recall, F1, AUC
ZHANG et al. (2019)	Unificado	Itens	MovieLens, Pinterest, Yelp, Douban	Hit, Recall, NDGC
WANG et al. (2019b)	Unificado	Itens e usuários	Amazon-Book, LastFM, Yelp	Recall, NDGC
AI et al. (2018)	Embedding	Itens e usuários	CDs & Vinyl, Clothing, Cell Phones, Beauty	Precision, Hit, Recall, NDGC

Fonte: Elaborada pelo autor.

baseada somente em itens reduz a complexidade e simplifica o treinamento e funcionamento dos modelos, uma modelagem que inclui os usuários pode agregar atributos essenciais sobre o usuário que acabam facilitando o entendimento de suas preferências e melhorando as recomendações. Conjuntos de dados que contém muitas informações adicionais sobre o usuário tendem a se beneficiar muito mais de um grafo de conhecimento que inclua os mesmos, um bom caso para utilização desta modelagem seria em um contexto de rede social por exemplo.

4 METODOLOGIA

De modo geral, este trabalho visa analisar os *frameworks* de recomendação baseados em grafos de conhecimento listados no Capítulo 3 aplicados sobre o conjunto de dados Globoplay, tendo como objetivo gerar recomendações relevantes e coesas para o usuário final. Este capítulo aborda os principais pontos referentes à metodologia utilizada para elaboração deste trabalho, tendo como objetivo fornecer todo o conhecimento necessário para que os experimentos realizados possam ser replicados.

4.1 Visão geral

A metodologia dos experimentos deste trabalho foi concebida com o objetivo principal de analisar a aplicabilidade de diferentes *frameworks* de recomendação sobre o conjunto de dados Globoplay. Para isso, várias etapas e processos intermediários precisam ser aplicados. Como artefatos iniciais que darão início ao processo, temos o conjunto de dados bruto Globoplay, um conjunto de cinco *frameworks* de recomendação propostos por outros trabalhos da literatura e um conjunto de métricas *baseline* para comparação, estas que foram extraídas dos próprios trabalhos que propuseram os sistemas de recomendação analisados.

A Figura 4.1 apresenta o fluxograma contendo todas as etapas da metodologia proposta. A primeira etapa no fluxo é a de pré-processamento. Nessa etapa, são aplicados três algoritmos que tem como objetivo tornar o conjunto de dados apto a ser utilizado pelos *frameworks* para gerar recomendações: (1) algoritmo de agrupamento por usuário, que tem como objetivo agrupar as interações dos usuário em sequência, evitando a perda de informações ao particionar o conjunto de dados original em conjuntos menores, (2) algoritmo de amostragem, que tem como objetivo extrair um subconjunto do conjunto de dados original e (3) algoritmo de cálculo de avaliação, que tem como objetivo atribuir uma nota para cada interação do usuário com um item. Todos estes algoritmos são detalhados na Seção 4.4.

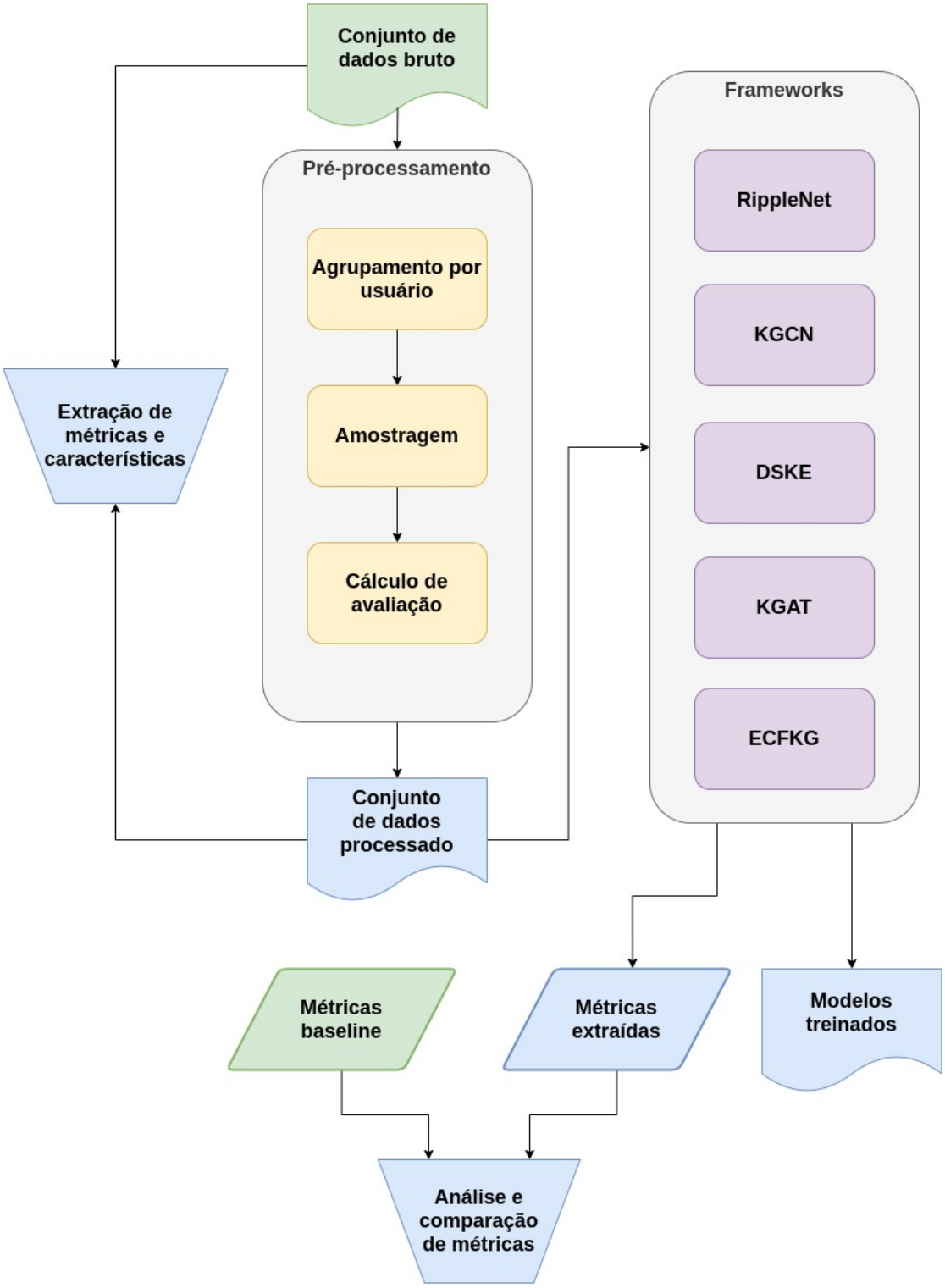
Em seguida, o conjuntos de dados tratado é utilizado como entrada para os *frameworks* de recomendação. Cada um desses *frameworks* exige

que o conjunto de dados seja modelado em diferentes arquivos de forma diferente antes de iniciar o processo de treinamento. Essa etapa de pré-processamento específica por sistema de recomendação é abordada no Capítulo 5. Com os arquivos de entrada prontos, os diferentes sistemas de recomendação podem ser treinados, cada um deles gerando um modelo de recomendação e um conjunto de métricas de avaliação.

Por fim, após finalizar o treinamento de todos os modelos, é feita uma etapa de análise das métricas extraídas. Nessa etapa, as métricas *baseline* fornecidas pelos trabalhos analisados são comparadas com as métricas extraídas do treinamento com o conjunto de dados Globoplay. Além disso, também são comparados os resultados dos *frameworks* entre sí, com o objetivo de entender vantagens, desvantagens e limitações de cada um deles.

Em paralelo com todas as etapas descritas anteriormente, também foi feita uma extração de métricas e análise de características sobre o conjunto de dados Globoplay antes e depois do pré-processamento. Essa análise tem como objetivo possibilitar que este conjunto de dados seja comparado com os utilizados pelos trabalhos analisados e fundamentar suposições que expliquem os resultados obtidos.

Figura 4.1: Fluxograma da metodologia.



Fonte: Elaborada pelo autor.

4.2 Conjunto de dados Globoplay

Para este trabalho foi fornecido um conjunto de dados do Globoplay contendo informações sobre a interação de usuários com mídias digitais da plataforma Globoplay, tais como documentários, séries, filmes notícias, entre outros. O conjunto de dados fornecido contém cerca de 200 milhões de linhas divididas igualmente entre 100 arquivos e identifica todas as entidades referenciadas com IDs, com objetivo de não incluir informações sensíveis sobre os usuários. Os dados foram fornecidos em formato csv, no qual cada linha representa uma interação de determinado usuário com uma mídia e é descrita pelos seguintes campos:

- **user_ID**: identificador de usuário.
- **media_ID**: identificador de mídia.
- **title_ID**: identificador de título.
- **playertype**: *player* utilizado para reproduzir a mídia.
- **deviceGroup**: dispositivo utilizado para reproduzir a mídia.
- **category**: categoria do título (entretenimento, notícias, entre outros).
- **genres**: gênero do título (aventura, ação, ficção, entre outros).
- **duration**: duração total da mídia.
- **played**: tempo total reproduzido da mídia pelo usuário.
- paused: tempo total em que a mídia ficou pausada.
- timecount
- playcount
- pausecount
- seekcount
- history
- maxtime
- timestamp: indica a data e hora em que o conteúdo foi assistido pelo usuário.

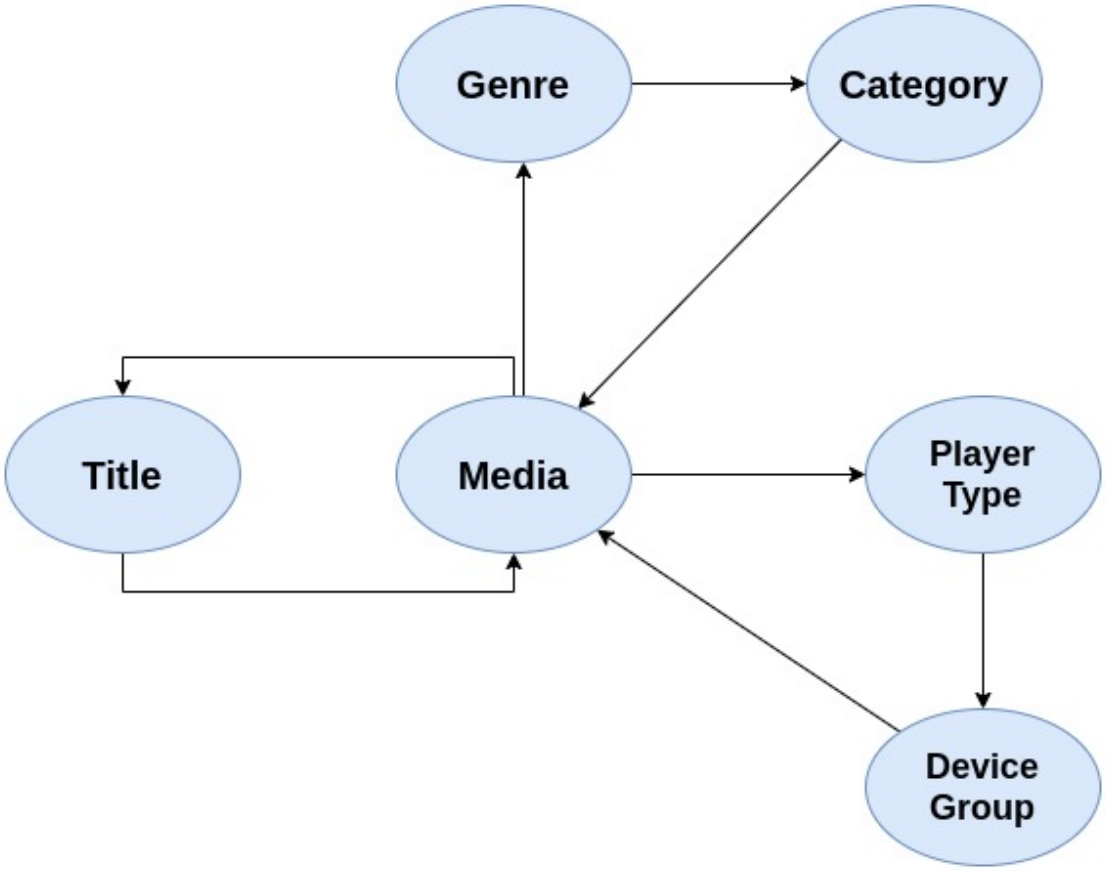
Os campos que estão destacados na lista foram selecionados empiricamente como sendo os mais relevantes para a modelagem dos dados, os demais campos foram desconsiderados por questão de simplicidade e

limitações de tempo e de performance dos *frameworks* analisados. Vale também resaltar que o conjunto de dados fornecido não está disponível publicamente, impossibilitando que os resultados sejam replicados por terceiros.

4.3 Modelagem do grafo de conhecimento

Para iniciar o processo de aplicação dos *frameworks* no conjunto de dados Globoplay, primeiro é necessário modelar um grafo de conhecimento abstrato que será utilizado como base de conhecimento para o treinamento dos sistemas de recomendação. Tendo em mãos a lista dos campos disponíveis do conjunto de dados, foram escolhidos os mais relevantes para compor uma base de conhecimento concisa. A Figura 4.2 representa a modelagem alvo que utilizaremos para representar as informações do conjunto de dados fornecido. Visto que a única informação relevante sobre o usuário no conjunto de dados é a sua interação com mídias, não existe a necessidade de o modelar como entidade no grafo de conhecimento.

Figura 4.2: Modelagem de grafo de conhecimento com relações direcionais.



Fonte: Elaborada pelo autor.

4.4 Pré-processamento dos dados

Tendo em mãos o conjunto de dados bruto e a modelagem do grafo de conhecimento, ainda é necessário realizar um pré-processamento dos dados antes de iniciar o treinamento dos *frameworks* com objetivo de evitar futuros problemas e melhorar os resultados obtidos. Um dos problemas que certamente atrapalharia a analise e o treinamento dos modelos é o tamanho do conjunto de dados bruto. Tendo 200 milhões de interações entre usuários e mídias, o processo de treinamento de alguns dos *frameworks* levaria dias ou até mesmo semanas para ser concluído. Por esse motivo, reduzimos a quantidade de interações utilizadas para treinar os modelos de 200 milhões para 2 milhões.

Com essa redução, foi constatado outro sério problema que afetaria a performance dos *frameworks*: a esparsidade dos dados. Como o conjunto de

dados bruto fornecido não possuía nenhum critério de ordenamento, as interações de um usuário poderiam estar espalhadas por todas as partes do arquivo original, ocasionando uma baixa proporção de interações por usuário caso fosse utilizada apenas parte do arquivo original. Para resolver este problema, realizamos um ordenamento do arquivo bruto utilizando o campo "user_ID" como chave de ordenamento. Como resultado, obtivemos um arquivo de mesmo tamanho que o original, mas que poderia ser facilmente seccionado em arquivos menores sem afetar a densidade de interações por usuário. Esse processo foi realizado com o uso da ferramenta de ordenamento *sort*, disponível em sistemas operacionais Linux.

Para quantificar a diferença na densidade de interações por usuário no conjuntos de dados, foram extraídas duas amostras de tamanho similar antes e depois da aplicação do algoritmo de agrupamento por usuários. Para cada amostra, foram extraídas informações sobre a quantidade de usuários, itens, e interações presentes na mesma e, em seguida, foi calculada a densidade de interações por usuário. Como podemos observar na Tabela 4.1, a quantidade de usuários em uma amostra de tamanho similar reduziu de 1,141,995 para 36,397, implicando em um aumento de quase 30 vezes na quantidade de interações por usuário, que aumentou de 1.92 para 54.94.

Tabela 4.1: Comparação de amostras do conjunto de dados Globoplay antes e depois do agrupamento por usuário

	Globoplay (sem agrupamento)	**Globoplay** (com agrupamento)
# usuários	1,141,995	36,397
# itens	32,117	30,608
# interações	2,195,785	2,000,000
interações/usuário	1.92	54.94

Fonte: Elaborada pelo autor.

Por fim, para conseguir treinar os *frameworks* de recomendação com o

conjunto de dados Globoplay, é necessário que cada interação contida nele possua um atributo que represente uma avaliação do usuário sobre a mídia assistida. Este atributo deve estar no formato de uma nota com valor inteiro em uma escala entre 0 e 4, sendo 0 a pior nota e 4 a melhor nota. Como o conjunto de dados Globoplay não possui nenhum atributo que indique a satisfação do usuário com o conteúdo consumido, é necessário extrair ou calcular este valor com base nos campos que temos acesso.

Para resolver este problema, propõe-se duas abordagens para realizar o cálculo das notas. A primeira e mais simples consiste em gerar números aleatórios na faixa desejada e atribuir um desses números para cada interação presente no conjunto de dados. Como não sabemos a real avaliação do usuário, é plausível que estas sejam atribuídas aleatoriamente. Essa abordagem foi dividida em duas diferentes implementações, variando como esses números são gerados: a primeira consiste em utilizar a função “randint” para gerar números inteiros pseudo-aleatórios na faixa desejada. Já a segunda implementação utiliza a função “uniform” para gerar números pseudo-aleatórios em função de uma distribuição uniforme. Ambas as funções estão disponíveis no módulo “random” da linguagem de programação Python.

A outra abordagem proposta para calcular a nota dada pelo usuário tem como ideia principal extrair essa informação com base no tempo que o mesmo passou assistindo a mídia. Para isso será calculada a razão entre o tempo que o usuário passou assistindo a mídia e a duração total da mesma. Tendo em mãos essa razão, cada faixa de possíveis valores é mapeada para um número inteiro representando a nota final dada pelo usuário. Este mapeamento pode ser observado na Tabela 4.2. Ambas informações necessárias para o cálculo desta razão estão disponíveis no conjunto de dados original, sendo elas os campos “play” e “duration”. O campo “play” representa o tempo de mídia assistida pelo usuário naquela sessão, enquanto o campo “duration” representa a duração total da mídia, ambos representados em milisegundos. Chamamos este método de *Implicit Rating*.

Tabela 4.2: Mapeamento de razão para nota

Razão Calculada	**Nota**
Acima de 0.9	4
De 0.8 a 0.9	3
De 0.6 a 0.8	2
De 0.2 a 0.6	1
Até 0.2	0

Fonte: Elaborada pelo autor.

4.5 Métricas de avaliação e análise dos resultados

Para comparar os *frameworks* analisados é necessário extrair métricas de avaliação de cada um deles. Além de servirem para comparar os resultados dos *frameworks* entre si, também é possível comparar a performance deles quando executados sobre diferentes conjuntos de dados. Com objetivo de definir uma métrica principal de comparação, observou-se as métricas avaliadas nos cinco trabalhos analisados em busca de uma métrica em comum. A Tabela 4.3 mostra as métricas utilizadas em cada um dos trabalhos. É possível observar que *Recall* é a métrica mais presente nos trabalhos, estando ausente apenas no RippleNet.

Tabela 4.3: Métricas utilizadas em cada trabalho

	Accuracy	**AUC**	**NDGC**	**F1**	**Precision**	**Hit**	**Recall**
RippleNet	X	X					
KGCN		X		X			X
DSKE			X				X
KGAT			X				X
ECFKG			X		X	X	X

Fonte: Elaborada pelo autor.

No contexto de sistemas de recomendação, a tarefa principal que se procura resolver é a de recomendar uma lista dos N itens mais relevantes para o usuário. É extremamente incomum a necessidade de recomendar uma lista com todos os itens relevantes ao usuário, já que isso pode ocasionar em um excesso de opções e sobrecarregar o usuário com informações. Por esse motivo, é comum que a avaliação da qualidade das recomendações seja feita sobre os top-K itens recomendados, e não sobre o conjunto completo. Na literatura dos trabalhos analisados, é utilizado K=20, portanto, neste trabalho utilizamos *Recall*@20 como principal métrica de comparação entre os *frameworks*.

A análise sobre os resultados dos experimentos será realizada em duas etapas: (1) uma análise individual de cada *framework*, comparando os resultados obtidos da execução com o conjuntos de dados Globoplay com os resultados de execuções com outros conjuntos de dados e (2) uma análise comparativa entre todos os *frameworks*, com foco em comparar os resultados utilizando a métrica de *Recall*@20 definida anteriormente. Também é importante ressaltar que a análise individual utilizará dados e métricas fornecidas pelos autores dos trabalhos em que os *frameworks* foram propostos para comparar com os resultados aqui obtidos.

5 EXPERIMENTOS E RESULTADOS

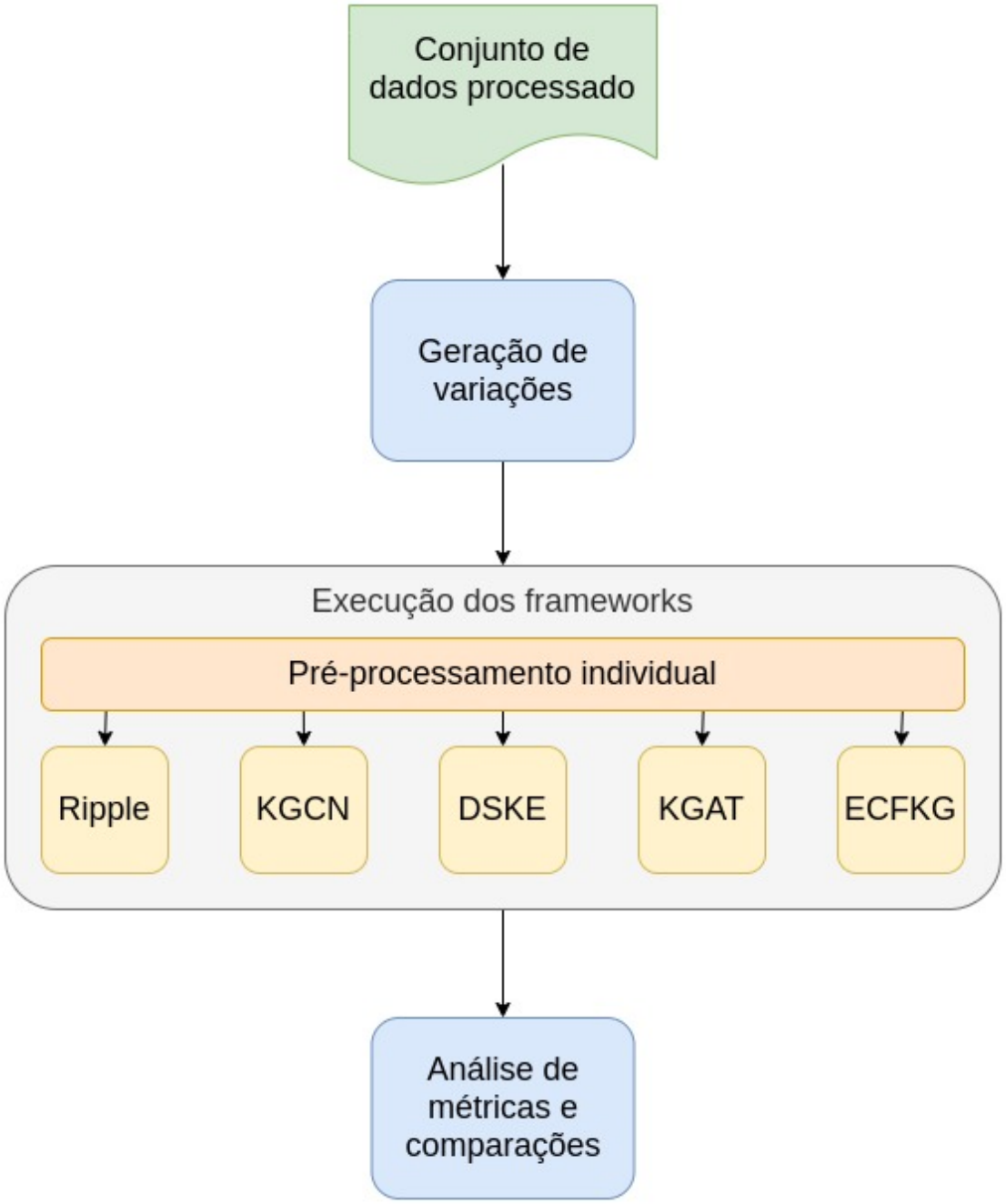
Este capítulo tem como objetivo apresentar as configurações de todos os experimentos realizados, assim como os resultados obtidos em cada um deles. Além disso, também é realizada uma análise individual das métricas extraídas e ao final é feita uma análise comparativa dos resultados obtidos.

5.1 Visão geral

Para este trabalho foram propostos 5 experimentos para serem executados sobre 4 diferentes variações do conjunto de dados Globoplay. Cada experimento extraiu informações sobre um *framework* de recomendações diferente, com objetivo de analisar seus resultados quando aplicado ao conjunto de dados Globoplay e comparar com as métricas *baseline* fornecidas pelos autores dos trabalhos. A Figura 5.1 ilustra o fluxo geral dos experimentos realizados.

A fim de analisar qual o melhor algoritmo de cálculo de notas e se é vantajoso utilizar o algoritmo de agrupamento de usuário, foram geradas diversas variações do conjunto de dados Globoplay para alimentar os *frameworks*. Os dois primeiros experimentos (RippleNet e KGCN), além de servirem para analisar a performance dos sistemas de recomendação, também permitiram definir qual a melhor variação gerada do Globoplay. Os experimentos 3, 4 e 5 (DSKE, KGAT e ECFKG) foram executados utilizando apenas a variação que obteve melhores resultados nos experimentos 1 e 2.

Figura 5.1: Fluxograma dos experimentos realizados



Fonte: Elaborada pelo autor.

5.2 Experimentos

Cada um dos *frameworks* analisados possui conjuntos de dados que foram utilizados pelos autores para avaliá-los, e, portanto, já possuem métricas extraídas para comparação com o conjunto de dados Globoplay.

Além destes conjuntos de dados, utilizamos quatro variações dos dados Globoplay, alterando o método de cálculo de nota (descritos na Seção 4.4) e a utilização do algoritmo de agrupamento por usuário. As variações utilizadas são listadas na Tabela 5.1 e foram pensadas para realizar duas comparações entre si: (1) comparação entre as variações com diferentes algoritmos de cálculo de nota (variações #1, #2 e #3) e (2) comparação entre as variações com e sem utilização de agrupamento por usuário (variações #3 e #4).

Tabela 5.1: Variações do conjunto de dados Globoplay utilizadas

	Cálculo de Nota	**Agrupamento**
Globo #1	RandInt	Não
Globo #2	Uniform	Não
Globo #3	Implicit	Não
Globo #4	Implicit	Sim

Fonte: Elaborada pelo autor.

As subseções a seguir descrevem em detalhes cada um dos experimentos realizados, como conjuntos de dados utilizados, hiperparâmetros configurados, métricas extraídas, suposições avaliadas e resultados obtidos.

5.2.1 RippleNet

O primeiro experimento foi realizado com o *framework* RippleNet (WANG et al., 2018b), utilizando todas as variações do conjunto de dados Globoplay detalhadas anteriormente. Neste experimento, avaliamos a performance do *framework* quando executado com nossos conjuntos de dados e comparamos com as métricas extraídas pelo autor. Além disso, também avaliamos qual o melhor algoritmo de cálculo de nota e se o agrupamento por usuário gerou melhora significante nos resultados obtidos. Para executar este *framework*, foi necessário modelar o conjunto de dados em 3 diferentes arquivos:

- **kg_rehashed**: Neste arquivo, são definidas as relações entre as entidades no grafo de conhecimento, representadas por uma tripla de identificadores de entidade (*headId*, *relationId*, *tailId*). Os identificadores desta tripla são separados por um espaço em branco.
- **item_index2entity_id_rehashed**: Neste arquivo, cada linha contém um mapeamento entre o identificador original do item que é recomendado (no caso do Globoplay, o campo media_ID representa este identificador) e um novo identificador interno que é utilizado pelo *framework*, no formato (*originalId*, *newId*) separados por um espaço em branco. Estes mapeamentos para identificadores internos tem como objetivo anonimizar os dados recebidos e também evitar potenciais problemas com identificadores não normalizados.
- **ratings**: Este arquivo contém as avaliações que os usuários deram aos itens consumidos, sejas estas implícitas ou explícitas. Cada linha está no formato (*userId*, *itemId*, *rating*), separados por um espaço em branco. O nome deste arquivo e o separador utilizado varia de acordo com o conjunto de dados, a descrição acima foi utilizada para o Globoplay.

Além das das variações do conjunto de dados **Globoplay**, também temos os conjuntos de dados **MovieLens1M** e **Book-Crossing** para comparação de resultados. As métricas da execução destes conjuntos de dados foram extraídas pelo autor do trabalho RippleNet (WANG et al.). A Tabela 5.2 mostra métricas extraídas destes conjuntos de dados em comparação com os dados Globoplay.

Tabela 5.2: Comparação entre os conjuntos de dados do experimento 1.

	# usuários	**# itens**	**# relações**	**# interações**	**Densidade**
Movies	6,036	2,445	12	753,772	124.88
Book	17,860	14,962	25	139,746	7.82
Globo #1	1,141,995	32,117	8	2,195,785	1.92
Globo #2	1,141,995	32,117	8	2,195,785	1.92
Globo #3	1,141,995	32,117	8	2,195,785	1.92
Globo #4	36,397	30,608	8	2,000,000	54.95

Fonte: Elaborada pelo autor.

Para executar este *framework*, primeiro é necessário definir os valores dos seus hiperparâmetros. A Tabela 5.3 ilustra os hiperparâmetros utilizados para execução deste experimento com o RippleNet, na qual d representa a dimensão dos *embeddings* dos itens, H representa a quantidade de *hops* utilizada, S representa o tamanho do *ripple set* em cada *hop*, λ_1 representa o peso do termo de regularização, λ_2 representa o peso dos *embeddings* e η representa a taxa de aprendizado do modelo.

Tabela 5.3: Hiperparâmetros utilizados nas execuções do RippleNet.

Movies	$d = 16, H = 2, S = 32, \lambda_1 = 10^{-7}, \lambda_2 = 0.01, \eta = 0.02,$
Book	$d = 4, H = 3, S = 32, \lambda_1 = 10^{-5}, \lambda_2 = 0.01, \eta = 0.001,$
Globo	$d = 16, H = 2, S = 32, \lambda_1 = 10^{-7}, \lambda_2 = 0.01, \eta = 0.02,$

Fonte: Elaborada pelo autor.

Os resultados das execuções do RippleNet com os conjuntos de dados Globoplay, em conjunto com as métricas fornecidas pelo autor, podem ser observados na Tabela 5.4. É possível observar que dentre as três primeiras variações do conjunto de dados Globoplay, a variação #3 demonstrou os melhores resultados em ambas as métricas, indicando superioridade do algoritmo de cálculo de notas baseado em tempo assistido. Além disso, a

variação #4 mostrou resultados significativamente superiores à variação #3, indicativo de que o algoritmo de agrupamento por usuário também possibilita obter melhores resultados devido ao aumento da densidade de interações por usuário mostrada na Tabela 5.2.

Por fim, quando comparamos os resultados da variação Globo #4 com as métricas dos conjuntos de dados fornecidos pelo autor, é possível notar uma similaridade com os resultados do conjunto de dados MovieLens. Por conta da alta densidade de interações por usuário, ambos obtiveram ótimos resultados, e isso se torna ainda mais evidente ao comparar com os resultados do conjuntos de dados Book e das outras 3 variações do Globo, visto que estes possuem baixa densidade de interações e apresentaram resultados relativamente piores.

Tabela 5.4: Resultados do experimento 1.

	Acurácia	**AUC**
Movies	0.844	0.921
Book	0.662	0.729
Globo #1	0.749	0.804
Globo #2	0.765	0.830
Globo #3	0.780	0.846
Globo #4	0.850	0.918

Fonte: Elaborada pelo autor.

5.2.2 KGCN

O segundo experimento foi realizado com o *framework* KGCN (WANG et al., 2019a). Neste experimento, assim como no primeiro, foram utilizadas todas as variações do conjunto de dados Globoplay. Caso os resultados aqui obtidos mostrem o mesmo padrão de superioridade da variação #4 observada no primeiro experimento, começaremos a utilizar apenas este para os próximos experimentos. Além disso, também é feita uma análise

comparativa dos resultados aqui obtidos com as métricas extraídas pelo autor do trabalho. Para a modelagem dos dados de entrada deste *framework*, é necessário gerar os mesmos arquivos descritos na Subseção 5.2.1, já que ambos utilizam o mesmo padrão.

Além das variações do conjunto de dados **Globoplay**, também temos os conjuntos de dados **MovieLens20M** e **LastFM** para comparação de resultados. As métricas da execução destes conjuntos de dados foram extraídas pelo autor do trabalho KGCN (WANG et al.). A Tabela 5.5 mostra métricas extraídas destes conjuntos de dados em comparação com os dados Globoplay.

Tabela 5.5: Comparação entre os conjuntos de dados do experimento 2.

	# usuários	**# itens**	**# relações**	**# interações**	**Densidade**
Movies	138,159	16,954	32	13,501,622	97.73
LastFM	1,872	3,846	60	42,346	22.62
Globo #1	1,141,995	32,117	8	2,195,785	1.92
Globo #2	1,141,995	32,117	8	2,195,785	1.92
Globo #3	1,141,995	32,117	8	2,195,785	1.92
Globo #4	36,397	30,608	8	2,000,000	54.95

Fonte: Elaborada pelo autor.

Para executar este *framework*, primeiro é necessários definir os valores dos seus hiperparâmetros. A Tabela 5.6 ilustra os hiperparâmetros utilizados para execução deste experimento com o KGCN, na qual d representa a dimensão dos *embeddings* dos itens, H representa a quantidade de *hops* utilizada, K representa o tamanho da amostra de vizinhos, λ representa o peso do termo de regularização e η representa a taxa de aprendizado do modelo.

Tabela 5.6: Hiperparâmetros utilizados nas execuções do KGCN.

Movies	$d = 32, H = 2, K = 4, \lambda = 10^{-7}, \eta = 0.02, batchsize = 65,536$
LastFM	$d = 16, H = 1, K = 8, \lambda = 10^{-4}, \eta = 0.0005, batchsize = 128$
Globo	$d = 32, H = 3, K = 8, \lambda = 10^{-7}, \eta = 0.0002, batchsize = 2,048$

Fonte: Elaborada pelo autor.

Os resultados das execuções do KGCN com os conjuntos de dados Globoplay, em conjunto com as métricas fornecidas pelo autor, podem ser observados na Tabela 5.7. É possível observar que o padrão de superioridade da variação #4 do conjunto de dados Globoplay se manteve neste experimento. As métricas de F1 e AUC indicam um desempenho significativamente superior desta variação quando comparada às outras 3, apesar da métrica *recall@20* não demonstrar tanta diferença. Assim como no experimento 1, as métricas obtidas sugerem que quanto maior a densidade de interações por usuário, melhores são os resultados obtidos e consequentemente melhores são as recomendações geradas.

Outro ponto interessante a se observar são as métricas fornecidas pelo autor para o conjunto de dados Movies. Diferente do RippleNet, no KGCN o autor utilizou a versão deste conjunto de dados com 20 milhões de interações entre usuários e filmes, disponibilizando 20 vezes mais informações para o treinamento do *framework* quando comparado à sua versão com apenas 1 milhão de interações. Essa mudança ocasionou em um grande aumento nas métricas de acurácia e AUC, estas que também estão disponíveis no RippleNet. Isso fornece grandes indícios de que o conjunto de dados Globoplay também poderia obter resultados aprimorados caso os modelos sejam treinados com maiores quantidades de dados.

Tabela 5.7: Resultados do experimento 2.

	F1	**AUC**	**Recall@20**
Movies	0.932	0.978	0.157
LastFM	0.688	0.794	0.116
Globo #1	0.623	0.635	0.049
Globo #2	0.644	0.651	0.050
Globo #3	0.659	0.672	0.049
Globo #4	0.831	0.898	0.083

Fonte: Elaborada pelo autor.

5.2.3 DSKE

O terceiro experimento foi realizado com o *framework* DSKE (ZHANG et al., 2019). Neste experimento, diferente dos anteriores, utilizamos apenas a variação #4 do conjunto de dados Globoplay, já que esta variação foi a que demonstrou melhores resultados nos experimentos 1 e 2. Também comparamos os resultados obtidos com as métricas *baseline* fornecidas pelo autor do *framework* relativas aos conjuntos de dados Yelp e Douban. A Tabela 5.8 apresenta um comparativo dos conjuntos de dados deste trabalho.

Tabela 5.8: Comparação entre os conjuntos de dados do experimento 3.

	# usuários	**# itens**	**# relações**	**# interações**	**Densidade**
Yelp	16,239	14,284	4	198,397	12.22
Douban	13,367	12,677	7	1,068,278	79.92
Globo	36,397	30,608	4	2,000,000	54.95

Fonte: Elaborada pelo autor.

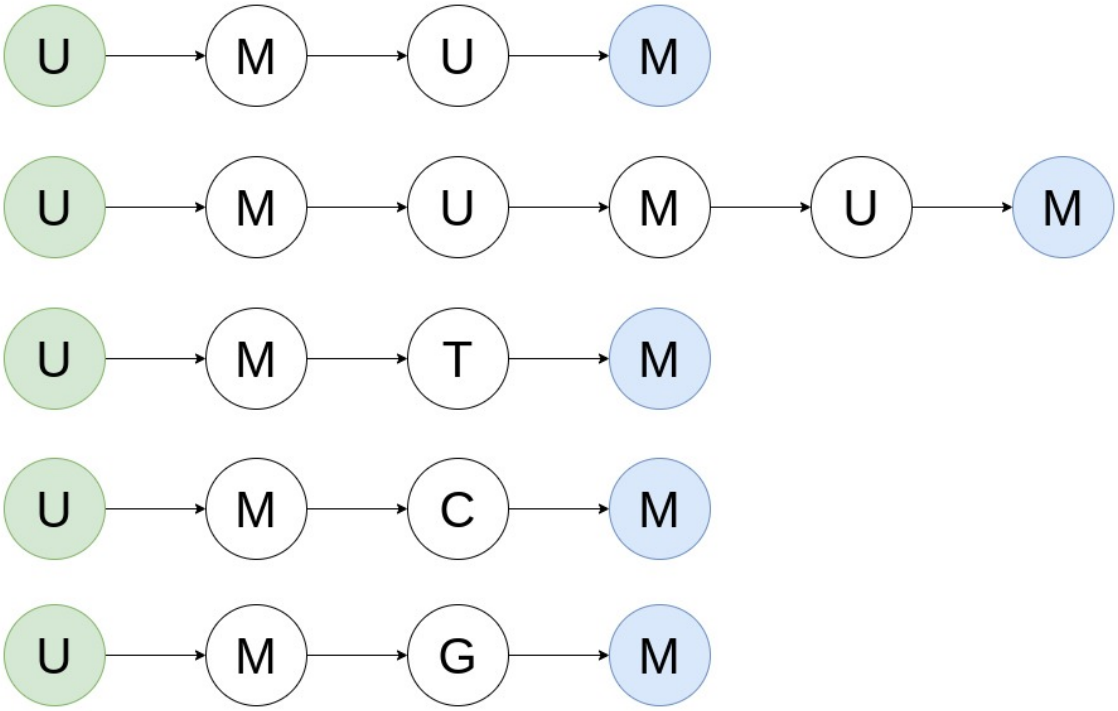
O repositório com o código fonte do DSKE contém um arquivo de código python e um arquivo de dados para cada conjunto de dados. Ao tentar

executar este *framework* com o conjunto de dados Globoplay, percebemos que os arquivos de entrada estavam formatados de uma maneira que deixaria a adaptação muito complexa. Por esse motivo, escolhemos o código fonte mais similar ao que precisávamos e fizemos adaptações para suprir nossas necessidades. O código fonte escolhido para a adaptação foi o do conjunto de dados Yelp, por aparentar ser o mais simples.

Neste código fonte, o arquivo de entrada é lido e cada uma das relações modeladas no grafo de conhecimento é mapeada para um objeto do tipo *dataframe*. Como o formato do arquivo de entrada era desconhecido, este trecho de código foi reescrito para ler arquivos em um formato conhecido, no qual cada linha contivesse dois identificadores de entidade e uma nota, todos separados por um espaço em branco. Cada relação é representada por um arquivo, totalizando quatro diferentes arquivos (user_media, media_title, media_genre e media_category) e apenas o arquivo que relaciona usuários e mídias tinha os valores da coluna de notas lidos.

Por fim, este *framework* também exige que os meta-caminhos sejam modelados manualmente no código fonte para cada conjunto de dados, e estes são essenciais para que o modelo consiga realizar boas recomendações ao usuário. Os meta-caminhos gerados devem seguir duas regras principais: (1) deve iniciar em um usuário; e (2) deve acabar em um item. A Figura 5.2 ilustra os meta-caminhos utilizados para o treinamento do DSKE.

Figura 5.2: Meta-caminhos modelados para o DSKE. U = Usuário, M = Mídia, T = Título, C = Categoria e G = Gênero.



Fonte: Elaborada pelo autor.

Os resultados das execuções do DSKE com o conjunto de dados Globoplay, em conjunto com as métricas fornecidas pelo autor, podem ser observados na Tabela 5.9. É possível observar que o conjunto de dados Yelp possui os piores resultados em ambas as métricas comparados aos outros conjuntos de dados, devido à baixa quantidade de interações em conjunto com a baixa densidade de informações por usuário.

O conjunto de dados Globo obteve os melhores resultados dentre os 3 conjuntos de dados avaliados em ambas as métricas. Em comparação com o Douban, o conjunto de dados Globo possui quase o dobro de interações, indicando que o número de interações tem forte influência sobre os resultados obtidos, mesma conclusão obtida no experimento 2.

Tabela 5.9: Resultados do experimento 3.

	NDGC@100	**Recall@20**
Yelp	0.110	0.101
Douban	0.259	0.236
Globo	0.461	0.474

Fonte: Elaborada pelo autor.

5.2.4 KGAT

O quarto experimento foi realizado com o *framework* KGAT (WANG et al., 2019b), e tem como principal objetivo comparar seu desempenho quando executado com o conjunto de dados Globoplay com relação às métricas *baseline* fornecidas pelo autor do trabalho. Neste experimento também utilizamos apenas a variação #4 do conjunto de dados Globoplay. Para executar este *framework* é necessário modelar o conjunto de dados em 7 diferentes arquivos:

- **entity_list**: Este aquivo possui 2 colunas: org_id e remap_id. Cada linha representa um mapeamento entre o identificador original da entidade e o identificador interno que o *framework* utiliza. Os identificadores gerados devem ser únicos para cada entidade.
- **user_list**: Este aquivo possui 2 colunas: org_id e remap_id. Cada linha representa um mapeamento entre o identificador original do usuário e o identificador interno que o *framework* usará. No caso do Globoplay, a primeira coluna contém o campo user_ID do conjunto de dados original e a segunda coluna contém um id gerado para representar este usuário internamente.
- **item_list**: Este arquivo possui 3 colunas: org_id, remap_id e freebase_id. Assim como no arquivo user_list, este arquivo mapeia um identificador do conjunto de dados original para um identificador interno. O mapeamento realizado neste arquivo deve ser o mesmo do arquivo entity_list. Este arquivo também conta com uma coluna

adicional para representar o identificador do item no Freebase. Apesar disso, para o conjunto de dados Globoplay, este campo é gerado aleatoriamente, não havendo a necessidade de utilizar o Freebase.

- **relation_list**: Este aquivo possui 2 colunas: org_id e remap_id. Cada linha representa um mapeamento entre o identificador original da relação e o identificador interno que o *framework* utiliza. No caso do Globoplay, precisamos gerar identificadores para cada uma das 8 relações modeladas no grafo de conhecimento da Figura 4.2 para preencher a primeira coluna, enquanto a segunda coluna contém um id gerado para representar esta relação internamente.
- **kg_final**: Este arquivo contém o grafo de conhecimento utilizado para o treinamento do *framework*. Cada linha representa uma tripla no formato (*head*, *relation*, *tail*), indicando uma relação entre duas entidades no KG.
- **train**: Este aquivo contém as interações positivas entre usuários e itens que serão utilizadas para treinar o *framework*. Cada linha contém um identificador de usuário e uma lista de itens interagidos por ele.
- **test**: Este aquivo contém as interações positivas entre usuários e itens que serão utilizadas para validar o *framework*. Cada linha contém um identificador de usuário e uma lista de itens interagidos por ele.

Além do conjunto de dados Globoplay, também temos as métricas *baseline* extraídas dos conjuntos de dados AmazonBook, LastFM e Yelp para comparação de resultados. As métricas de execução destes conjuntos de dados foram extraídas pelo autor do trabalho KGAT (WANG et al.). A Tabela 5.10 mostra um comparativo entre as métricas extraídas destes conjuntos de dados e do Globoplay.

Tabela 5.10: Comparação entre os conjuntos de dados do experimento 4.

	# usuários	**# itens**	**# relações**	**# interações**	**Densidade**
AmazonBook	70,679	24,915	39	847,733	11.99
LastFm	23,566	48,123	9	3,034,796	128.78
Yelp	45,919	45,538	42	1,185,068	25.81
Globo	36,397	30,608	8	2,000,000	54.95

Fonte: Elaborada pelo autor.

Para executar este framework é necessário antes realizar a configuração dos hiperparâmetros. Os valores configurados foram os seguintes: $pretrain = -1$, indica que o modelo foi treinado utilizando os *embeddings* aprendidos, $embed_size = 64$, indica a dimensão dos embbedings, $epoch = 1000$, indica a quantidade de épocas, $batch_size = 1024$, indica o tamanho do batch, $alg_type = bi$, especifica o tipo da camada convolucional, $regs = [1e-5, 1e-5]$, representa o termo de regularização dos embeddings de usuario e item, $layer_size = [64, 32, 16]$, indica os tamanhos das saídas das camadas, $lr = 0.0001$, indica a taxa de aprendizado, $use_att = True$, indica a utilização do mecanismo de atenção, $use_kge = True$, indica a utilização do algoritmo de KGE.

Os resultados das execuções do KGAT com o conjunto de dados Globoplay, em conjunto com as métricas fornecidas pelo autor, podem ser observados na Tabela 5.11. É possível notar que, apesar do LastFM ter uma maior quantidade de interações e densidade quando comparado ao Globoplay, os resultados das métricas de NDGC@20 e *Recall*@20 são inferiores. Uma possível explicação para este fato é a natureza do conjunto de dados. Em todos os experimentos realizados até o momento neste trabalho os melhores resultados foram obtidos pelo Globoplay ou pelo MovieLens, ambos conjuntos de dados referentes à mídias audiovisuais.

Tabela 5.11: Resultados do experimento 4.

	NDGC@20	**Recall@20**
AmazonBook	0.101	0.149
LastFM	0.132	0.087
Yelp	0.087	0.071
Globo	0.253	0.199

Fonte: Elaborada pelo autor.

5.2.5 ECFKG

O quinto e último experimento foi realizado com o *framework* ECFKG (AI et al., 2018), e tem como principal objetivo comparar seu desempenho quando executado com o conjunto de dados Globoplay com relação às métricas *baseline* fornecidas pelo autor do trabalho. Neste experimento também utilizamos apenas a variação #4 do conjunto de dados Globoplay. A implementação deste modelo foi fornecida pelo autor do KGAT, portanto as métricas, conjuntos de dados, arquivos de entrada e hiperparâmetros utilizados são idênticos aos do experimento 4, descritos na Subseção 5.2.4.

Os resultados das execuções do ECFKG com o conjunto de dados Globoplay, em conjunto com as métricas fornecidas pelo autor, podem ser observados na Tabela 5.12. Podemos observar que este foi o único trabalho em que o conjunto de dados Globoplay não se destacou com bons resultados. Diferentes dos outros 4 trabalhos, o ECFKG é o único que utiliza somente métodos baseados em *embeddings* para gerar recomendações, utilizando *collaborative filtering*. Este fato pode acabar prejudicando a captura das preferências do usuário por fazer pouco uso do grafo de conhecimento fornecido.

Tabela 5.12: Resultados do experimento 5.

	NDGC@20	**Recall@20**
AmazonBook	0.077	0.114
LastFM	0.114	0.072
Yelp	0.064	0.052
Globo	0.107	0.067

Fonte: Elaborada pelo autor.

5.3 Análise geral dos resultados

No geral, as execuções dos *frameworks* com o conjunto de dados Globoplay apresentaram bons resultados. A Tabela 5.13 mostra um compilado dos resultados da métrica *Recall*@20 obtidos para todos os conjuntos de dados. O *framework* RippleNet não possui métricas de *Recall*@20 disponíveis, portanto não faz parte da tabela comparativa. É possível perceber que o DSKE apresentou resultados significativamente superiores ao outros *frameworks*, atingindo um valor de *Recall*@20 238% superior ao KGAT, este que é o segundo com maior valor apresentado.

Realizando uma análise sobre o código fonte do DSKE disponibilizado pelo autor, foram constatados 3 fatores que podem implicar no aumento dessa métrica:

1. O cálculo do *Recall*@20 feito no DSKE é diferente do normalmente utilizado, de modo que o denominador da fórmula seja o mínimo entre quantidade de itens relevantes para o usuário e 20, ocasionando um resultado maior que o esperado quando o usuário possui muitos itens relevantes.
2. O autor filtra os usuários pela quantidade de interações que este possui. Caso essa quantidade seja menor que 5, o usuário é descartada, ocasionando em um processo de treinamento e validação somente com usuários com maior quantidade de informações.

3. As interações de um dado usuário são sempre divididas entre os conjuntos de treino, teste e validação, de modo que ao realizar a validação, todos os usuário tiveram interações processadas no treinamento.

O ECFKG foi o *framework* que apresentou os piores resultados, seguindo um padrão que já apresentava nos experimentos realizados pelo autor do mesmo. Estes resultados provavelmente se devem ao fato do ECFKG ser o único trabalho analisado que utiliza apenas filtragem colaborativa para gerar as recomendações, sem fazer o uso de nenhum método baseado em caminhos. Apesar da performance, o ECFKG ainda possui as vantagens de possuir um módulo que gera explicações para as recomendações, além da simplicidade do modelo gerado.

De forma geral, os trabalhos aqui analisados demonstraram ótimos resultados e muito potencial para melhoras futuras. Um ponto importante a se observar é a notável diferença nos resultados obtidos ao treinar os modelos com o MovieLens quando comparado com outros conjuntos de dados. Em conjunto com a performance acima da média apresentada pelo Globoplay, é possível notar uma tendência que os sistemas de recomendações baseados em grafos tem de gerar boas recomendações de conteúdos de mídia audiovisual.

Além disso, ao comparar a métrica AUC apresentada pelos *frameworks* RippleNet e KGCN para o conjunto de dados MovieLens, podemos observar um melhora significante dos resultados obtidos pelo KGCN. Esta melhora está diretamente relacionada com o aumento na quantidade de informações utilizadas deste conjunto de dados. Enquanto o RippleNet utilizou o MovieLens1M, que contém 1 milhão de interações, o KGCN utilizou o MovieLens20M, que contém 20 milhões de interações. Então, em conjunto com a similaridade de natureza entre o MovieLens e o Globoplay, podemos assumir que um aumento na quantidade de informações utilizadas do Globoplay implicaria em uma melhora significativa das recomendações geradas.

Tabela 5.13: Recall@20 para os diferentes experimentos.

	KGCN	**DSKE**	**KGAT**	**ECFKG**
MovieLens	0.157	-	-	-
Book-Crossing	-	-	-	-
Douban	-	0.236	-	-
AmazonBook	-	-	0.149	0.114
LastFM	0.116	-	0.087	0.072
Yelp	-	0.101	0.071	0.052
Globo #4	0.083	0.474	0.199	0.067

Fonte: Elaborada pelo autor.

6 CONCLUSÃO

Neste trabalho foi feita uma análise de aplicabilidade de diferentes sistemas de recomendação baseados em grafos propostos na literatura sobre um conjunto de dados da plataforma de *streaming* digital Globoplay. Inicialmente foi feita uma busca por trabalhos na literatura com códigos fonte disponíveis e replicáveis. Em seguida, o conjunto de dados Globoplay foi modelado em um grafo de conhecimento e utilizado para treinar os *frameworks* selecionados. Por fim, foi feita uma análise dos resultados obtidos pela execução dos *frameworks*.

Os resultados obtidos demonstraram que os sistemas de recomendação baseados em grafos tem grande potencial para gerar boas recomendações quando aplicados ao conjuntos de dados do Globoplay. Além disso, os sistemas de recomendação baseados em grafos de conhecimento que utilizam métodos unificados para gerar recomendações se mostraram superiores aos métodos baseados em *embeddings*, pelo fato de explorarem melhor a conectividade do grafo de conhecimento e modelar mais profundamente as preferências do usuário. Por fim, também foi observado que estes *frameworks* tendem a obter resultados ainda melhores quando aplicados à conjuntos de dados em que os itens são conteúdos de mídia audiovisual.

Para trabalhos futuros, seria interessante incrementar a modelagem do grafo de conhecimento do conjunto de dados Globoplay, explorando mais atributos para gerar uma representação mais profunda e rica das informações disponíveis. Além disso, por conta de limitações de performance não foi possível utilizar uma quantidade maior de dados para treinar os modelos. As métricas *baseline* fornecidas pelos autores dos *frameworks* para o conjunto de dados MovieLens indicam que a utilização de uma maior quantidade de dados influencia positivamente nos resultados obtidos. Portanto, seria interessante explorar a possibilidade de treinar os modelos com uma maior quantidade de informações, realizando também uma análise comparativa de tempos de execução e correlacionando com as melhoras obtidas.

REFERÊNCIAS

AI, Q.; AZIZI, V.; CHEN, X.; ZHANG, Y. Learning heterogeneous knowledge base embeddings for explainable recommendation. **Algorithms**, v. 11, n. 9, 2018. ISSN 1999-4893. Disponível na Internet: <https://www.mdpi.com/1999-4893/11/9/137>.

CHEW, L.-J.; HAW, S.-C.; SUBRAMANIAM, S. Recommender system for retail domain: An insight on techniques and evaluations. Em: . New York, NY, USA: Association for Computing Machinery, 2020. (ICCMS '20), p. 9–13. ISBN 9781450377034. Disponível na Internet: <https://doi.org/10.1145/3408066.3408101>.

EHRLINGER, L.; WÖSS, W. Towards a definition of knowledge graphs. Em: **SEMANTiCS**. [S.l.: s.n.], 2016.

GUO, Q.; ZHUANG, F.; QIN, C.; ZHU, H.; XIE, X.; XIONG, H.; HE, Q. **A Survey on Knowledge Graph-Based Recommender Systems**. arXiv, 2020. Disponível na Internet: <https://arxiv.org/abs/2003.00911>.

KLYNE, G.; CARROLL, J. J. **Resource Description Framework (RDF): Concepts and Abstract Syntax**. 2004. W3C Recommendation. Disponível na Internet: <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>.

MA, W.; ZHANG, M.; CAO, Y.; Woojeong; Jin; WANG, C.; LIU, Y.; MA, S.; REN, X. **Jointly Learning Explainable Rules for Recommendation with Knowledge Graph**. arXiv, 2019. Disponível na Internet: <https://arxiv.org/abs/1903.03714>.

MICHELUCCI, U. **An Introduction to Autoencoders**. arXiv, 2022. Disponível na Internet: <https://arxiv.org/abs/2201.03898>.

TANG, X.; WANG, T.; YANG, H.; SONG, H. Akupm: Attention-enhanced knowledge-aware user preference model for recommendation. Em: **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining**. New York, NY, USA: Association for Computing Machinery, 2019. (KDD '19), p. 1891–1899. ISBN 9781450362016. Disponível na Internet: <https://doi.org/10.1145/3292500.3330705>.

WANG, H.; ZHANG, F.; HOU, M.; XIE, X.; GUO, M.; LIU, Q. Shine: Signed heterogeneous information network embedding for sentiment link prediction. Em: **Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2018. (WSDM '18), p. 592–600. ISBN 9781450355810. Disponível na Internet: <https://doi.org/10.1145/3159652.3159666>.

WANG, H.; ZHANG, F.; WANG, J.; ZHAO, M.; LI, W.; XIE, X.; GUO, M. Ripple network: Propagating user preferences on the knowledge graph for recommender systems. **CoRR**, abs/1803.03467, 2018. Disponível na Internet: <http://arxiv.org/abs/1803.03467>.

WANG, H.; ZHAO, M.; XIE, X.; LI, W.; GUO, M. Knowledge graph convolutional networks for recommender systems. Em: **The World Wide Web Conference**. New York, NY, USA: Association for Computing Machinery, 2019. (WWW '19), p. 3307–3313. ISBN 9781450366748. Disponível na Internet: <https://doi.org/10.1145/3308558.3313417>.

WANG, X.; HE, X.; CAO, Y.; LIU, M.; CHUA, T.-S. Kgat: Knowledge graph attention network for recommendation. Em: **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining**. New York, NY, USA: Association for Computing Machinery, 2019. (KDD '19), p. 950–958. ISBN 9781450362016. Disponível na Internet: <https://doi.org/10.1145/3292500.3330989>.

YU, X.; REN, X.; SUN, Y.; STURT, B.; KHANDELWAL, U.; GU, Q.; NORICK, B.; HAN, J. Recommendation in heterogeneous information networks with implicit user feedback. Em: **Proceedings of the 7th ACM Conference on Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2013. (RecSys '13), p. 347–350. ISBN 9781450324090. Disponível na Internet: <https://doi.org/10.1145/2507157.2507230>.

ZHANG, F.; YUAN, N. J.; LIAN, D.; XIE, X.; MA, W.-Y. Collaborative knowledge base embedding for recommender systems. Em: **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. ISBN 9781450342322. Disponível na Internet: <https://doi.org/10.1145/2939672.2939673>.

ZHANG, Y.; XU, X.; ZHOU, H.; ZHANG, Y. **Distilling Structured Knowledge into Embeddings for Explainable and Accurate Recommendation**. arXiv, 2019. Disponível na Internet: <https://arxiv.org/abs/1912.08422>.