UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

CLAUDIA FRANCESCA SUAREZ MARISCAL

Uma Reflexão em Ciência de Dados sobre o Comportamento de Compra e Sistemas de Recomendação Explicáveis

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Orientador: Prof. Dr. Weverton Cordeiro
Co-orientador: Prof. Dra. Renata Galante

Porto Alegre
2024

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Mariscal, Claudia Francesca Suarez

Uma Reflexão em Ciência de Dados sobre o Comportamento de Compra e Sistemas de Recomendação Explicáveis / Claudia Francesca Suarez Mariscal. – Porto Alegre: PPGC da UFRGS, 2024.

95 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2024. Orientador: Weverton Cordeiro; Coorientador: Renata Galante.

1. Long Short-Term Memory (LSTM). 2. Gated Recurrent Unit (GRU). 3. Sistemas de Recomendações Explicáveis. 4. Grafo de Conhecimento. I. Cordeiro, Weverton. II. Galante, Renata. III. Título.

“A persistência realiza o impossível.”

– PROVÉRBIO CHINÊS

AGRADECIMENTOS

Agradeço à Universidade Federal do Rio Grande do Sul que contribuiu para o meu desenvolvimento na pesquisa durante esses anos. Muito obrigada, professoras Viviane, Mariana, Karin, entre outros, por compartilhar seus conhecimentos comigo.

Agradeço ao meu orientador Weverton Cordeiro e à minha co-orientadora Renata Galante, por terem me incluído como sua aluna de mestrado. Obrigada pela paciência e pela motivação diária na pesquisa que hoje me traz novas oportunidades na minha vida. Também agradeço-lhes por ter aberto as portas do seu país, Brasil, onde encontrei uma cultura incrível e vivi muitas experiências inesquecíveis durante minha estada. Agradeço também, à professora Bruna por ter aceitado em ajudar na preparação final desta tese.

Agradeço à minha família por ser minha motivação nessa etapa do mestrado. Ao meu pai Gabriel e à minha mãe Gladis, por me darem as ferramentas para construir o meu futuro como eu quiser. Aos meus irmãos Marcelo e Silvia, por serem meus cúmplices e me ensinarem que cada um tem um jeito de ser, e aos meus bichinhos, por serem seres de luz nos meus momentos mais difíceis nos últimos anos.

Agradeço às minhas maravilhosas melhores amigas Cynthia e Carmen, pela amizade durante esses quase quinze anos e por sempre estarem comigo nos momentos bons e ruins. Agradeço ao meu namorado Abraham, por ser meu apoio incondicional e não me deixar desistir nesta ultima etapa para alcançar este objetivo.

Agradeço e dedico todo o esforço do meu trabalho de mestrado ao meu querido primo Héctor Rafael, por ser meu apoio incondicional em inúmeras decisões. Embora sua partida tenha sido muito difícil de aceitar, confio nos planos de Deus.

Por fim, agradeço à banca avaliadora por aceitar o convite.

Muito obrigada,

RESUMO

A análise de dados no *E-commerce* por meio de ciência de dados tem sido positiva para fornecer informações valiosas que podem gerar *insights* significativos para melhorar a estratégia de negócio. No entanto, é necessário garantir o correto tratamento dos dados em relação ao tipo de informação (dados pessoais) e a quantidade de dados que são empregados. Esta dissertação apresenta dois propostas para aplicar a ciência de dados na analise do comportamento de compra do usuário, uma delas por meio redes recurrentes e uma outra por meio de sistemas de recomendações baseados em grafo de conhecimento. As duas são focadas em garantir o correto tratamento dos dados respeitando aquelas legislações que garantem a privacidade de informações pessoais, como são a Lei de Proteção Geral de Dados Pessoais (LGPD) e o *General Data Protection Regulation* (GDPR). As duas propostas seguiram os passos da ciência de dados, desde a eleição do *dataset* até a avaliação dos resultados. A partir dos resultados, constatou-se que é importante efetuar as etapas de ciência de dados de forma ordenada e completa para garantir bons resultados. Ademais, o uso da ciência de dados na área de *E-commerce* é promissora, mas, o cientista de dados deve garantir o tratamento responsável dos dados, e sua ética deve prevalecer nas decisões ao realizar diversas análises.

Palavras-chave: Long Short-Term Memory (LSTM). Gated Recurrent Unit (GRU). Sistemas de Recomendações Explicáveis. Grafo de Conhecimento.

An Insight into the Data Science on Customer Behavior Purchase and Explainable Recommendation Systems

ABSTRACT

Data analysis in E-commerce via data science has proven beneficial in providing valuable insights to enhance business strategy. However, ensuring accurate data processing, particularly concerning personal data and data volume, is crucial. This study introduces two approaches for leveraging data science in analyzing user purchasing behavior: one utilizing recurrent networks and the other employing recommendation systems based on knowledge graphs. Both methods prioritize proper data handling and compliance with privacy regulations like the General Data Protection Law (LGPD) and the General Data Protection Regulation (GDPR). Following standard data science procedures, including dataset selection and result evaluation, both approaches underscore the importance of methodical execution for favorable outcomes. While the application of data science in E-commerce shows promise, data scientists must uphold ethical standards to responsibly handle data and make informed decisions during analyses.

Keywords: Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Explainable Recommendation Systems, Knowledge Graph (KG).

LISTA DE ABREVIATURAS E SIGLAS

CF	*Collaborative Filtering*
CFKG	*Collaborative Filtering Knowledge Graph*
CKG	*Collaborative Knowledge Graph*
CLV	*Customer Lifetime Value*
CTR	*Click-through Rate*
DKN	*Deep knowledge-aware network*
DL	*Deep Learning*
DSKE	*Distilling Structured Knowledge into Embeddings*
ECFKG	*Explainable Collaborative Filtering Knowledge Graph*
FFT	*Fast Fourier Transform*
GAFC	*Graph attention-based collaborative filtering*
GCN	*Graph Convolutional Network*
GDRP	*General Data Protection Regulation*
GNN	*Graph Neural Network*
GRU	*Gated Recurrent Unit*
HAGERec	*Hierarchical Attention Graph Convolutional Network Incorporating Knowledge Graph for Explainable Recommendation*
HCI	*Human-Computer Interaction*
HIN	Heterogeneous Information Network
IA	*Inteligência Artificial*
KDE	*Kernel Density Estimation*
KGAT	*Knowledge Graph Attention Network for Recommendation*
KGCN	*Knowledge Graph Convolutional Networks*
KGE	*Knowledge Graph Embedding*
KGIN	*Knowledge Graph-based Intent Network*

KSR	*Knowledge Enhanced sequential recommender*
KTUP	*Knowledge-enhanced Translation-based User Preference model*
LR	*Logistic Regression*
LSTM	*Long Short-Term Memory*
MKR	*Multi-task feature learning for Knowledge graph enhanced Recommendation*
ML	*Machine Learning*
MLP	*Multilayer Perceptron*
NB	*Naive Bayes*
PGM	*Probabilistic Graphical Models*
PGPR	*Policy-Guided Path Reasoning*
POI	*Pontos de Interesse*
PRED	*Periodic Region Detection for Mobility Modeling of Social Media Users*
RNN	*Recurrent Neural Network*
SR	*Sistemas de Recomendação*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O fato de analisar dados em plataformas de *E-commerce* tem sido de muita importância nas últimas décadas, mas, devido a que os dados crescem excessivamente, as empresas devem encontrar as estrategias certas que possam executar essa analise e, desse modo, explorar o comportamento do cliente. A ciência de dados tem o poder transformar os dados em fonte de conhecimento e ajudar na tomada de decisões. Nesse sentido, pode ser determinado o maior item vendido em certas épocas do ano, ou pode ser estabelecida uma segmentação dos clientes, assim também podem-se criar recomendações personalizadas, adaptadas às preferências de cada pessoa. No entanto, como cientistas de dados é necessário garantir o correto tratamento dos dados respeitando aquelas legislações que garantem a privacidade de informações pessoais, como são a Lei de Proteção Geral de Dados Pessoais (LGPD) e o *General Data Protection Regulation* (GDPR). Então, o desafio reside em utilizar o poder computacional adequadamente, efetuando cada uma das etapas da ciência de dados, para identificar tendencias ou padrões nos dados, e assim obter informações valiosas sobre o comportamento de seus clientes.

Para analisar o comportamento do cliente, é necessário inferir algumas ações, por exemplo, as ações de compra quase sempre podem estar relacionadas entre si ou a compra de um item pode ser repetida de acordo com cenários específicos. Desse modo, podem ser aplicados modelos probabilísticos, modelos de *Machine Learning* (ML) ou *Deep Learning* (DL) de acordo ao objetivo que seja estabelecido na analise preditiva. A fim de que os modelos possam prever ações futuras, muitas vezes deve-se considerar informações privadas da pessoa na analise, nesse sentido, a função de cientista de dados não deve focar apenas nos resultados, mas também em estabelecer os limites das análises empregadas e aprimorar os resultados por meio de dados que oferecem um certo grau de informação relevante para a ciência sem revelar detalhes da privacidade das pessoas (VASUPULA; MUNNANGI; DAGGUBATI, 2022). Por outro lado, os sistemas de recomendações procuram alcançar itens cada vez mais personalizados garantindo a confiabilidade do usuário na forma em que sua informação é processada, isso é adquirido por os Sistemas de Recomendações Explicáveis. Além disso, os SRs enfrentam desafios como a escassez de dados ou problemas de *cold-start*. No entanto, as recomendações explicáveis são muito bem abordadas por meio da construção do grafo de conhecimento, que permite achar aquela "explicação" ao descobrir relações implícitas nos dados (WANG et al., 2021).

Na analise preditiva no *E-commerce*, os modelos probabilísticos (YUAN et al., 2017; WEN et al., 2018) têm a capacidade de extrair relações das observações com a finalidade de calcular as probabilidades de acontecerem certos eventos, mas eles precisam ser atualizados com frequência. No caso dos modelos de ML e DL, eles são treinados e testados com os dados do histórico, o que permite-lhes aprender e identificar automaticamente padrões de comportamento (LI et al., 2019). Porém, o comportamento de compra é variável e precisa de uma análise ainda mais profunda que permita levar em consideração ações passadas. Nesse sentido, as abordagens baseadas em (DL) (HUANG et al., 2019; ZHU et al., 2020; ZHOU et al., 2018; SARKAR; BRUYN, 2021) podem modelar problemas sequenciais e assim capturar as dependências temporais no comportamento do cliente. Enquanto isso, os *frameworks* baseados em gráficos de conhecimento são menos complexos de implementar quando estão focados em *embeddings* (ZHANG et al., 2018; AI et al., 2018; CAO et al., 2019), mas o processo de aprendizagem torna-se complexo à medida que a quantidade de dados aumenta. Da mesma forma, as propostas focadas em encontrar padrões de acordo com as conexões estabelecidas (MA et al., 2019; ZHANG et al., 2020) tornam-se não escaláveis em determinado ponto, pelo fato da enumeração ser muito entendiante deixando de fora informações relevantes. Por outro lado, os modelos focados em propagação (WANG et al., 2018a; WANG et al., 2019b; WANG et al., 2019), melhoram suas eficiências com o uso de técnicas mais complexas (*Graph Neural Network*) para aplicar a amostragem vizinha em cada camada e cobrir o conhecimento mais significativo com menor grau de complexidade.

A abordagem principal nesta dissertação foi aplicar a ciência de dados na área do *E-commerce*, de modo que cada uma das etapas seja efetuada de forma ordena e completa. Dessa maneira, foram estabelecidos dois objetivos descritos a seguir:

1. Analisar o comportamento de compra do usuário empregando arquiteturas baseadas em redes neurais recorrentes (RNN, LSTM e GRU) usando um *dataset* que não inclua informações privadas dos usuários.
2. Avaliar *frameworks* de sistemas de recomendações explicáveis baseados em grafos de conhecimento usando dados reais de uma plataforma *streaming* brasileira.

A motivação foi dar solução a problemas reais que podem acontecer na areá do *E-commerce*, por meio da aplicação da ciência de dados usando modelos de aprendizado supervisado. Além disso, apresenta-se o processo detalhado na etapa da experimentação para garantir os melhores resultados. As contribuições são apresentadas a seguir:

- Deixar a implementação dos modelos baseados em redes neurais recurrentes para que possam ser empregados na análise de outros *datasets*, assim como em outros cenários que tem a ver com a analise preditiva no *E-commerce*.
- Deixar o processo de adaptação do *dataset* extenso de uma plataforma *streaming* em cada *framework* para ser empregado em tempo real.

O trabalho foi organizado da seguinte forma. O Capítulo 2 apresenta os principais conceitos deste trabalho. O Capítulo 3, apresenta e compara os trabalhos relacionados considerados na pesquisa, também inclui a seleção dos *baselines*. Nos Capítulos 4 e 5, são discutidas as metodologias utilizadas, bem como as etapas empregadas para atingir os objetivos propostos. O Capítulo 6, aborda os experimentos e os resultados obtidos pela aplicação das metodologias descritas. Finalmente, o Capítulo 7 apresenta as considerações finais sobre os focos deste trabalho e direções futuras.

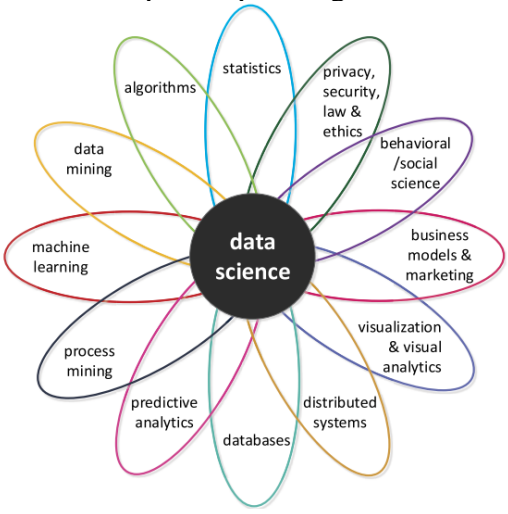
2 FUNDAMENTAÇÃO TEÓRICA

O objetivo deste Capítulo é descrever os conceitos mais importantes que permitem a compreensão desta tese de acordo aos objetivos abordados. Primeiramente, é apresentado o conceito de ciência de dados. Em seguida, são apresentados os focos relacionados à previsão do comportamento de compra, o conceito de *General Data Protection Regulation (GDRP)* e uma classificação dos modelos utilizados nesse tipo de análise. Também são apresentados a definição dos sistemas de recomendações explicáveis, a definição de grafo de conhecimento (ou *Knowledge Graph*) e uma classificação dos modelos empregados em sistemas de recomendação que usam grafo de conhecimento.

2.1 Ciência de dados

A ciência de dados é um campo interdisciplinar que visa transformar dados em valor real, ou seja, o objetivo é extrair conhecimento e tomar decisões a partir dos dados. O valor pode ser fornecido na forma de previsões, decisões automatizadas, modelos aprendidos a partir de dados ou qualquer tipo de visualização de dados que forneça *insights*. A ciência de dados inclui extração de dados, preparação de dados, exploração de dados, transformação, vários tipos de mineração e aprendizagem, apresentação de explicações e previsões e exploração de resultados tendo em conta aspectos éticos, aspectos sociais, aspectos legais e empresariais (AALST; AALST, 2016). A Figura 2.1 mostra que a ciência de dados é uma mistura de diferentes sub-disciplinas, que são descritas a seguir:

Figura 2.1: Sub-disciplinas que integram a ciência de dados



Fonte: Aalst and Aalst (2016)

1. As **estatísticas** podem ser vistas como a origem da ciência de dados. Elas se dividem entre estatísticas descritivas e estatísticas inferenciais.
2. Os **algoritmos** são cruciais em qualquer abordagem, pois sua complexidade aumenta quando o conjunto de dados é maior.
3. A **mineração de dados** refere-se à análise de conjuntos de dados para descobrir padrões e relacionamentos ocultos por meio de modelos de regressão, modelos de classificação, *clusters*, entre outros.
4. O ***machine learning*** **(ML)** surgiu dentro da Inteligência Artificial (IA) e refere-se ao uso de diversos modelos que "aprendem" por meio de dados de entrada.
5. A **mineração de processos** busca a engrenagem entre os dados de eventos (ou seja, comportamento observado) e os modelos de processos (feitos manualmente ou descobertos automaticamente).
6. A **análise preditiva** refere-se ao processo de extrair informações de um conjunto de dados para determinar padrões e prever resultados e tendências futuras.
7. A disciplina de **bancos de dados** é muito importante no gerenciamento de dados, pois tem dois propósitos: estruturar dados para que possam ser gerenciados facilmente e fornecer escalabilidade e desempenho confiável.
8. Os **sistemas distribuídos** fornecem a infraestrutura para conduzir análises. Algumas tarefas de análise podem ser muito complexas para serem executadas em um único computador, dessa forma essas tarefas podem ser divididas em muitas tarefas menores que podem ser executadas simultaneamente.
9. A **visualização de dados** combina técnicas de análise automatizada com visualizações interativas para uma compreensão, raciocínio e tomada de decisão eficazes com base en conjuntos de dados muito grandes e complexos.
10. Os **modelos de negócios** são incluídos porque a ciência de dados busca transformar dados em valor, incluindo o valor comercial.
11. As **ciências comportamentais** são necessárias para interpretar os resultados, pois permitem compreender o comportamento humano e o contexto social em que os humanos e as organizações operam.
12. **Privacidade, segurança, lei e ética** são ingredientes essenciais para proteger os indivíduos e as organizações de práticas "ruins" de ciência de dados.

2.2 Leis de proteção de dados

1. ***General Data Protection Regulation* (GDRP)** - Um dos mecanismos que ajudam a proteger os dados é a anonimização, que é uma forma de tratamento dos dados que impossibilita a identificação da pessoa a quem esses dados pertencem. Por meio desse processo, pretende-se evitar riscos de tratamento massivo de dados, permitindo a proteção de informação sensível sem violar os direitos de proteção dos dados das pessoas e organizações (WIERINGA et al., 2021). A análise de dados trouxe benefícios importantes em diversas áreas, mas ainda são reportados problemas com a privacidade dos dados (EREVELLES; FUKAWA; SWAYNE, 2016). O GDRP integra iniciativas de regulamento de privacidade de dados que buscam criar confiança nos usuários para coletar dados pessoais (VOIGT; BUSSCHE, 2017), pois é a lei de privacidade e segurança mais rígida do mundo que impõe obrigações às organizações, sempre que dados relativos a indivíduos na União Europeia são direcionados ou coletados. Desse modo, o posicionamento que exerce é rigoroso com aquelas empresas que pretendem utilizar dados privados, fortalecendo a privacidade e a segurança dos dados, fazendo com que mais pessoas confiem em seus dados pessoais e evitando as violações que comumente ocorrem.
   No artigo 4, o GDRP[1] define os dados pessoais como "qualquer informação relativa a uma pessoa singular identificada ou identificável", ou seja, o ele exige esforços das empresas para proteger os dados, bem como descreve os direitos dos consumidores sobre os seus dados pessoais. Consequentemente, existem cenários que podem ser beneficiados ao permitir a divulgação de dados pessoais, por exemplo, os consumidores podem obter maiores benefícios ao fazerem parte de uma segmentação de mercado. No entanto, as consequências negativas podem estar relacionadas ao compartilhamento não autorizado com empresas desconhecidas do consumidor. Diante disso, para neutralizar o lado negativo, o GDPR exige que as empresas obtenham o consentimento do consumidor (DINEV; HART, 2006), de modo que as preocupações com a privacidade tenham um efeito direto e limite na capacidade das empresas de coletar, processar e analisar dados pessoais.
2. **Lei Geral de Proteção de Dados (LGPD)** - Afim de para promover a proteção aos dados pessoais de todo cidadão que esteja no Brasil, essa lei é focada na criação de um cenário de segurança jurídica com a padronização de regulamentos e práticas de

[1] https://gdpr-info.eu/

acordo com os parâmetros internacionais existentes (PINHEIRO, 2020). O LGPD garante ao titular dos dados pessoais o direito a: (1) ter acesso aos tipos de dados e a quais de seus dados pessoais são utilizados para alimentar algoritmos responsáveis por processos automatizados, e (2) receber explicações sobre os critérios utilizados para tomar a decisão automatizada, que deve ser analisado caso-acaso.

A lei define o que são dados pessoais e explica que alguns deles estão sujeitos a cuidados ainda mais específicos, como os dados pessoais sensíveis e dados pessoais sobre crianças e adolescentes (MONTEIRO, 2018). Esclarece ainda que todos os dados tratados, tanto no meio físico quanto no digital, estão sujeitos à regulação. Além disso, a LGPD estabelece que não importa se a sede de uma organização ou o centro de dados dela estão localizados no Brasil ou no exterior: se há o processamento de informações sobre pessoas, brasileiras ou não, que estão no território nacional, a LGPD deve ser observada (MULHOLLAND, 2018). A lei autoriza também o compartilhamento de dados pessoais com organismos internacionais e com outros países, desde que observados os requisitos nela estabelecidos.

2.3 Focos na Previsão do Comportamento de Compra

Uma análise focada nos registros das atividades de compra do usuário no *E-commerce* permite uma melhor compreensão dos padrões de comportamento do cliente para determinar os aspectos relevantes que levam à sua satisfação (SAURA, 2021). A seguir, são descritos alguns focos na previsão do comportamento de compra:

- Identificação de padrões e tendências - as técnicas de ML possibilitam cumprir esse objetivo de forma automatizada por meio de modelos capazes de analisar grandes quantidades de dados a fim de identificar padrões de mercado ao longo de períodos de tempo. Por exemplo, no mês de novembro, as vendas de determinados produtos aumentam por algum motivo, então, as equipes de *marketing* podem planejar uma campanha de lançamento ou uma promoção de produtos similares.
- Previsibilidade - as empresas esperam estar à frente de demandas que precisam ser conhecidas antecipadamente. A seguir, são listados alguns objetivos específicos:
    1. Previsão se um determinado usuário fará uma compra em uma determinada categoria de produto em tempo real, o que permite capturar o momento exato em que ele poderia estar interessado em fazer uma compra.

2. Previsão na continuidade das compras, o que ajuda a planejar campanhas de *marketing* personalizadas oferecendo diversos produtos para os clientes.
3. Previsão do *Customer Lifetime Value* (CLV), o que ajuda a identificar e avaliar os clientes de alto valor e reduzir a exposição a perdas. O valor do CLV é uma métrica fundamental para medir e avaliar o desempenho das campanhas.
4. Previsão de rotatividade de clientes, o que permite aumentar a taxa de retenção e trazer um fluxo constante de clientes em qualquer período do ano.
5. Previsão de demanda dos produtos, o que permite atender às necessidades do cliente no futuro, mantendo atualizados a variedade e o estoque dos produtos.

- Proteção contra fraudes - quanto maior quantidade de dados forem coletados (como, por exemplo, o histórico de consumidores e suas compras), mais fácil será detectar anomalias. Portanto, pode-se aplicar modelos de ML para identificar padrões e consequentemente determinar o que é "normal" ou não.
- Sistema de recomendação de produtos - a personalização é fundamental no direcionamento do cliente à compra. Os SRs permitem identificar com precisão os produtos ou serviços que os usuários estão procurando.

2.4 Modelos de Análise Preditiva para *E-commerce*

Existem vários aspectos que podem ser analisados a partir do comportamento de compra e, por isso, diversos modelos são associados ao objetivo a ser alcançado. Nesse sentido, é descrita uma classificação desses modelos a seguir: (1) modelos probabilísticos, (2) modelos de *Machine Learning* (ML) e (3) modelos de *Deep Learning* (DL).

2.4.1 Modelos Probabilísticos

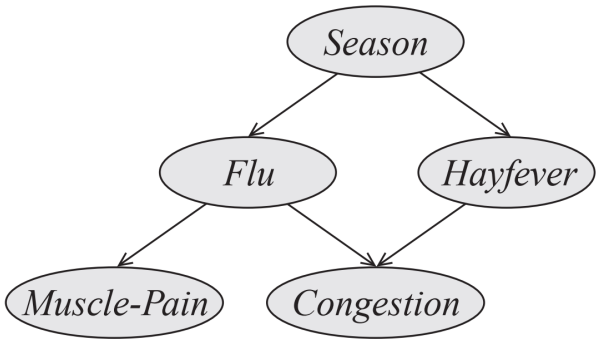
Definição: Um modelo probabilístico é uma representação matemática da incerteza. Ele permite fazer previsões ou decisões em situações em que os resultados não são certos, mas têm algum grau de aleatoriedade associado a eles. Esses são usados em vários campos, como estatística, aprendizado de máquina e inteligência artificial, para lidar com informações incertas ou incompletas. No modelo probabilístico, normalmente são definidas as probabilidades para diferentes eventos ou resultados, com base em dados ou

suposições disponíveis. Essas probabilidades são usadas para calcular expectativas, fazer previsões ou inferir quantidades desconhecidas (KOLLER; PFEFFER, 1998).

Diversas propostas têm optado por utilizar modelos estatísticos e empíricos baseados em probabilidades, pois um modelo probabilístico permite estabelecer modelos matemáticos para fazer suposições sobre o comportamento de uma amostra de dados aleatória por meio de probabilidades (KOLLER; FRIEDMAN, 2009). Esses modelos podem ser aplicados a variáveis aleatórias discretas, bem como variáveis contínuas.

O *Probabilistic Graphical Models* (PGM) foi definido por Koller and Friedman (2009) como *"aquele modelo que utiliza uma representação baseada em grafo como base para codificar de forma compacta uma distribuição complexa em um espaço de alta dimensão. Nessa representação, os nós correspondem às variáveis e as arestas correspondem às interações probabilísticas diretas entre elas"*. Ou seja, o PGM organiza a informação de acordo com as entidades correspondentes nos nós, enquanto as arestas armazenam as probabilidades que indicam o peso da relação entre as entidades. Esse tipo de modelo é baseado em inferências propostas a partir de uma hipótese em conjuntos de dados não muito grandes, pois as probabilidades requerem atualização constante.

Figura 2.2: Exemplo de grafo de modelo probabilístico para a gripe



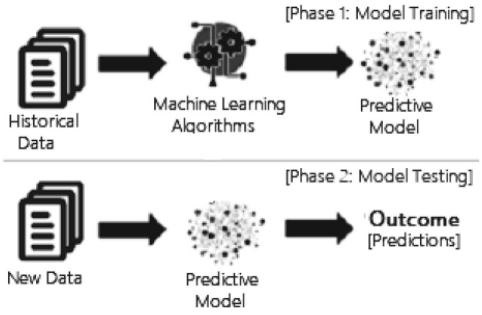
Fonte: Koller and Friedman (2009)

Por exemplo, a Figura 2.2 mostra um grafo para estruturar as variáveis da gripe. Nesse grafo, observa-se que não há interação direta entre *Muscle Pain* (dor muscular) e *Season* (temporada), mas ambas interagem diretamente com o *Flu* (gripe). De modo similar, *Hayfever* (febre alta) interage diretamente com *Congestion* (congestão).

2.4.2 Modelos de *Machine Learning* (ML)

Esse tipo de modelo é capaz de aprender a partir dos dados, considerando os atributos de entrada e os atributos-alvo. A Figura 2.3 mostra a estrutura geral de um modelo preditivo baseado em ML, em que o modelo é treinado a partir de dados históricos na fase 1 e o resultado é gerado na fase 2 para os dados de teste (SARKAR; BRUYN, 2021). Esses modelos podem empregar classificação, regressão, agrupamento de dados, engenharia de recursos para redução de dimensionalidade etc.

Figura 2.3: Fase de treinamento e teste no modelo de ML



Fonte: Sarkar and Bruyn (2021)

A seguir, são listados alguns modelos de ML usados para esse tipo análise:

- ***Adaptive Boosting* (*AdaBoost*)** - esse modelo executa um aprendizado conjunto que emprega uma abordagem iterativa para melhorar classificadores ruins aprendendo a partir dos seus erros (FENG et al., 2020). O *Adaboost* usa *ensemble* sequencial, consequentemente o classificador final combina muitos classificadores de baixo desempenho para obter um classificador de alta precisão.
- ***Extreme Gradient Boostin*g (*XGBoost*)** - esse modelo executa um conjunto de algoritmos de aprendizado que gera um modelo final baseado em uma série de modelos individuais, tipicamente árvores de decisão. O gradiente é usado para minimizar a função de perda, semelhante a como as redes neurais usam gradiente de descida para otimizar pesos (WADE; GLYNN, 2020). O *XGBoost* é, portanto, uma forma mais complexa de aumento de gradiente que leva em consideração mais aproximações detalhadas ao determinar o melhor modelo.

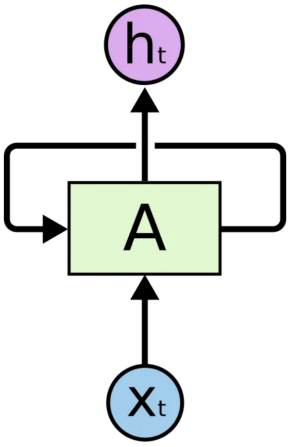
2.4.3 Modelos de *Deep Learning* (DL)

Esse tipo de modelo é utilizado em diversas áreas do *E-commerce* (como, por exemplo, personalização da experiência do cliente, automação de processos, detecção de fraudes etc.) lidando com limitações de escalabilidade ao utilizar grandes conjuntos de dados. Comparados com um modelo de aprendizado tradicional, os modelos de DL possuem um aprendizado mais profundo por meio de diferentes arquiteturas, funções de ativação ou técnicas de regularização; o que permite capturar dependências temporais no comportamento do cliente e compreender a evolução do comportamento do cliente.

Recurrent Neural Network (RNN)

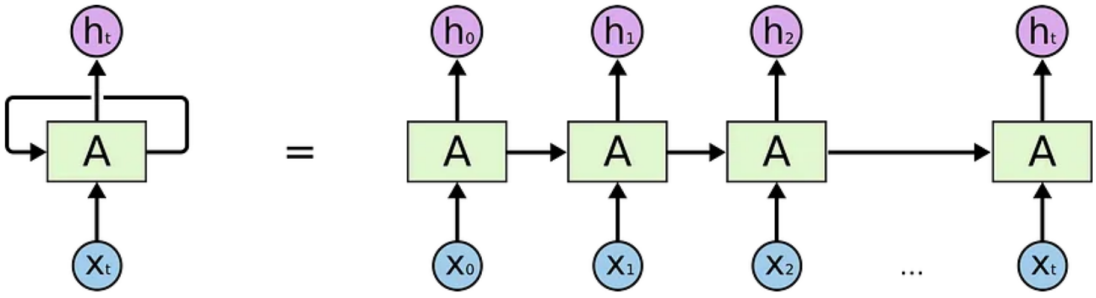
Na Figura 2.4, observa-se uma célula padrão de uma rede neural, em que para uma entrada x_t é gerada uma saída h_t, o que permite que as informações sejam passadas de uma etapa a outra. Então, uma rede neural recorrente pode ser pensada como múltiplas células interconectadas que transmitem a mensagem, como se mostra na Figura 2.5.

Figura 2.4: Célula padrão



Fonte: Yu et al. (2019)

Figura 2.5: Esquema da rede neural recorrente



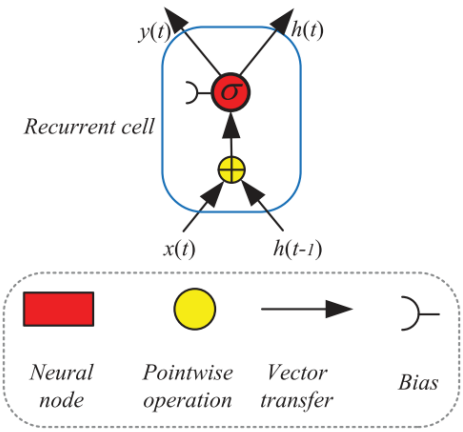
Fonte: Yu et al. (2019)

As RNNs são utilizadas em problemas que envolvem a análise de sequências dinâmicas e variantes no tempo, ou seja, podem processar comportamentos não lineares considerando informações espaço-temporais (YU et al., 2019). Com a conexão cíclica das células, é possível atualizar o estado atual baseado em estados anteriores, e assim o processo de *feedback* consegue lembrar e conectar informações por meio de conexões arbitrárias. A Figura 2.6 ilustra um esquema desse tipo de célula, sendo as expressões matemáticas de uma célula recorrente padrão descritas na Equação 2.1.

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b), y_t = h_t, \tag{2.1}$$

Em que x_t denota a entrada, h_t denota a informação recorrente e y_t denota a saída da célula no tempo t; W_h e W_x são os pesos; e b é o viés.

Figura 2.6: Esquema da célula recorrente padrão



Fonte: Yu et al. (2019)

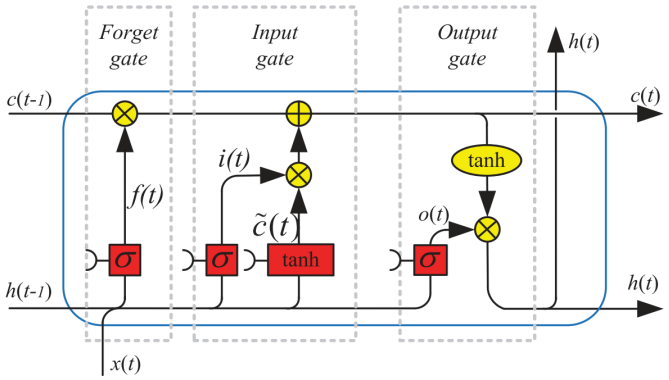
Por outro lado, as RNNs com esse tipo de células não conseguem lidar com as dependências de longo prazo, por isso foi proposta a memória de longo prazo. A retropropagação torna-se mais pesada, levando a problemas com a gradiente (ALOM et al., 2019). Por isso, existem outras variações que permitem lidar com esse problema, conforme os modelos descritos a seguir.

Long Short-Term Memory (LSTM)

O modelo LSTM, proposto por Hochreiter and Schmidhuber (1997) para lidar com *"long-term dependencies"*, é uma extensão do RNN (YU et al., 2019). Ele é usado principalmente para problemas de processamento de texto, que permite aos neurônios escolherem entre manter ou excluir as informações necessárias. Ao contrário das abor-

dagens RNN puras, o LSTM ativa sua capacidade de manter informações essenciais e identificar padrões sequenciais variáveis no tempo. A Figura 2.7 apresenta as conexões internas da célula, sendo matematicamente descritas na Equação 2.2.

Figura 2.7: Arquitetura da célula LSTM com *forget gate*



Fonte: Yu et al. (2019)

$$
\begin{aligned}
f_t &= \sigma(W_{fh}h_{t-1} + W_{fx}x_t + b_f), \\
i_t &= \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i), \\
\tilde{c}_t &= tanh(W_{\tilde{c}h}h_{t-1} + W_{\tilde{c}x}x_t + b_{\tilde{c}}), \\
c_t &= f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t, \\
o_t &= \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o), \\
h_t &= o_t \cdot tanh(c_t),
\end{aligned}
\tag{2.2}
$$

Em que, c_t denota o estado da célula do LSTM; W_i, W_c e W_o são os pesos, e o operador "$\cdot$" denota a multiplicação ponto de dois vetores. Observa-se que o *forget gate* f_t pode decidir quais informações serão descartadas no estado da célula. Quando o valor de f_t é 1, ele mantém essa informação; enquanto isso, um valor 0 significa que ele elimina todas as informações (SHRESTHA; MAHMOOD, 2019).

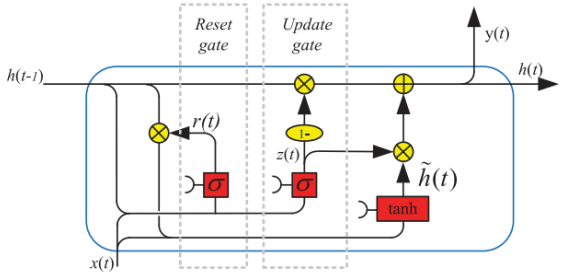
Gated Recurrent Unit (GRU)

A arquitetura GRU, proposta por Cho et al. (2014), é uma variação da arquitetura LSTM caracterizada por sua simplicidade e baixa complexidade. A capacidade de aprendizado da célula LSTM é superior à da célula recorrente padrão, porém os parâmetros adicionais aumentam a carga computacional. Para reduzir o número de parâmetros, a célula GRU integra o *forget gate* e o *input gate* da célula LSTM como um *update gate*, ou seja, a célula GRU tem apenas dois portões: o *update gate* e o *reset gate*. Esses portões

mantêm um registro das informações mais relevantes e mediante dois vetores que decidem quais informações são passadas para a saída (SHRESTHA; MAHMOOD, 2019).

A Figura 2.8 mostra a arquitetura da célula GRU, enquanto as expressões matemáticas da célula GRU são mostradas na Equação 2.3.

Figura 2.8: Arquitetura da célula GRU.



Fonte: Yu et al. (2019)

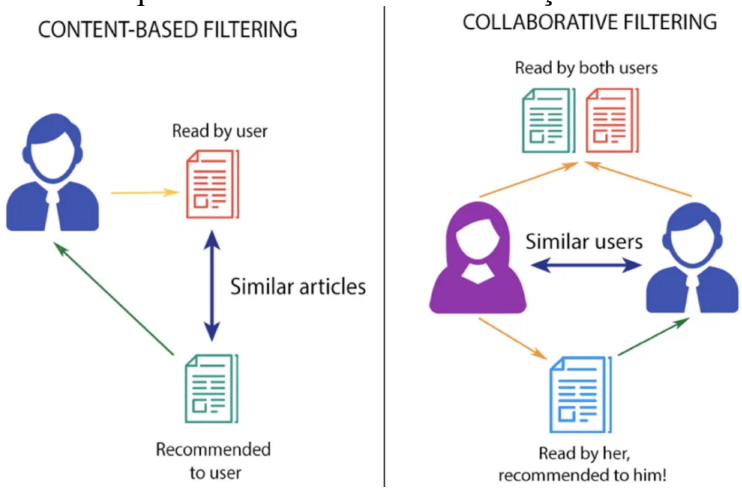
$$
\begin{aligned}
r_t &= \sigma(W_{rh}h_{t-1} + W_{rx}xt + b_r), \\
z_t &= \sigma(W_{zh}h_{t-1} + W_{zx}x_t + b_z), \\
\tilde{h}_t &= tanh(W_{\tilde{h}h}(rt \cdot h_{t-1}) + W_{\tilde{h}x}x_t + b_z), \\
h_t &= (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t
\end{aligned}
\tag{2.3}
$$

2.5 Sistema de Recomendação Explicável

Um sistema de recomendação (SR) tradicional fornece um item ou lista de itens. Em princípio, o processo se inicia aprendendo a representação do usuário u_i e do item v_j. Logo, uma função de pontuação é definida como $f : u_i \times v_j \rightarrow \hat{y}_{i,j}$ que modela a preferência do u_i pelo v_j. Depois, a recomendação é gerada mediante um *ranking* das pontuações de preferência dos itens baseado na função anterior (GUO et al., 2020).

Os SRs têm sido usado em diferentes domínios. A Figura 2.9 mostra como os SRs podem ser classificados: (1) Recomendação baseada em conteúdo ou *Content-Based Filter*, que usa informações de características dos itens (cor, categoria, tamanho etc.); (2) *Collaborative Filter* (CF) ou Filtragem colaborativa, que usa informações de interações entre usuário e itens; e (3) Recomendação híbrida, que combina os dois tipos de enfoques (AGGARWAL et al., 2016). Modelos baseados no CF são a estratégia mais popular, porém o desenvolvimento de arquiteturas de recomendação está evoluindo drasticamente devido à enorme quantidade de dados a serem analisados nos diversos cenários.

Figura 2.9: Esquema de sistema de recomendação baseado em CF



Fonte: Gao et al. (2023)

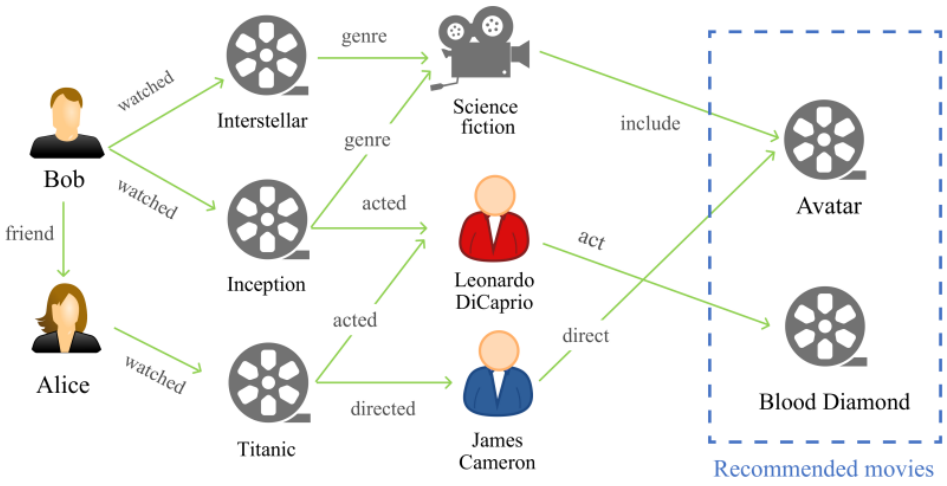
Os sistemas de recomendação fornecem resultados personalizados ao usuário, mas, no caso de uma recomendação explicável, também são adicionadas informações sobre o porquê o item é recomendado. Essas informações podem ser, por exemplo, quando, onde, quem, o quê e por quê (ZHANG; CHEN et al., 2020). Uma recomendação explicável pode ser expressada de duas formas: (1) a fonte de informação das explicações é exibida, por exemplo, em uma explicação de sentença textual ou em uma explicação visual; (2) o modelo deve descobrir as informações da recomendação explicável, por exemplo, utilizando o vizinho mais próximo, fatoração de matrizes, modelos de grafos, aprendizado profundo, mineração de regras de associação, entre outros.

2.6 Grafo de Conhecimento

O grafo de conhecimento tem sido um modo simples de organizar e representar a informação, em que os nós representam as entidades, enquanto as arestas representam a relação entre a entidade principal e a entidade final. Por exemplo, <Pelé, jogador, futebol> indica o fato de que Pelé foi um jogador de futebol, ou <Fernanda Montenegro, nacionalidade, Brasil> indica o fato de que Fernanda Montenegro é uma atriz brasileira. Esse grafo é uma rede heterogênea que armazena diferentes tipos de nós e relações, como é mostrado na Figura 2.13. Os atributos das representações podem ser obtidos seguindo as arestas no grafo, enquanto as relações de alto nível de entidades podem ser descritas por caminhos por meio das arestas. Até o momento, o grafo de conhecimento têm sido

aplicado em vários cenários, incluindo motores de busca, sistemas de recomendação, sistemas de Pergunta & Resposta, entre outros. Existem grafos de conhecimento abertos que fornecem conhecimento externo, por exemplo, *Freebase*, *DBpedia*, entre outros.

Figura 2.10: Exemplo de grafo de conhecimento no sistema de recomendação



Fonte: Guo et al. (2020)

O grafo de conhecimento $\mathcal{G} = (V, E)$, cujos nós são entidades e as arestas são as relações estabelecidas como tripletas do tipo sujeito-propriedade-objeto. Cada aresta *(head_entity, relation, tail_entity)* é representada como $< e_h, r, e_t >$, em que r indica a relação entre a entidade e_h com a entidade e_t. Esse pode ser considerado como uma instância de um *Heterogeneous Information Network* (HIN) e pode ser de dois tipos:

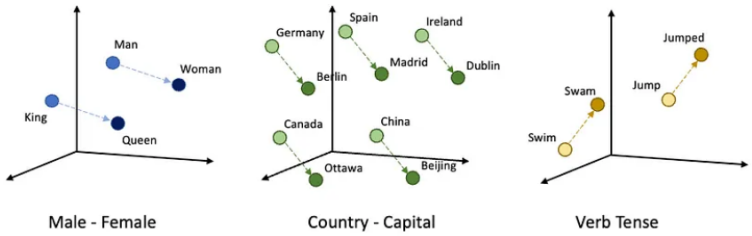
- *Grafo de conhecimento de Item* - nesse grafo, são relacionados os itens e as entidades associadas ao item. As arestas representam relações em nível do atributo, como marca ou categoria, ou ações do usuário (como, por exemplo, "co-visualização").
- *Grafo de conhecimento de User-item* - nesse grafo, os usuários, os itens e suas entidades associadas são os nós. Ademais, as relações entre o usuário e o item também são incluídas (como, por exemplo, "comprar", "clicar" e "mencionar").

A seguir, são descritas as principais definições relacionadas com o grafo de conhecimento que precisam ser esclarecidas para a compreensão do trabalho:

- **Heterogeneous Information Network (HIN)** - é um grafo direcionado $G = (V, E)$ com uma função de mapeamento do tipo entidade $\phi : V \rightarrow \mathcal{A}$ e uma função de mapeamento do tipo link $\psi : E \rightarrow \mathcal{R}$. Cada entidade $v \in V$ pertence a um tipo de entidade $\phi(v) \in \mathcal{A}$ e cada link $e \in E$ pertence a um tipo de relação $\psi(e) \in \mathcal{R}$. Por fim, o número de tipos de entidades $|\mathcal{A}| > 1$ e o número de relações $|\mathcal{R}| > 1$.

- **Meta-caminho** - um meta-caminho $\mathcal{P} = A_0 \xrightarrow{R_1} A_1 \xrightarrow{R_2} A_2 \ldots \xrightarrow{R_k} A_k$ é um caminho definido no grafo $G_T = (\mathcal{A}, \mathcal{R})$, que define uma relação composta $R_1 R_2 \ldots R_k$ entre o tipo A_0 e A_k. Ou seja, é uma sequência de relações que conecta dois pares de nós no HIN e pode ser usada para extrair recursos de conectividade no grafo.
- **Meta-grafo** - ao contrário do meta-caminho que define apenas uma relação sequencial, o meta-grafo é uma combinação de diferentes meta-caminhos, portanto, ele contém informações estruturais mais precisas entre as entidades.
- **Representação do *Embedding*** - Os dados categóricos se referem a recursos de entrada que representam um ou mais itens discretos de um conjunto finito de opções, por exemplo, pode ser o conjunto de filmes que um usuário assistiu. Os *embeddings* são vetores de baixa dimensionalidade que codificam a informação estrutural do grafo. No pequeno espaço multidimensional, os *embeddings* podem agrupar itens semanticamente semelhantes e manter os itens diferentes um do outro. A posição (distância e direção) no espaço vetorial pode codificar a semântica. Na Figura 2.11, é apresentada a visualização de *embeddings* reais, em que são apresentadas as relações geométricas que capturam relações semânticas.

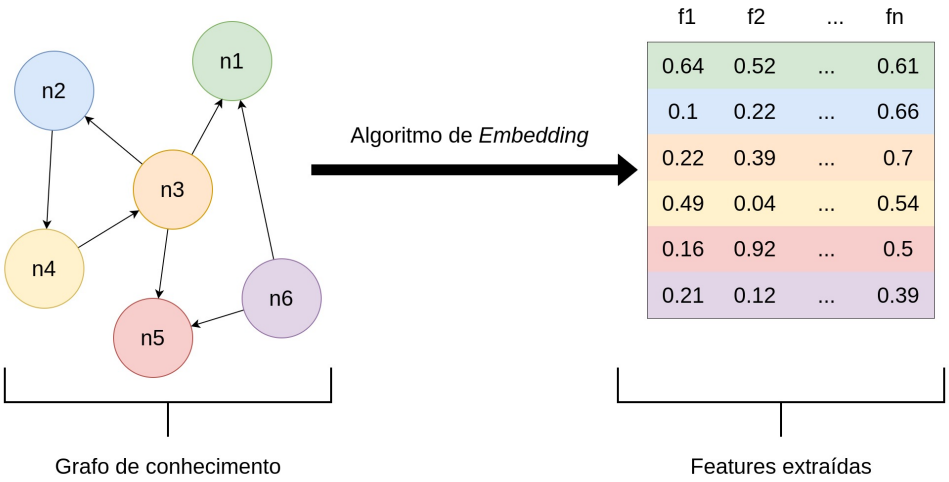
Figura 2.11: Exemplo da representação de *embeddings*



Fonte: Dolphin, Smyth and Dong (2022)

- **Knowledge Graph Embedding (KGE)** - deve transformar o grafo de conhecimento $\mathcal{G} = (V, E)$ em um espaço de baixa dimensão. Após o processo do *embedding*, cada componente do grafo, incluindo a entidade e a relação, é representado por um vetor d_dimensional que deve preservar as propriedades inerentes do grafo. A incorporação de baixa dimensão ainda preserva a propriedade inerente do gráfico, que pode ser quantificado por significado semântico ou proximidade de ordem superior no grafo, como é apresentado na Figura 2.12.

Figura 2.12: Exemplo de grafo de conhecimento baseado em *embeddings*



Fonte: Lima (2022)

- ***H-hop Neighbor*** - os nós do grafo podem ser conectados com um caminho *multi-hop*: $e_0 \xrightarrow{r_1} e_1 \xrightarrow{r_2} \ldots \xrightarrow{r_H} e_H$, neste caso, e_H é o vizinho H-hop de e_0, que pode ser representado como $e_H \in \mathcal{N}_{e_0}^H$. Observe que $\mathcal{N}_{e_0}^0$ denota o próprio e_0.
- **Entity Triplet Set** - o conjunto tripleto de uma entidade $e \in G$ é definido como:

$$\mathcal{S}_e^k = \{(e_h, r, e_t) | (e_h, r, e_t) \in \mathcal{G}, e_h \in \mathcal{N}_e^{k-1}\},$$
$$k = 1, 2, \ldots, H.$$

  Pode ser considerado como múltiplas camadas de trigêmeos contendo entidades de vizinhos de 1 salto para vizinhos de salto H.

2.7 Classificação de SR Baseados em Grafo de Conhecimento

Esta seção recopila trabalhos que fornecem *frameworks* de sistemas de recomendações baseados em grafo de conhecimento. A seguir, é descrita a classificação proposta por Guo et al. (2020) em três categorias: (1) métodos baseados em *embeddings*, (2) métodos baseados em conexão e (3) métodos baseados em propagação.

2.7.1 Métodos Baseados em *Embeddings*

Esse tipo de método usa o processo de *embeddings* para enriquecer a representação dos itens e usuários no grafo de maneira similar. A fim de cumprir esse processo, são incluídos dois módulos: (1) Módulo do *Graph Embedding*, que permite que as representações das entidades e relações sejam aprendidas no grafo; e o (2) Módulo de recomendação, em que, a partir do aprendizado de *features*, pode-se prever a preferência do usuário u_i pelo item v_j. A seguir, são listadas três abordagens em relação aos módulos:

1. ***Two-stage Learning*** - nessa abordagem, as representações das entidades e das relações são aprendidas no módulo do *Graph Embedding* utilizando a técnica de *Knowledge Graph Embedding* (KGE). Depois, os *embeddings* são inseridos no grafo pré-treinado do módulo de recomendação para fazer as recomendações. Cada processo é realizado sequencialmente um após o outro.
2. ***Joint Learning*** - essa abordagem utiliza um treinamento *end-to-end*, em que ambos os módulos são treinados de maneira simultânea. Ademais, é utilizado um *regularization term* para evitar o *overfitting*, mas às vezes é necessário realizar o *fine-tuned*.
3. ***Multi-task Learning*** - essa abordagem é motivada pela descoberta de semelhanças nas entidades relacionadas no *grafo de user-item*. O compartilhamento de estruturas ajuda na generalização do modelo, mas requer treinamento *end-to-end*.

2.7.2 Métodos Baseados em Conexão

O principal objetivo é estabelecer *connection patterns* no grafo para explorar os relacionamentos entre as entidades, nesse sentido, dois métodos são apresentados a seguir:

1. **Baseado em meta-estruturas** - esses métodos utilizam uma meta-estrutura no grafo, ou seja, o meta-caminho e o meta-grafo, para calcular o semelhança entre as entidades. Para fazer a recomendação, leva-se em consideração os interesses dos usuários semelhantes ou dos itens semelhantes encontrados no histórico de interações. O objetivo é identificar aquelas entidades com alta similaridade baseadas no meta-caminho que devem estar próximas no espaço latente. Esse tipo de método é de baixa complexidade e, desse modo, é fácil de implementar, pois os meta-caminhos são definidos manualmente, mas os métodos podem deixar de ser escaláveis à medida que se acrescentam informações no domínio da recomendação.

2. **Baseado no *Path-embedding*** - o objetivo é codificar o *path-embedding* entre cada par usuário-item ou par item-item em vetores para depois aprendê-los. Ao contrário do foco anterior, os *path-embeddings* conseguem ser modelados explicitamente, o que permite descobrir relações ocultas entre as entidades. Esse foco não é mais escalável quando já é necessário empregar mais cálculos na enumeração de caminhos, assim como mais *path-embeddings* precisam ser aprendidos.

2.7.3 Métodos Baseados em Propagação

Esses tipos de métodos associam as representações das entidades e das relações, assim como os *path embeddings* de alta ordem, para conseguir recomendações mais personalizadas. A ideia central é executar a propagação de *embeddings* por meio de estruturas complexas como *Graph Neural Network* (GNN), desse modo, as representações das entidades podem ser aprimoradas (WANG et al., 2022). Esses métodos são divididos de acordo com o tipo de entidade que é refinada, tais como:

1. **Refinamento da Representação do Usuário** - primeiro, é construído o *grafo de item*, que associa os itens visitados e os itens candidatos. Os usuários podem ser representados pelos itens visitados, bem como pelos itens que pertencem aos vizinhos *multi-hop* desses itens. Os itens são usados como "sementes" na propagação, assim, a representação do usuário u_i é formulada como:
    1. calcular-se a representação do usuário, levando em consideração seus vizinhos *multihop* associados, agregando entidades em cada camada do conjunto tripleto $\mathcal{S}_{u_i}^k = (k = 1, 2, \ldots, H)$.
    2. combina-se essas informações para a representação do usuário final o_u.
3. **Refinamento da Representação do Item** - nesse refinamento devem ser aprendidas apenas as representações de alta ordem do item-candidato v_j agregando *embeddings* dos vizinhos *multi-hop* do item. Durante o processo de propagação, é adotado o mecanismo de *attention graph*, que estabelece um nível de importância da informação para cada vizinho por meio de pesos para cada par de nós. Como resultado, podem ser descobertos múltiplos itens por meio das relações no fluxo de informações. A propagação é formulada como indicam as Equações 2.4 e 2.5:

1. agregam-se os vizinhos de uma entidade e_i:

$$\boldsymbol{e}_{\mathcal{N}_i}^{h-1} = AGG(\boldsymbol{e}_m^{h-1}) \forall m \in \mathcal{N}_i, h = 1, 2, \ldots, H \tag{2.4}$$

2. atualiza-se a representação de $h - order$ da entidade com o *embedding* do vizinho de $h - 1 - order$ e seu próprio *embedding*:

$$\boldsymbol{e}_i^h = g(\boldsymbol{e}_{\mathcal{N}_i}^{h-1}, \boldsymbol{e}_i^{h-1}), h = 1, 2, \ldots, H \tag{2.5}$$

3. **Refinamento da Representação do *User-Item*** - no grafo, os usuários, itens e suas entidades associadas são conectados, além de considerar a interação usuário-item como um tipo de relação adicional. Os *embeddings* do usuário e os *embeddings* do item podem ser refinados com seus vizinhos correspondentes durante o processo de propagação, conforme nas Equações 2.4 e 2.5.

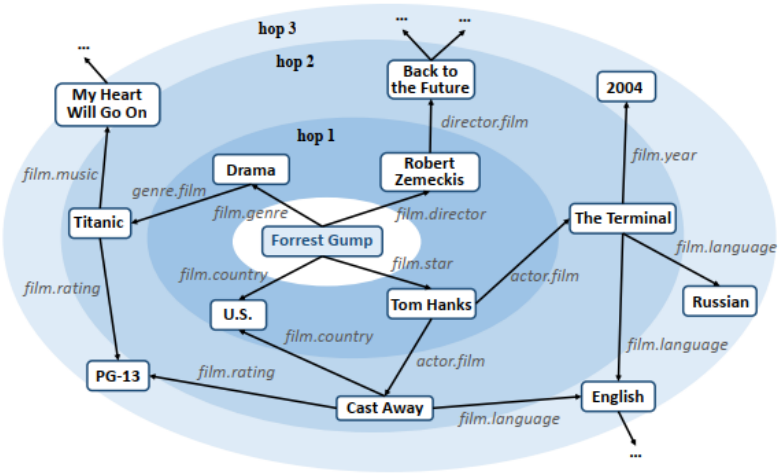
Em síntese, esses métodos possuem alta complexidade à medida que o grafo cresce. Zhao et al. (2019) propõem utilizar o *graph convolutional operation* para aplicar amostragem vizinha em cada camada e reduzir a complexidade. Porém, a amostragem aleatória leva à perda de informação, deixando de explorar completamente o grafo.

2.8 *Frameworks de SR baseados em Grafo de Conhecimento*

RippleNet

Wang et al. (2018a) propõem *RippleNet*, o primeiro *framework* que lida como as limitações dos métodos baseados em *embeddings*, assim com as limitações dos métodos baseados em conexões, porque consegue propagar automaticamente as preferências potenciais dos usuários e explora seus interesses hierárquicos no grafo. Nesse contexto, os *embeddings* agem iterativamente no *k-hops* para prever a probabilidade de chegar ao item de preferência. Ademais, os interesses já conhecidos são utilizados como uma semente definida no grafo; eles são estendidos iterativamente ao longo dos caminhos do grafo para possibilitar a descoberta dos interesses potenciais em relação a um item-candidato. A propagação das preferências é por *hop* como mostra a Figura 2.13, depois obtém-se uma distribuição de preferência resultante do usuário sobre o grafo.

Figura 2.13: Exemplo da estrutura de um grafo no *RippleNet*



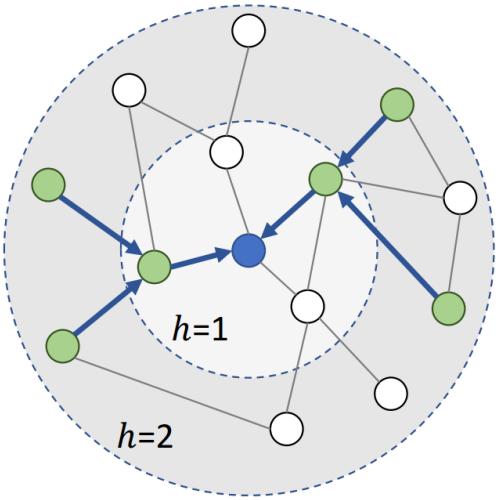
Fonte: Wen et al. (2018)

Uma preocupação sobre conjuntos de *ripples* é com relação ao seus tamanhos que aumentam de acordo com o número de saltos k. Então, Wang et al. (2018a) definem o conjunto de vizinhos de tamanho fixo para reduzir a sobrecarga computacional, pois nem toda informação deveria ser explorada. *RippleNet* é a primeira proposta que lida com as limitações dos primeiros focos, o que sugere considerá-lo como o ponto de partida para explorar os métodos baseados na propagação.

Knowledge Graph Convolutional Networks (KGCN)

Wang et al. (2019b) propõem KGCN, um *framework* baseado em redes convolucionais que capturam efetivamente o relacionamento entre itens, minerando seus atributos associados no grafo. Ele é focado na descoberta automática das informações de estrutura de alta ordem e informações semânticas do grafo. Dessa forma, os vizinhos de cada entidade são selecionados e suas informações são extraidos ao calcular a representação de uma determinada entidade, assim os interesses potenciais são capturados dos interesses de longa distância dos usuários. No grafo de conhecimento, a quantidade de entidades relacionadas diretamente podem variar significativamente. Para manter o padrão computacional mais eficiente, seleciona-se uniformemente um conjunto de vizinhos de tamanho fixo para cada entidade, em vez de usar todos os vizinhos. A Figura 2.14 dá um exemplo do espaço de exploração de duas camadas para uma determinada entidade.

Figura 2.14: Exemplo da estrutura de um grafo no KGCN



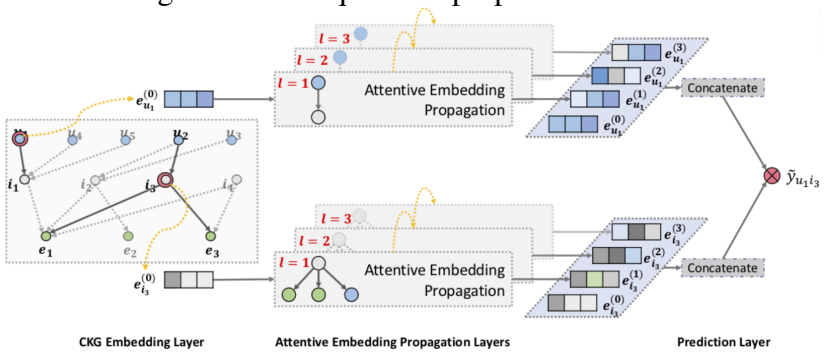
Fonte: Wang et al. (2019b)

KGCN reduz a complexidade em grande medida porque seleciona uniformemente por meior de um tamanho fixo de vizinhos para cada entidade no grafo, bem como *mini-batch* para treinamento da rede. Ambos os aspectos são favoráveis para *datasets* extensos. Porém, KGCN pode apresentar *overfitting*, visto que apenas é baseado nas interações de usuário-item considerando, por exemplo, relações duplicadas.

Knowledge Graph Attention Network for Recommendation (KGAT)

Wang et al. (2019) propõem KGAT, um *framework* que executa a propagação de *embeddings* recursivos a fim de atualizar o *embedding* de um nó com base nos *embeddings* de seus vizinhos para capturar conectividades de alta ordem. Além disso, emprega o *attention mechanism* para aprender o peso de cada vizinho durante uma propagação, de modo que os pesos das propagações em cascata possam revelar a importância de uma conectividade. Na Figura 2.15, são representadas as três camadas principais de KGAT: (1) *CKG Embedding Layer*, que parametriza cada nó como um vetor, ou seja, constrói os *embeddings* considerando as entidades e relações dos triplos entre usuário e item, (2) *Attentive Embedding Propagation Layers*, que propaga recursivamente os *embeddings* ao longo da conectividade de alta ordem empregando *Graph Attention Network* (VELICKOVIC et al., 2017) para aprender o peso (importância) de cada conexão com cada vizinho, e (3) *Prediction layer*, que agrega as representações do usuário e item de todas as camadas de propagação, para gerar o *score* entre usuário e item.

Figura 2.15: Arquitetura proposta do KGAT



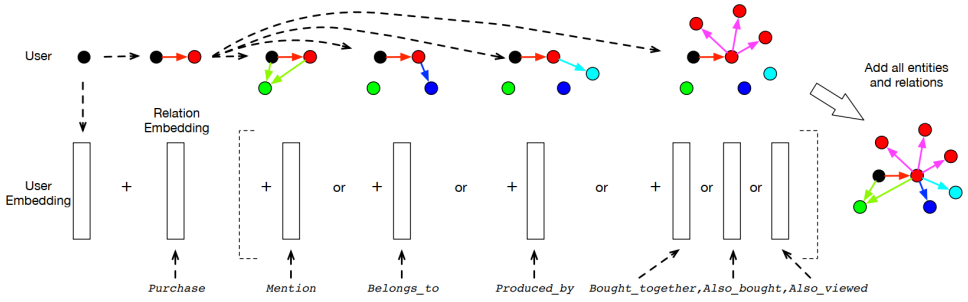
Fonte: Wang et al. (2019)

KGAT utiliza *Graph Neural Networks* em que é aplicada a amostragem vizinha em cada camada para reduzir a complexidade, mas a amostragem aleatória não explora completamente o grafo. Além disso, esse *framework* inclui os *embeddings* pré-treinados aprendidos na camada de CKG, em que o vetor é inicializado aleatoriamente.

Learning heterogeneous knowledge base embeddings for explainable recommendation

Ai et al. (2018) propõem *Explainable Collaborative Filtering Knowledge Graph* (ECFKG), um *framework* baseado no CFKG, que integra o Collaborative Filtering (CF) com a aprendizagem de *embeddings*. Assim, o *grafo de user-item* codifica as interações entre usuário e as propriedades do item como uma estrutura de grafo relacional. Uma entidade pode ser associada a uma ou mais entidades por meio de uma única ou múltiplas relações, por isso é proposto separar a modelagem da entidade e da relação para a filtragem colaborativa. Nesse sentido, cada entidade é projetada no espaço latente e cada relação é tratada como uma função de tradução que converte uma entidade em outra. A Figura 2.16 mostra o processo de construção do grafo, em que o *embedding* de cada entidade representada com um vetor latente, e cada relação é modelada como uma translação linear de uma entidade para outra, parametrizada pela incorporação da relação.

Figura 2.16: Construção do grafo de conhecimento de acordo com ECFKG



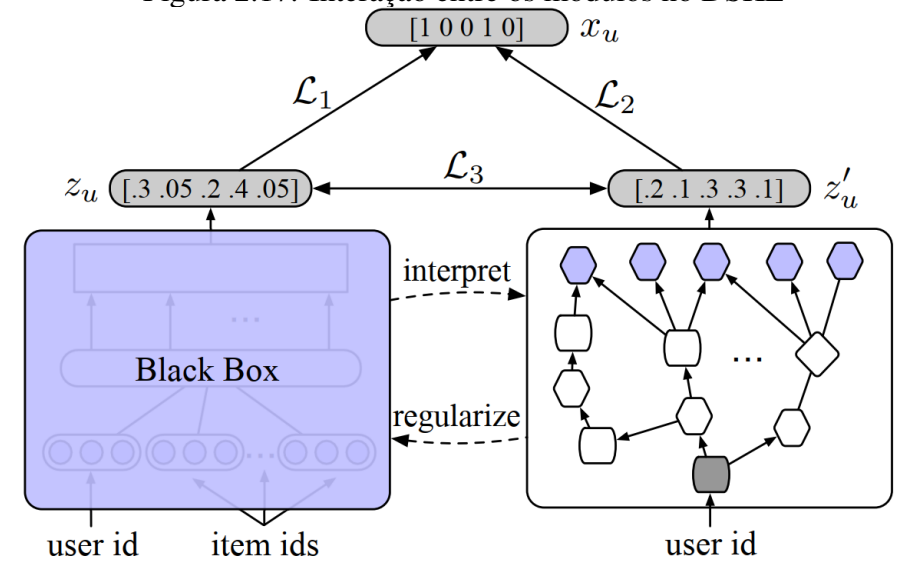
Fonte: Ai et al. (2018)

Esse *framework* aprimora os *embeddings* das entidades a partir de um treinamento *end-to-end*, porém as relações de alta ordem são ignoradas durante esse processo. ECFKG é um *framework* muito flexível porque pode ser aplicado em diversos cenários.

Distilling Structured Knowledge into Embeddings (DSKE)

Zhang et al. (2020) propõem DSKE, um *framework end-to-end* para complementar as limitações dos modelos baseados nos *embeddings* e caminhos. Na Figura 2.17, observa-se que o modelo baseado em *embeddings* é regularizado pelo conhecimento estruturado em grafos, que também são codificados no modelo baseado em conexões, para evitar mínimos locais pouco generalizáveis, assim como os caminhos aprendidos são otimizados interpretando os *embeddings*. Essa abordagem é de "destilação do conhecimento" no grafo, com a diferença de que o modelo do professor (baseado em conexões) e o modelo do aluno (baseado em *embeddings*) aprendem um com o outro simultaneamente, como mostra a Figura 2.17. Ou seja, o modelo do aluno destila o conhecimento estruturado do modelo baseado em conexões. Ao mesmo tempo, o modelo do professor aprimora seu conhecimento codificado os caminhos de raciocínio, sintetizando as previsões feitas no modelo baseado em *embeddings*.

Figura 2.17: Interação entre os módulos no DSKE



Fonte: Zhang et al. (2020)

Ademais, essa abordagem propaga informações de dados não observados para resolver problemas de dados ausentes em sistemas de recomendação. Certamente nem todos os itens não observados são irrelevantes para um usuário, mas o modelo baseado em conexões pode aprender a propagar rótulos esparsos em itens não observados, em que os itens relevantes não observados provavelmente receberão pontuações de probabilidade relativamente mais altas em relação aos irrelevantes.

O DSKE utiliza parâmetros compartilhados para ambos os modelos por meio de um termo de regularização mútua na função objetiva. O aprendizado com os rótulos aumentados supera o problema de escassez de dados, assim como permite um aprendizado eficiente da estrutura e, portanto, melhora a precisão das recomendações. Embora esse *framework* tenha uma metodologia de aprendizagem distinta, a complexidade é mantida.

3 TRABALHOS RELACIONADOS

Este Capítulo apresenta o contexto no qual essa dissertação está inserida. Desse modo, ele é dividido em duas seções: (1) Análise preditiva no *E-commerce* e (2) *Frameworks* de Sistemas de Recomendação baseados em grafo de conhecimento. Cada seção apresenta a coleta de trabalhos relacionados, além de uma descrição dos *baselines* (no caso da primeira seção). Por fim, é descrito um comparativo desses trabalhos.

3.1 Análise Preditiva no *E-commerce*

Esta seção apresenta diversos trabalhos focados na análise do preditivo no *E-commerce* abordando três tipos de modelos: (1) modelos probabilísticos, (2) modelos de *Machine Learning* (ML) e (3) modelos de *Deep Learning* (DL).

3.1.1 Modelos Probabilísticos

Uma abordagem focada na personalização da cesta foi resolvida por meio da fatoração de matrizes e cadeias de Markov (RENDLE STEFFEN; SCHMIDT-THIEME, 2010); essa proposta é orientada na análise sequencial de compras para prever a próxima ação. Yuan et al. (2017) propõem unir informações geográficas com informações temporais por meio de um modelo bayesiano não paramétrico denominado PRED; esse *framework* leva em consideração conjuntamente os dois tipos de informações, logo uma distribuição gaussiana é empregada para tratar a informação geográfica. No entanto, o tempo entre dois registros consecutivos é analisado por intervalos. Nesse sentido, Wen et al. (2018) propõem considerar informações temporais, geográficas, de pagamento e de categoria para modelar um modelo probabilístico. A abordagem centra-se na inferência dos relacionamentos para obter os parâmetros do modelo, em que a periodicidade é analisada pela *Fast Fourier Transform* (FFT), além, certos fatores típicos do comportamento de compra em séries temporais são aprendidos e modelados por uma distribuição gaussiana. Por meio deste trabalho, identifica-se a importância de ter outros tipos de informações para melhorar a previsão do comportamento de compra, por exemplo, a ordenação de dois padrões de mobilidade e a distribuição de compras de acordo com o limite de crédito.

Shopper (RUIZ; ATHEY; BLEI, 2020) é um modelo probabilístico sequencial que permite associar cada item por meio de atributos latentes de modo que a cesta é representada por pares de itens, ou seja, os itens que foram comprados um após o outro. Da mesma forma, as preferências do cliente são extraídas das interações dos diferentes tipos de itens que normalmente são adquiridos. Como resultado, são estabelecidas inferências para estimar as preferências dos itens e os preços. Li, Chen and Zhao (2021) propõem PRIMA++, um *framework* probabilístico, que consegue aprender as preferências do usuário a partir de dados limitados por meio dos atributos que possuem o mesmo *tradeoff*.

Em resumo, os modelos probabilísticos são empregados para extrair inferências dos dados, mas esse tipo de análise pode não ser eficiente quando os dados crescem porque alguns aspectos passam despercebidos, deixando de fora informações importantes.

3.1.2 Modelos de *Machine Learning* (ML)

Fleder and Shah (2020) propõem um algoritmo computacional iterativo de baixo custo, a fim de encontrar o conjunto de itens adquiridos, bem como a quantidade por meio do valor do pagamento. Depois uma matriz é construída para decompor os possíveis produtos que foram adquiridos na transação. O algoritmo é aplicado em dados anonimizados de cartões de crédito e débito utilizados em serviços, como *Netflix*, *Spotify*, *Apple* e *Chipotle*. Similarmente, Martínez et al. (2020) propõem extrair 274 *features* das transações de um cliente durante um mês. Como resultado, os *features* são computados por meio de *Logistic Lasso* e *Gradient Tree Bosting* para prever se o cliente fez ou não a compra. A proposta de Tabianan, Velu and Ravi (2022) utiliza *clustering K-Means* para segmentar clientes entre três *clusters*: tipo de evento, produtos e categorias. Desse modo, analisou-se os grupos que compartilham critérios semelhantes para identificar e focar no segmento de alta rentabilidade. Eles verificam que o trabalho com *datasets* de menor tamanho poderiam complicar a mineração de dados, por isso um processo de suavização e estandardização no pré-processamento ajuda a que esta limitação não afete na análise.

Amphawan, Lenca and Surarerks (2011) propóe um TR-CT, uma proposta de mineração dos $top - k$ itens baseada em *Tidsets* compactados para manter informações de ocorrência de padrões e descobrir k conjuntos de itens regulares com suportes mais altos. Desse modo, o foco esta na frequência de ocorrência de um conjunto de itens considerando a regularidade temporal dos comportamentos frequentes.

Tatti, Moerchen and Calders (2014) propõe uma nova estrutura teórica para redução de padrões por meio de uma técnica sem parâmetros para classificar conjuntos de itens que pode ser usada para abordagens top-k. Os experimentos demonstram que a medida de robustez pode ser utilizada para reduzir o número de padrões e que a classificação produz conjuntos de itens interessantes. Uma outra proposta aborda a previsão para antecipar o comportamento dos consumidores durante a pandemia adicionando o *feature "Effective"*, que indica se a compra esteve ou não em relação a COVID-19. Os classificadores empregam *Bagging* e *Boosting* para obter melhores resultados. Além disso, a análise de correlação é realizada para determinar as características mais importantes que influenciaram as compras online durante a pandemia (SAFARA, 2022). Wang et al. (2023) propõem *XGBoost*, um modelo *ensemble* para prever o comportamento de compra por meio de um modelo de valor do usuário LDTD (*"login_diff_time"* (LDT), *"login_time"* (LT) e *"login_day" (LD)*) que consegue diferenciar os tipos de usuários baseado nas contas de usuário. Por fim, o XGBoost funciona como um extrator dos *features* mais importantes.

Em resumo, os modelos de ML são promissores quando consideram uma fase prévia para a extração dos *features*, a fim de adquirir informações valiosas na fase de treinamento e detectar padrões no comportamento de compra.

3.1.3 Modelos de *Deep Learning* (DL)

Ładyżyński, Żbikowski and Gawrysiak (2019) propõem um modelo que identifica aqueles clientes que poderiam ter interesse em produtos bancários. Esse modelo utiliza uma janela móvel que permite a detecção de dependências sequenciais baseadas no tempo entre transações específicas. Por meio de modelos de classificação, como o *Random Forest* e Redes Neurais Profundas, são extraídos padrões significativos do histórico de transferências e dados transacionais dos clientes possibilitando avaliar a probabilidade de compra. A fim de *trackear* o comportamento de compra, Huang et al. (2019) propõem a *clusterização* de clientes. por meio de uma rede neural recorrente que estabelecem grupos de clientes de acordo com comportamentos semelhantes e posteriormente essa mesma rede descobre possíveis dependências entre compras de diferentes categorias.

No comportamento de compra, podem-se identificar padrões por meio de uma análise superficial, porém é necessário estudar as influências de comportamentos estacionários, conhecidos como microcomportamentos. Uma análise desse tipo acrescenta, por sua vez, uma arquitetura do tipo DL, como LSTM ou GRU (ZHOU et al., 2018). Outra

proposta utiliza a arquitetura LSTM para processar os dados brutos sem a necessidade de realizar um *feature engineering*, com a finalidade de apenas ajustar os hiperparâmetros, contudo esse modelo obtém bons resultados quando o *dataset* contém dados completos (SARKAR; BRUYN, 2021). Zhu et al. (2020) propõe um modelo híbrido de DL (EE-CNN) que combina os *embeddings* das entidades numa rede neural convolucional para conhecer o próximo item que deveria ser comprado. Além disso, ele sugere que o *feature* local de compra consegue aprimorar os resultados.

Em resumo, embora os modelos de DL sejam complexos, sua estrutura consegue ser viável para executar análises em dados sequenciais e variáveis. Dessa maneira, o comportamento de compra poderia ser muito bem modelado por meio de sua capacidade de memória a longo prazo, o que permite descobrir informações ocultas nos dados.

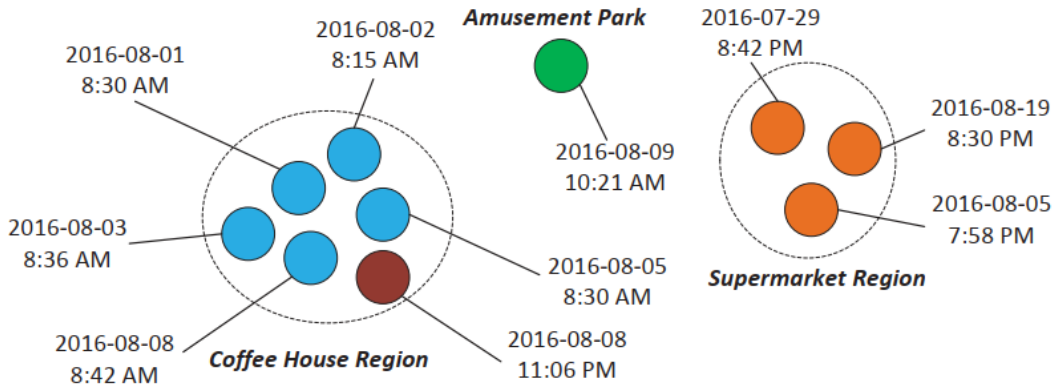
3.1.4 Baselines

Esta seção descreve os trabalhos escolhidos para o primeiro foco desta dissertação. Esses trabalhos foram escolhidos porque consideraram informações privadas dos usuários na análise (por exemplo, coordenadas geográficas para rastrear a mobilidade do usuário), assim como informações comuns (dia da compra, categoria, pagamento, entre outros). Desse modo, procurou-se trabalhos focados na previsão do próximo local, categoria ou pagamento de compra que apenas utilizaram esses tipos de informação.

PRED (Periodic Region Detection for Mobility Modeling of Social Media Users): Detecção Periódica de Região para Modelagem de Mobilidade de Usuários de Mídias Sociais

Yuan et al. (2017) propõem PRED, um modelo bayesiano não parametrizado para a descoberta de padrões levando em consideração as informações geográficas e temporais. Eles consideram modelar os *gaps time* entre dois registros consecutivos para não deixar de fora informações importantes, utilizando *clusters* que determinam os locais mais próximos que foram visitados entre os mesmos períodos em uma região; caso sejam reconhecidos padrões periódicos considera-se que o *gap time* se repetirá em outros períodos. A Figura 3.1 apresenta o agrupamento das atividades baseadas em sua localização.

Figura 3.1: Agrupamento das atividades do usuário por localização



Fonte: Yuan et al. (2017)

Esse modelo é difícil de parametrizar porque a localização é independente e muitas vezes não é possível achar um padrão. Muitas atividades similares podem ser realizadas em diversas localidades diferentes, por isso o período está ligado às atividades rotineiras. Para a estimativa dos parâmetros foi utilizado o *Markov Chain Monte Carlo*, mas primeiramente os registros foram organizados sequencialmente.

Problema: Sendo D_u o histórico de registros do usuário u, e cada registro $d_i \in D_u$ é uma dupla $di = \{li, ti\}$, em que l_i representa as coordenadas geográficas e t_i representa o tempo de postagem de d_i. Um usuário u possui um comportamento de visita periódica na região r com período T, em que é provável que ele visite r a cada T horas. As regiões são um conjunto de acumulados geográficos dentro dos quais a maioria dos registros em D_u são observados. Dado D_u, o objetivo é encontrar (1) um conjunto de regiões geográficas R_u nas quais o usuário u tem comportamento de visita periódica, e (2) o período t_r associado a cada região $r \in R_u$. A seguir, descreve-se os objetivos:

- **Periodic Region Detection:** A ideia principal é segmentar a série temporal em pequenos *chunks* e sobrepô-los com base em cada período candidato. Primeiro são extraídas as regiões por *Kernel Density Estimation* (KDE) e depois o período é estimado para cada região usando um periodograma. Visto que o tempo não é considerado na detecção de regiões, muitos registros de ruído ou mesmo registros com períodos diferentes ficam agrupados, dificultando a detecção de períodos. Assim, leva-se em consideração o *gap time* entre registros para lidar com o problema de escassez de dados, como se mostra na Figura 3.2.

Figura 3.2: Exemplo do tratamento dos *gap time*

ID	time	exact time	gap	rmdr.	ct.
d_1	D1 8:30 AM	8.50	–	–	–
d_2	D2 8:15 AM	32.25	23.75	23.75	1
d_3	D3 8:36 AM	56.60	24.35	24.35	1
d_4	D5 8:30 AM	104.50	47.90	23.90	2
d_5	D8 8:42 AM	176.70	72.20	24.20	3

Fonte: Yuan et al. (2017)

- **Localização:** Os locais fora de suas regiões regularmente visitadas são excluídos, porque são menos prováveis de serem gerados pelas distribuições geográficas gaussianas das regiões existentes. Apenas assume-se que ocorrerá um pico por cada região, mas uma região pode ter mais de um pico. Nesse sentido, empregam-se métricas para medir a proximidade entre *Geografical Gaussian Density* e as regiões que poderiam surgir no mesmo período.

Na Tabela 3.1, são relatados os objetivos da fase de experimentação proposta por Yuan et al. (2017). Também são apresentados os *datasets* utilizados, os métodos empregados e as métricas usadas para a avaliação. No artigo não são expostos os resultados numéricos, apenas figuras comparativas.

Tabela 3.1: Objetivos da fase experimental no PRED

Objetivo	***Dataset***	**Métodos**	**Métricas**
Detecção de periodicidade em dados de série temporal	Sintético	-*Fast Fourier Transform* (FFT) -Autocorrelação and FFT -Autocorrelação and FFT (Periódico) -Discrepância -Deteção do período	Acurácia
Detecção periódica de região em rados rspaço-temporais	Sintético	-Periodicidade -KernelDiscp -PREDsep -PRED	*F1 score*
Previsão de localização em dados de mídia social	-*Gowalla* -*Twitter*	-PMM -KernelDisp -PRED	-*Macro Error Distance* -*Micro Error Distance*

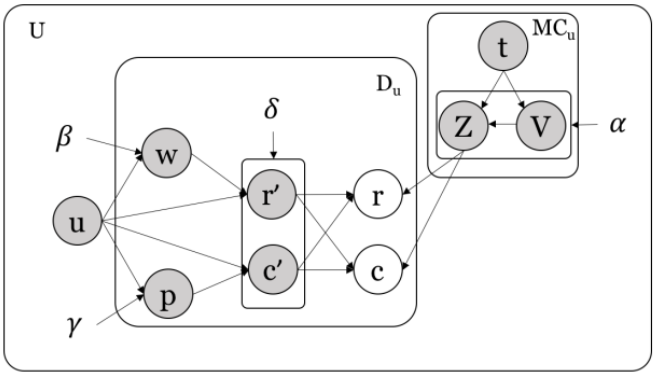
Customer Purchase Behavior Prediction from Payment datasets

Wen et al. (2018) propõem STPC-PGM (**s**patial, **t**emporal, **p**ayment and product **c**ategory in **P**robability **G**raphic **M**odel), um modelo probabilístico que modela a informação de espaço, tempo, pagamento, item e categoria. O objetivo é descobrir o comportamento, portanto, primeiro é efetuada uma clusterização dos perfis baseada em 30 *features*, depois, se calculam as probabilidades de compra numa região e categoria determinada e, por fim, o pagamento é previsto por meio de modelos de regressão.

Problema: Sendo D_u a coleção das transações do usuário u, e cada registro $di = \{u, li, ci, pi, ti\}$, em que l_i representa as coordenadas geográficas da loja, c_i representa a categoria do produto, p_i representa o valor do pagamento, e t_i o dia em que foi registrada a transação. O tempo t é modelado como uma variável discreta no formato $aa : mm : dd$ que categoriza os dias em duas classes: os dias de semana e os fins de semana. Dado D_u, o tempo atual t_{now} e o intervalo de tempo específico ft, os objetivos são: (1) encontrar um conjunto de categorias c e regiões r, e (2) prever o pagamento.

Na Figura 3.3, é mostrado o grafo STPC-PGM, em que existem duas variáveis latentes: região r e categoria c. Após de calcular as possíveis categorias e regiões de compra, é feita uma combinação delas para calcular o pagamento por meio de modelos de regressão. A seguir, são descritas as abordagens principais para executar essa proposta.

Figura 3.3: Representação gráfica do modelo STPC-PGM



Fonte: Wen et al. (2018)

- **Tracke do período:** O *Fast Fourier Transform* (FFT) é utilizado para conseguir o espectro e a autocorrelação circular de acordo com cada categoria e localidade. Depois, são extraídas uma lista das datas nas quais tiveram registros, logo são convertidas em uma sequência de sete dias (uma semana), em que o valor 1 corresponde aos dias que foram registradas, e 0 corresponde aos dias que não tiveram registro nenhum. Na Figura 3.4, é apresentado um exemplo de tratamento para o período.

Figura 3.4: Exemplo da segmentação do período

	1	2	3	4	5	6	7
2017/5/28~2017/6/03	1	0	1	0	0	0	0
2017/6/04~2017/6/10	1	0	0	0	0	0	0
2017/6/11~2017/6/17	0	0	0	0	0	0	0
2017/6/18~2017/6/24	1	0	1	0	0	0	0
2017/6/25~2017/7/01	0	0	1	0	0	0	0

Fonte: Wen et al. (2018)

- ***Route Patterns*:** Wen et al. (2018) inferem que existem cenários em que algumas compras podem ser influenciadas por outras, nessa situação uma matriz *Markov Chain* é construída para calcular as probabilidades das atividades recentes.
- ***Monthly Budget*:** Esse *feature* corresponde à quantidade limitada de dinheiro a cada mês, ou seja, o "valor fixo de orçamento". Ele mede o nível de consumo em cinco níveis que são associados ao valor de compra.

Na fase de experimentação, foi utilizado o *dataset* de transações bancárias de uma entidade taiwanesa. A Tabela 3.2 apresenta objetivos da fase experimental proposta por Wen et al. (2018). Também são apresentados os métodos empregados, assim como as métricas utilizadas durante a avaliação.

Tabela 3.2: Objetivos da fase experimental no STPC-PGM

Objetivo	**Métodos**	**Métricas**
Detecção de periodicidade	-TOP *-Matrix Factorization* *-Markov Chain* *-Random Forest* -W4 -STC -STPC	-Acurácia *-Recall* *-F1 score*
Previsão do pagamento	*-Decision Tree* *-Random Forest* *-Regressão Lineal* *-Super Vector Machine*	*Macro Error Distance*

3.1.5 Comparação

Na Tabela 3.3, são apresentados os trabalhos coletados classificados em três categorias: (1) Modelos probabilísticos, (2) Modelos de ML e (3) Modelos de DL. Observa-se que os trabalhos possuem diferentes objetivos focados na análise preditiva, neste caso o foco da primeira proposta é a previsão da próxima transação de compra. Assim também, observa-se que os três tipos de modelos são promissores na análise preditiva no *E-commerce*, porém os modelos probabilísticos dependem diretamente da quantidade de dados por causa da atualização nos cálculos das probabilidades. Enquanto isso, os modelos de ML e DL precisam empregar um *dataset* completo que permita treinar corretamente o modelo; eles são mais complexos, mas isso é compensado pela qualidade dos resultados. Além disso, algumas propostas (WEN et al., 2018; HUANG et al., 2019; MARTÍNEZ et al., 2020; SARKAR; BRUYN, 2021) consideram a fase de *feature-engineering* importante para extrair ainda mas informação dos *features*-base e aprimorar a análise.

No caso dos *datasets* empregados, a maioria de propostas usam *datasets* disponibilizados de forma confidencial, descartando a possibilidade de utilizá-los para uma análise adicional. Nesse cenário, apesar do acesso aos *datasets* ser limitado, o fato de proteger dados pessoais permite escolher um *dataset* público e mesmo assim seguir em frente com a proposta. Em relação ao *dataset*, algumas propostas (AMPHAWAN; LENCA; SURARERKS, 2011; TATTI; MOERCHEN; CALDERS, 2014) fazem uso do *Online Retail Dataset*, um conjunto de dados que não expõe informações privadas dos usuários, o qual poderia ser empregado nessa proposta. Além disso, PRED (YUAN et al., 2017) e STPC-PGM (WEN et al., 2018) consideraram-se como abordagens mais simples para conseguir prever a próxima transação de compra, por isso, eles foram selecionados como os *baselines*. Dessa forma, optou-se por implementar os *baselines* baseados na informação fornecida nos artigos, como também *Online Retail Dataset* foi empregado.

Como resultado, esta primeira proposta visa implementar trés arquiteturas baseadas em redes recurrentes para prever a próxima transação de compra usando informações não privadas dos clientes. Por exemplo, as coordenadas geográficas para identificar a localização da compra, é substituída pelo nome da região onde se fez a compra. Desse modo, procura-se avaliar o impacto de utilizar esse tipo de informações nos modelos abordados.

Tabela 3.3: Resumo das propostas focadas na análise preditiva no *E-commerce*

Categoria	**Modelo**	**Objetivo**	***Dataset***
Modelos Probabilísticos	Cadeias de Markov (RENDLE STEFFEN; SCHMIDT-THIEME, 2010)	Prever o item de compra	*E-commerce website*
	PRED (YUAN et al., 2017)	Prever o próximo local de compra	Sintético
	STPC-PGM (WEN et al., 2018)	Prever o local, categoria e pagamento da próxima compra.	Transações de um banco taiwanês (não disponibilizado)
	Shopper (RUIZ; ATHEY; BLEI, 2020)	Prever o próximo item de compra	*Grocery store*
	PRIMA++ (LI; CHEN; ZHAO, 2021)	Recomendar um item	*EBay sellers*
Modelos de DL	TR-CT (AMPHAWAN; LENCA; SURARERKS, 2011)	Regras de associação	*Online Retail Dataset*
	Amostragem de itens (TATTI; MOERCHEN; CALDERS, 2014)	Identificação de padrões	*Online Retail Dataset*
	Logistic Lasso e *Gradient Tree Bosting* (MARTÍNEZ et al., 2020)	Prever o próximo item de compra	Transações de uma empresa de atacado (não disponibilizado)
	K-means (TABIANAN; VELU; RAVI, 2022)	Prever se o cliente faz ou não a compra	*Malaysia's E-commerce dataset*
	Modelos de classificação (SAFARA, 2022)	Classificar usuários	*Online shopping**
	XGBoost (WANG et al., 2023)	Prever se o usuário fará uma compra	*Dataset* de contas de usuário
	Random forest e Redes Neurais Profundas (ŁADYŻYŃSKI; ŻBIKOWSKI; GAWRYSIAK, 2019)	Prever produtos em crédito	Transações bancárias
	RNN (HUANG et al., 2019)	Segmentação de clientes	*Online shopping**
	LSTM e GRU (ZHOU et al., 2018)	Segmentação de comportamentos	*Online shopping**
	LSTM (SARKAR; BRUYN, 2021)	Extrator de *features*	-
	EE-CNN (ZHU et al., 2020)	Prever o próximo item de compra	Transações de uma empresa de varejo (não disponibilizado)

**Online shopping dataset* não é o mesmo, os conjuntos de dados pertencem a diferentes lojas

3.2 Avaliação de SR Baseados em Grafo de Conhecimento

Esta seção descreve *surveys* focados de sistemas de recomendação baseados no grafo de conhecimento. A seguir, são descritos cada um deles.

Zhang, Chen et al. (2020) fornece uma revisão de aquelas propostas de sistemas de recomendações explicáveis, dessa maneira propõe-se uma taxonomia para classificar recomendações explicáveis existentes: a fonte de informação das explicações, e o mecanismo algorítmico para gerar recomendações explicáveis. Além disso, é resumido as diferentes tarefas de recomendação, como recomendação do produto, recomendação social e recomendação de POI (Pontos de Interesse). Enquanto isso, Guo et al. (2020) faz uma revisão sistemática daquelas propostas de sistemas de recomendação baseadas em grafos de conhecimento em três categorias: métodos baseados em *embeddings*, métodos baseados em conexão e métodos baseados em propagação. Por outro lado, Chicaiza and Valdiviezo-Diaz (2021) explora aquelas propostas recentes de métodos de filtragem para um sistema de recomendação baseados em grafos de conhecimento.

Wang et al. (2022) realiza uma revisão abrangente do desenvolvimento recente em métodos e técnicas de *embeddings* de grafos heterogêneos. Desse modo, esses métodos são categorizados sistematicamente com base nas informações que eles usaram no processo de aprendizagem para enfrentar os desafios colocados pela heterogeneidade. Além disso, são apresentados vários sistemas amplamente implantados que demonstraram o sucesso das técnicas de *embeddings* na resolução de problemas de aplicação do mundo real. Numa outra abordagem, Wu et al. (2022) fornece uma revisão de pesquisas recentes sobre sistemas de recomendação baseados em GNN. Especificamente, é apresentada uma taxonomia de modelos de recomendação baseados em *Graph Neural Network* (GNN) de acordo com os tipos de informação utilizados e tarefas de recomendação.

Li, Qu and Wang (2023) também revisa estudos recentes, discutindo o estado atual e as aplicações práticas do conhecimento sistemas de recomendação baseados em grafos, além disso são resumidos os pontos fortes e pontos fracos desses métodos observando se eles melhoram significativamente o desempenho em áreas como precisão, diversidade, interpretabilidade e novidade. De modo similar, Zhang et al. (2024) fornece uma revisão sistemática de sistemas de recomendação baseadas em *Knowledge Graph Embedding* (KGE) em termos de métodos e aplicações. Da mesma forma, uma série de cenários são resumidos juntamente com as informações e as estatísticas sobre *datasets* relacionados.

3.2.1 Comparação

Na Tabela 3.7, é apresentada uma recopilação dos trabalhos focados em SR baseados no grafo de conhecimento em três categorias: (1) baseados em *embeddings*, (2) baseados em conexões e (3) baseados em propagação. As propostas baseadas em *embeddings* e baseadas em conexões, são de anos passados porque as propostas mais recentes são focadas em propagação. Mesmo assim, decidiu-se incluir essas duas classificações para melhor compreensão da evolução dos *frameworks* que utilizam o grafo de conhecimento com a intenção de aprimorar as recomendações. Embora os modelos baseados em *embeddings* sejam os mais simples de implementar, eles poderiam deixar de fora caminhos de alta ordem no grafo. Os modelos baseados em conexões devem definir os meta-caminhos, o que pode ser entediante e, no caso dos modelos baseados em propagação, a agregação e a atualização precisam ser minuciosa, o que aumenta a complexidade.

Por outro lado, os modelos baseados em *embeddings* conseguem ser aplicáveis em múltiplos cenários devido ao conhecimento externo que podem ser achados em conjunto de dados externos, como *Freebase* ou *Satori*. Os modelos baseados em meta-estruturas ou meta-caminhos são estabelecidos para um cenário específico e não podem ser generalizados. Além disso, os modelos baseados em *path-embedding* e modelos de propagação não são adequados para *datasets* extensos, porque a complexidade pode crescer muito na enumeração de caminhos e vizinhos. Mesmo assim, o tamanho do *dataset* pode ter impacto na qualidade e na quantidade de caminhos, e no pior dos casos poderia-se não fornecer caminhos suficientes para minar relações. Com o desenvolvimento das técnicas GNN, o método baseado em propagação tornou-se uma nova tendência porque permite a exploração quase completa do grafo por meio de diversas técnicas.

Na Tabela 3.7, são incluídas as informações sobre o tipo de grafo empregado, bem como os métodos KGE. O foco foi selecionar aqueles *frameworks* que conseguiram obter recomendações explicáveis utilizando o grafo de conhecimento a fim de armazenar as informações sobre os usuários, elementos e suas interações. Desse modo, foram selecionados cinco*frameworks* que foram avaliados de duas formas: (1) na Tabela 3.4, compara-se os *frameworks* (ECFKG, DSKE, KGAT) que usam o *grafo de user-item*; e (2) na Tabela 3.5, compara-se os *frameworks* baseados em propagação (*RippleNet*, KGCN, KGAT).

Tabela 3.4: *Frameworks* que usam o *grafo de user-item*

Tipo de grafo	***Framework***	**Método**	**Método KGE**
Grafo de User-item	ECFKG	*Embedding*	TransE
	DSKE	Conexão	*end-to-end*
	KGAT	Propagação	TransR

Tabela 3.5: *Frameworks* baseados em propagação.

Modelo	***Framework***	**Tipo de grafo**	**Método KGE**
Propagação	*RippleNet*	*Grafo de Item*	*end-to-end*
	KGCN	*Grafo de Item*	*end-to-end*
	KGAT	*Grafo de User-item*	TransR

As métricas dos sistemas de recomendação são utilizadas de acordo com o que precisa ser avaliado (GAO et al., 2023). A Tabela 3.6 resume as métricas que foram utilizadas para cada *framework*. A seguir, uma breve descrição das métricas:

- **Precisão e AUC:** Na previsão da *click-through rate* (CTR), as estruturas aplicam o modelo treinado no conjunto de teste e geram a probabilidade de clique prevista. A precisão e a AUC avaliam o desempenho da previsão de CTR.
- **Precision@K, Recall@K, F1@K e NDGC**[1]**:** Na recomendação top-K, as estruturas usam o modelo treinado para selecionar K itens com a maior probabilidade de clique prevista para cada usuário no conjunto de teste.
- **Hit:** Refere-se à porcentagem de usuários que possuem, pelo menos, uma recomendação correta obtida pelo sistema de recomendação.

Tabela 3.6: Métricas utilizadas em cada *framework*

	Acurácia	AUC	F1	NDGC	*Precision@K*	Hit	*Recall@K*
RippleNet	X	X					X
KGCN		X	X				X
DSKE				X			X
KGAT				X			X
ECFKG				X	X	X	X

[1]Ganho cumulativo com desconto normalizado

Como resultado, esta segunda proposta visa comparar cinco *frameworks* baseados em grafo de conhecimento. Os *frameworks*foram selecionados por empregarem o grafo de *user-item* (ECFKG, DSKE e KGAT) ou por serem *frameworks* baseados em propagação (*RippleNet*, KGCN, KGAT). Desse modo, procurou-se avaliar o tipo de grafo empregado, bem como o tipo de método mais abordado em propostas recentes. Cada um deles é avaliado por meio do *dataset* extenso de uma plataforma de *streaming* no Brasil, e assim observar seu desempenho ao utilizar dados reais.

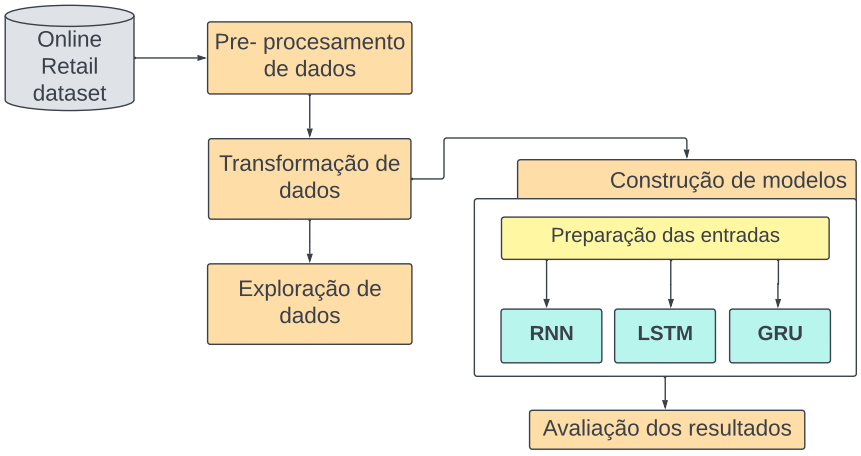
Tabela 3.7: Resumo dos *frameworks* baseados em SR Explicáveis baseados em grafo de conhecimento

Modelo	***Framework***	**Tipo de KG**	**Método KGE**	***Dataset***
Baseado em *embedding*	DKN (WANG et al., 2018b)	*Grafo de Item*	TransD	*MovieLens-1M*
	KSR (HUANG et al., 2018)	*Grafo de Item*	TransE	*MovieLens-1M, MovieLens-20M, Amazon-Book, Last-FM*
	CFKG (ZHANG et al., 2018)	*Grafo de User-Item*	TransE	*Amazon Review*
	ECFKG (AI et al., 2018)	*Grafo de User-Item*	TransE	*Amazon Review*
	MKR (WANG et al., 2019a)	*Grafo de Item*	*end-to-end*	*MovieLens-1M, Book-Crossing, Last-FM, Bing-News*
	KTUP (CAO et al., 2019)	*Grafo de Item*	TransH	*MovieLens-1M, DBbook2014*
Baseado em conexões	HeteRec (YU et al., 2013)	*Grafo de User-Item*	-	*MovieLens-100k, Yelp*
	HeteRec-p (YU et al., 2014)	*Grafo de User-Item*	-	*MovieLens-100k, Yelp*
	RuleRec (MA et al., 2019)	*Grafo de Item*	*end-to-end*	*Amazon Review*
	PGPR (XIAN et al., 2019)	*Grafo de User-Item*	*end-to-end*	*Amazon Review*
	DSKE (ZHANG et al., 2020)	*Grafo de User-Item*	*end-to-end*	*MovieLens-1M, Pinterest, Yelp, Douban*
Baseado em propagação	*RippleNet* (WANG et al., 2018a)	*Grafo de Item*	*end-to-end*	*MovieLens-1M, BookCrossing, BingNews*
	KGCN (WANG et al., 2019b)	*Grafo de Item*	*end-to-end*	*MovieLens-20M, BookCrossing, LastFM*
	KGAT (WANG et al., 2019)	*Grafo de User-Item*	TransR	*AmazonBook, Yelp, LastFM*
	KGIN (WANG et al., 2021)	*Grafo de Item*	-	*AmazonBook, LastFM*
	HAGERec (YANG; DONG, 2020)	*Grafo de User-Item*	*end-to-end*	*MovieLens-1M, MovieLens-20M, BookCrossing, BingNews*
	GACF (ELAHI; HALIM, 2022)	*Grafo de User-Item*	*end-to-end*	*MovieLens-10M, BookCrossing, BingNews*

4 MODELO PARA PREVER O COMPORTAMENTO DE COMPRA

O primeiro foco da proposta foi baseado na previsão da próxima transação de compra. Este Capítulo apresenta as etapas a serem desenvolvidas para construir três modelos baseados em arquitetura RNN: (1) modelo baseado em uma arquitetura RNN simples; (2) modelo baseado em uma arquitetura LSTM; e (3) modelo baseado em uma arquitetura GRU. O *dataset* empregado é *Online Retail Dataset* e a metodologia apresentada na Figura 4.1 contém as etapas fundamentais de ciência de dados: a coleta de dados, o pré-processamento de dados, a transformação de dados, a exploração de dados, a construção do modelo e a avaliação dos modelos. Na seção 6.1, são apresentados os experimentos para avaliar os três modelos propostos, de acordo com a configuração da rede.

Figura 4.1: Metodologia para prever a próxima transação



Fonte: Elaborado pelos autores

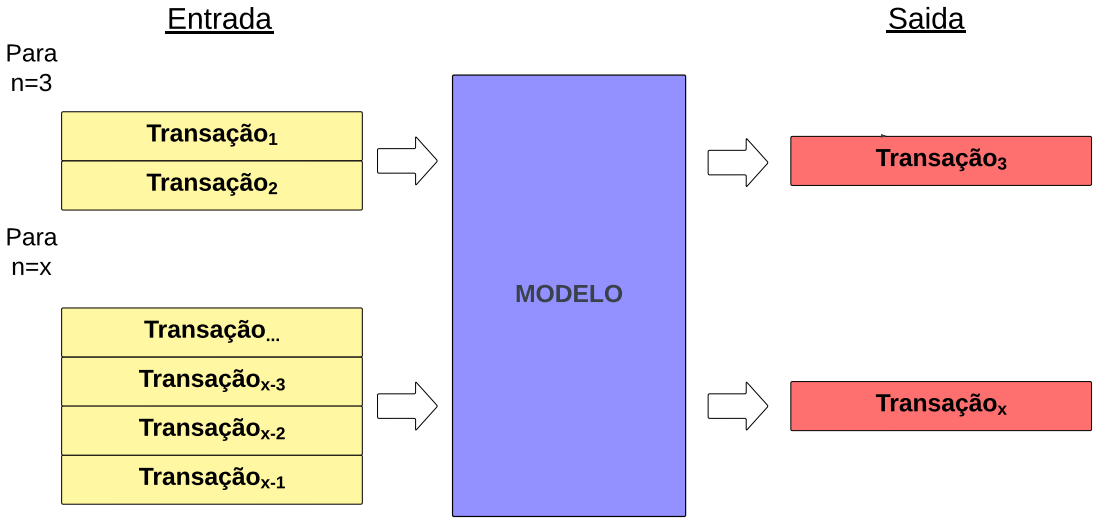
4.1 Materiais e Métodos

Online Retail Dataset - Na subseção 3.1.5, foi exposto o fato de lidar com *datasets* não públicos para a análise preditiva no *E-commerce*, pois as empresas não podem compartilhar informações sensíveis de seus clientes. Desse modo, decidiu-se utilizar um *dataset* público que contém todas as transações ocorridas entre 12/01/2010 e 12/09/2011 para uma loja transnacional *online* localizada no Reino Unido, em que a loja oferece presentes exclusivos para todas as ocasiões. *Online Retail Dataset*[1] (CHEN; SAIN; GUO, 2012) contém 541.909 instâncias e 8 atributos.

[1]https://archive.ics.uci.edu/ml/datasets/online+retail

- **Anotação de Dados** - Para empregar arquiteturas do tipo RNN, os dados precisaram ser modelados de forma sequencial, ou seja, o alvo de uma transação foi a próxima transação registrada no histórico de compras. Da mesma forma, para um histórico de tamanho n ($n > 2$), o alvo será a *transação*$_n$ após da última *transação*$_{n-1}$ desse histórico. Na Figura 4.2, é apresentada a anotação dos dados para um histórico de tamanho $n = 3$ e $n = x$.

Figura 4.2: Entrada é saída para um histórico de tamanho $n = 3$ e $n = x$



Fonte: Elaborado pelos autores

- **Dicionário de Dados** - Os itens da Tabela 4.1 apresentam a descrição dos atributos:

Tabela 4.1: Descrição dos atributos de *Online Retail Dataset*

Atributo	**Descrição**
InvoiceNo	Corresponde ao ID atribuído exclusivamente a cada transação.
StockCode	Corresponde ao código do produto.
Description	Corresponde ao nome do produto.
Quantity	Corresponde à quantidade de cada produto por transação.
InvoiceDate	Corresponde à data e hora de cada transação.
Unit Price	Corresponde ao preço unitário de cada produto.
CustomerID	Corresponde ao ID atribuído exclusivamente a cada cliente.
Country	Corresponde ao nome do país em que foi registrado na transação.

4.2 Coleta de Dados

Os dados foram coletados de uma loja transnacional *online* localizada no Reino Unido, porém o foco da previsão do comportamento de compra precisa contar com o histórico dos usuários. Assim, o *dataset* agrupou-se de acordo com o tamanho do histórico por usuário. A ideia de propor essas versões, tem a ver como o tamanho dos históricos que é utilizado durante o treinamento dos modelos. Nesse sentido, é importante conhecer o impacto que tera cada uma delas em relação com as configurações de cada redes durante a fase dos experimentos. Na Tabela 4.2, são descritos cada um desses grupos.

Tabela 4.2: Agrupamento efetuado no *Online Retail Dataset*

Grupo	**Descrição**
Grupo A	Composto pelo histórico de usuários que possuem menos de 100 transações
Grupo B	Composto pelo histórico de usuários que possuem mais de 100 transações
Grupo C	Composto pelo histórico de compras de todos os clientes

4.3 Pré-processamento de Dados

O *dataset* continha algumas inconsistências que precisavam ser corrigidas antes de continuar com a próxima etapa. Dessa forma, as seguintes instâncias foram excluídas: instâncias duplicadas que possuíam exatamente a mesma informação registrada na tupla, instâncias que registravam um valor de pagamento negativo (devoluções) porque o foco apenas foi limitado na analise das saídas da loja, e as instâncias que possuíam *missing values* no caso do atributo *CustomerID*. Por fim, o *dataset* foi reduzido de 541.909 instâncias para 392.732 instâncias.

Figura 4.3: Pré-processamento de dados

```
In [18]: #Preprocesamento de dados
         df_v1 = df_online.dropna()
         df_dup = df_v1.drop_duplicates()
         df_dup.drop(df_dup[(df_dup['Quantity'] < 0)].index, inplace=True)
```

Fonte: Elaborado pelos autores

4.4 Transformação de Dados

Nesta etapa se estabeleceram seis *features* com base nos atributos originais, a Tabela 4.3 contém a descrição de cada um deles.

Tabela 4.3: Descrição dos *features* conseguidos após a transformação dos dados

Atributo	**Descrição**
CustomerID	Mantém a identificação (ID) do cliente.
Tipo de dia	Determina quando as compras foram feitas; se foram feitas entre a segunda-feira e sexta-feira, são considerados nos dias da semana (0), e se foi entre sábado e domingo, são considerados no fim de semana (1).
Região	Mantém o nome do país em que a compra está registrada.
Categoria	Corresponde aos códigos da categoria de compra. Para agrupá-los, foi necessário manter as quatro primeiras strings do código de *StockCode*.
Monthly Budget	Corresponde ao nível de consumo por mês de acordo com o pagamento.
Pagamento	Calculado multiplicando o preço unitário pela quantidade de cada produto.

Monthly Budget: Wen et al. (2018) propõem o cálculo de orçamento mensal; para isso os usuários são divididos em cinco grupos de acordo com o nível de atividade de consumo. Definiu-se o pagamento com nível k, em que cada usuário pertence a uma faixa de pagamento diferente. Na Equação 4.1, é definido o cálculo desse *feature*.

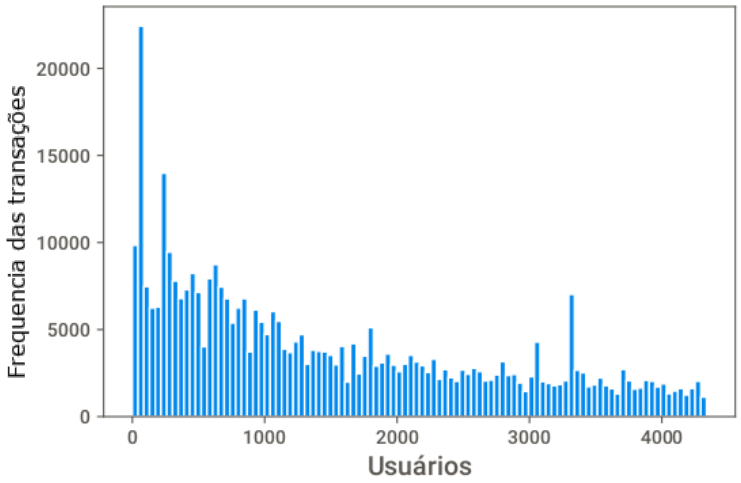
$$\begin{aligned} PaymentLevel(d) &= \lfloor \frac{P_d}{PaymentRange^u} \rfloor, where \\ PaymentRange^u &= \lfloor \frac{\sum_{\{y,m\}} \sum_j p_j | t_{j \cdot year=y \&\& j \cdot month=m}}{|\{y,m\}| \cdot k} \rfloor \end{aligned} \tag{4.1}$$

4.5 Exploração de Dados

Nesta etapa, os dados foram explorados e analisados para descobrir informações implícitas associadas ao histórico de compras, assim foi possível identificar alguns pontos fracos que podem prejudicar a análise. A seguir, é apresentado um resumo dos aspectos mais relevantes que foram identificados durante a exploração de dados.

- O conjunto de dados contém 4.339 clientes. Na Figura 4.4, observa-se que as transações por usuário estão agrupadas em $bins = 100$, além disso a maioria dos usuários possuíam menos de 100 transações registradas. Também foi identificado que o histórico de maior tamanho é de 7.676 registros. Logo, essa desproporção do tamanho dos históricos motivou a dividir o *dataset* em três grupos.

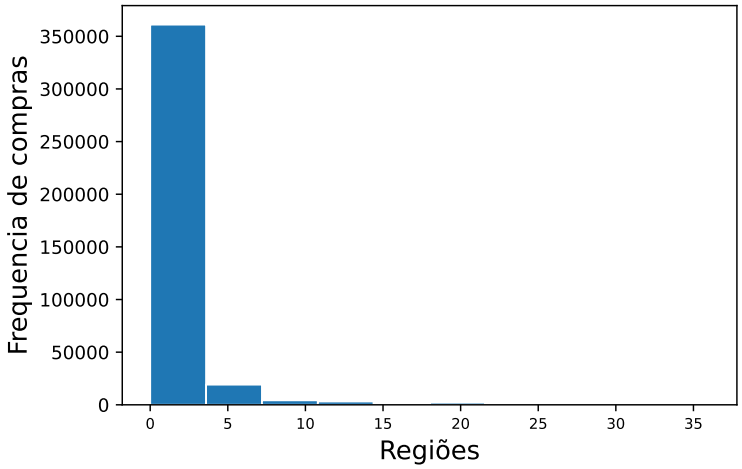
Figura 4.4: Histograma do *feature CustomerID*



Fonte: Elaborado pelos autores

- O conjunto de dados contém 37 regiões em que foram efetuadas as compras. Na Figura 4.5, observa-se que a distribuição concentra-se apenas em cinco regiões. Esse aspecto poderia prejudicar ao tentar identificar o padrão de mobilidade do cliente porque a maioria apenas compram em uma região.

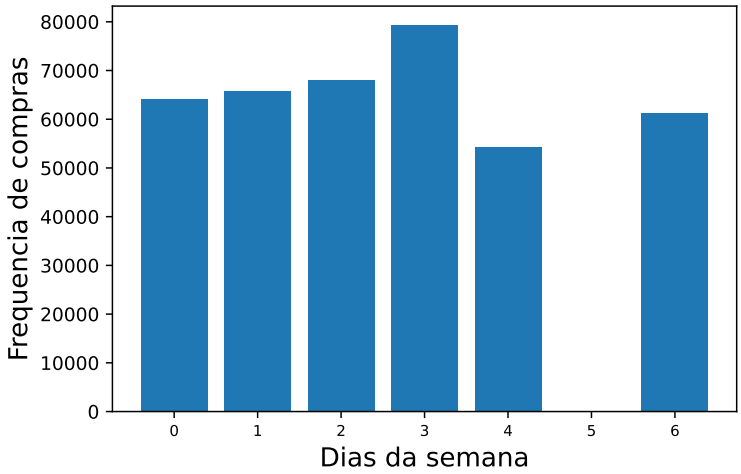
Figura 4.5: Histograma do *feature Country*



Fonte: Elaborado pelos autores

- A Figura 4.6 mostra a distribuição das compras durante os dias da semana, em que segunda-feira é 0, terça-feira é 1, quarta-feira é 2 etc. Observa-se que não foram registradas compras no dia sábado. Além, existe uma desproporção entre os *weekdays* (de segunda a sexta-feira) e *weekends* (sábados e domingos).

Figura 4.6: Histograma do *feature* Tipo do dia



Fonte: Elaborado pelos autores

4.6 Construção de Modelos

4.6.1 Proposta

Esta proposta foi baseada na avaliação da arquitetura RNN, LSTM e GRU para a previsão da próxima transação por meio da análise do histórico de transações registradas. Apresenta-se uma solução para descobrir a próxima transação usando métodos sequenciais. Dado Tu um conjunto de transações do usuário u, cada transação contém $t_i = \{id_i, t_i, c_i, r_i, mb_i, p_i\}$: *Customer_ID* ($id$), tipo de dia ($t$), região ($r$), categoria do produto (c), *monthly budget* (mb) e pagamento (p). Tendo um conjunto Tu_m de tamanho m em que $m > 2$, para isso cada Tu deve ser dividido em subconjuntos de tamanho k que são denominados "bloco", o que servirá para prever a *transação*$_{k+1}$. A seguir, são apresentados dois pontos importantes levados em consideração nessa proposta:

- Baseado nos seis *features* da Tabela 4.3 foram identificadas relações entre os *features* para prever a próxima transação no histórico; por exemplo, identificar a região mais visitada de acordo com o dia da semana ou identificar qual produto é possível comprar de acordo como o *monthly budget*. Por meio desses *features* foi possível coletar informações e identificar algumas relações que geralmente são comuns, assim como aquelas ações que não cumprem um padrão que passam despercebidas.
- A proposta utilizou o parâmetro k, que é o tamanho dos blocos dos quais as informações devem ser extraídas, o que permite treinar o modelo com históricos de diversos tamanhos. Por meio desse parâmetro, esperou-se construir um modelo mais especializado em termos de reconhecimento de padrões.

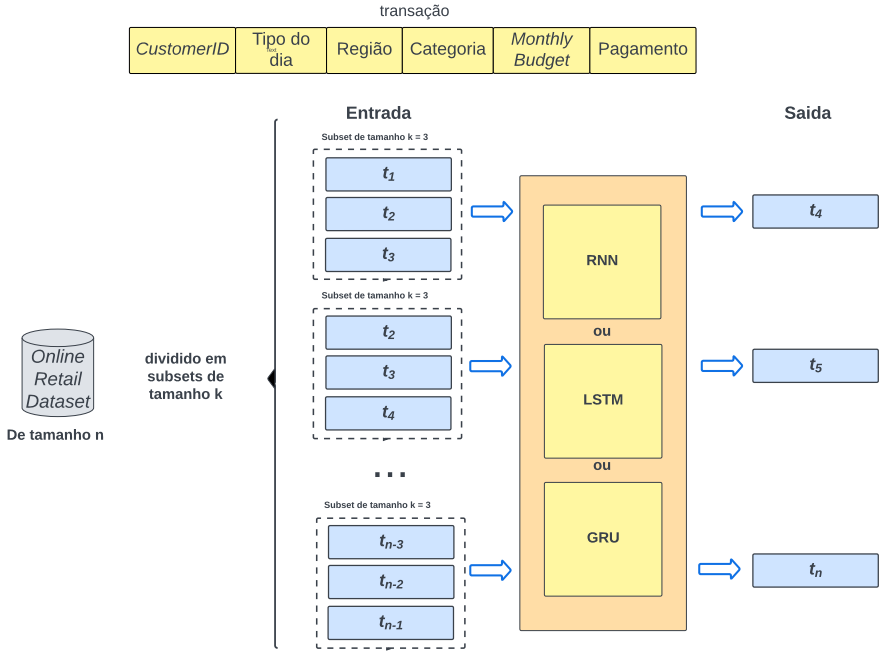
Em geral, o comportamento de compra costuma ser variável e dinâmico, isso significa que uma análise superficial não é suficiente. Nesse sentido, poderia-se utilizar um modelo sequencial para modelar as transações de um histórico em blocos de tamanho k. Assim, a capacidade de memória das redes recorrentes pode "lembrar" relações pouco frequentes nos dados e conseguir uma melhor previsão da próxima transação.

4.6.2 Preparação de Entradas

Para o trabalho com redes neurais, é necessário codificar os atributos não numéricos. Existem diferentes formas de codificar aquelas variáveis que são categóricas, as mais conhecidas são: **(a) *Ordinal Encoding***, em que cada instância é atribuída a um número inteiro, e **(b) *One Hot Encoding***, em que cada instância é atribuída a um vetor binário. Essa proposta usou *Ordinal Encoding* pelo fato de ser mais simples de executar.

Diante isso, as entradas e saídas foram modeladas de um jeito diferente. O objetivo do modelo foi prever a próxima transação em cada bloco de tamanho k ($k > 2$), ou seja, o alvo de um bloco de tamanho k será a transação $k + 1$, como é explicado na Figura 4.7. O modelo precisou ser treinado com todos os blocos de tamanho k que podem ser montados a partir de um histórico de tamanho n, dessa forma, o modelo é alimentado com as múltiplas sequencias do histórico.

Figura 4.7: Treinamento do modelo



Fonte: Elaborado pelos autores

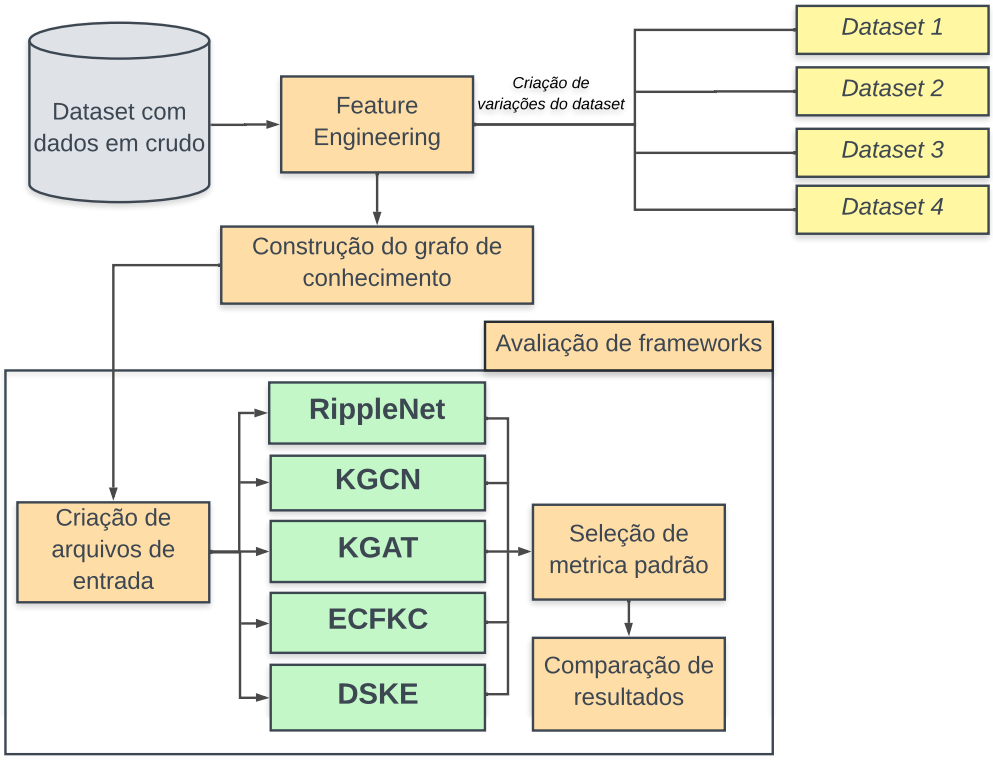
4.6.3 Implementação de Modelos

Depois de preparar as entradas, as arquiteturas baseadas em redes recorrentes, foram implementadas usando o pacote *tensorflow*. De acordo com a modelagem anterior, para a RNN foi usado *BasicRNNCell*, para LSTM foi usado *BasicLSTMCell* e para GRU foi usado *GRUCell*. Na seção 6.1, é descrita a configuração de cada uma das redes.

5 METODOLOGIA PARA AVALIAR *FRAMEWORKS* DE RECOMENDAÇÕES EXPLICÁVEIS QUE USAM GRAFO DE CONHECIMENTO

O segundo foco da proposta foi fundamentado na avaliação de *frameworks* baseados em grafo de conhecimento, empregando *Streaming Platform Dataset*. Este Capítulo apresenta a metodologia desenvolvida para avaliar cinco *frameworks*: *RippleNet*, KGCN, KGAT, ECFKG e DSKE. A Figura 5.1 contém as etapas da metodologia empregada: *feature engineering*, construção do grafo de conhecimento, criação de arquivos de entrada e avaliação de *frameworks*. Na seção 6.2, são apresentados os experimentos efetuados.

Figura 5.1: Metodologia para avaliar *frameworks* baseados em grafo de conhecimento



Fonte: Elaborado pelos autores

5.1 Materiais e Métodos

Streaming Platform Dataset: Esse *dataset* foi disponibilizado de forma privada por uma empresa de serviços de *streaming* no Brasil. Os dados são registros de diferentes usuários que assistiram a itens de entretenimento, por exemplo, filmes, séries ou programas de TV. O *dataset* contém 200 milhões de registros e 17 atributos.

- **Anotação de Dados** - A anotação dos dados precisou ser executada de forma diferente em cada *framework*, uma vez que cada um deles considerou um tratamento diferente dos dados. Na seção 5.5 as informações são descritas com maior detalhe.
- **Dicionário de Dados** - A versão original de *Streaming Platform Dataset* contém 17 atributos, mas, na Tabela 5.1, são apresentadas as informações de 10 *features*.

Tabela 5.1: Descrição dos atributos do *Streaming Platform Dataset*

Atributo	**Descrição**
Title_ID	Contém o identificador de cada filme, série ou programa.
Media_ID	Contém o identificador relacionado ao Title_ID, que contém os episódios ou temporadas.
Duration	Contém a duração estabelecida para cada filme, série ou programa.
Played	Contém o tempo de reprodução de cada filme, série ou programa.
Paused	Contém o tempo de reprodução até quando o usuário decidiu fazer uma pausa.
PlayerType	Contém o dispositivo no qual foi assistido o filme, série ou programa.
DeviceGroup	Contém os tipos de categorização para os dispositivos de reprodução considerados no *PlayerType*.
Category	Contém a categoria do filme, série ou programa.
Genres	Contém o gênero do filme, série ou programa.
Timestamp	Contém a data exata em que foi assistido o filme, série ou programa.

5.2 Coleta de Dados

Os dados coletados são históricos de itens assistidos em uma plataforma *streaming*. Optou-se por utilizar apenas 2 milhões de dados, devido ao longo volume de dados e à complexidade do tempo, pois ao trabalhar com 200 milhões de dados, o tempo de treinamento levaria dias ou até semanas. Além disso, a qualidade e a quantidade de dados no histórico deviam de ser consideradas na análise, por isso são estabelecidos dois grupos de dados: no (1) *dataset* sem agrupamento, os primeiros 2 milhões de linhas foram selecionados sem nenhum critério de ordenação e as interações dos usuários ficaram dispersas; e no (2) *dataset* com agrupamento, os dados foram ordenados pelo *user_id* e depois foram selecionados os dados dos primeiros 36.397 usuários, o que permitiu o acesso ao histórico de interações de cada um desses usuários. O termo de "densidade" refere-se a divisão do número de interações pelo número de usuários, para uma melhor interpretação dos dados.

Tabela 5.2: Estatísticas do *dataset* sem agrupamento e com agrupamento

	***Dataset* sem agrupamento**	***Dataset* com agrupamento**
# usuarios	1.141.995	36.397
# itens	32.117	30.608
# iterações	2.195.785	2.000.000
Densidade (#iterações / #usuário)	1,92	54,94

5.3 *Feature Engineering*

Na exploração dos trabalhos relacionados (seção 3.2), observou-se que o *feature Rating* pode ser necessário para melhorar o processo de recomendação. Esse *feature* foi definido como a qualificação de um elemento dado pelo usuário. Ademais, o *Streaming Platform Dataset* não possuía esse atributo, por isso foram propostas diferentes formas de gerar esse atributo. Na Tabela 5.3 são descritas as quatro versões do *dataset* original, cada versão contém um jeito diferente de gerar o *feature Rating*.

Tabela 5.3: Descrição das versões do *Streaming Platform Dataset*

	***Dataset* sem agrupamento**	***Dataset* com agrupamento**	**Tipo de função (*Rating*)**
Dataset v1	X		Números inteiros aleatórios
Dataset v2	X		Números inteiros com distribuição uniforme
Dataset v3	X		*Rating* Implícito
Dataset v4		X	*Rating* Implícito

- *Dataset* v1: Utilizou o *Streaming Platform Dataset* sem agrupamento (descrito na subseção 5.2). Ademais, os valores do *feature Rating* foram gerados por meio da função *random.Int* para obter números aleatórios no intervalo de 0 a 4.
- *Dataset* v2: Utilizou o *Streaming Platform Dataset* sem agrupamento, além disso, os valores do *feature Rating* foram gerados por meio da função *random.Uniform* para obter numeros inteiros com uma distribuição uniforme no intervalo de 0 a 4.
- *Dataset* v3: Também utilizou o *Streaming Platform Dataset* sem agrupamento. Para o cálculo do *feature Rating*, a ideia principal foi extrair essa informação com base no tempo que o usuário passou assistindo à mídia. Nessa perspectiva, os valores do *feature Rating* foram calculados implicitamente baseados nos *features Duration* e *Play*. O *feature Play* representa o tempo de mídia assistida pelo usuário

naquela sessão, enquanto o campo *feature "duration"* representa a duração total da mídia, ambos representados em milisegundos. Então, ambos os *features* foram divididos, depois é feita a normalização para que os valores fiquem no intervalo entre 0 e 1. Esse método foi denominado ***Rating* Implícito**. Na Tabela 5.4, é descrito o mapeamento final para obter valores no intervalo de 0 a 4.

Tabela 5.4: Mapeamento do *feature Rating*

Valor normalizado	***Rating***
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

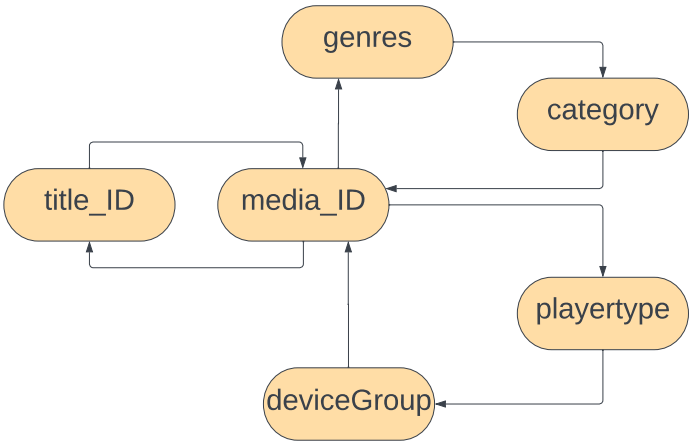
- *Dataset* v4: Utiliza o *Streaming Platform Dataset* com agrupamento e os valores do ***Rating* Implícito**.

A ideia de propor as versões da Tabela 5.3 surgiu para avaliar o desempenho de trabalhar com históricos completos ou não, bem como utilizar funções de *Rating* que são calculados a partir de outros *features* (*Rating* Implícito).

5.4 Construção do Grafo de Conhecimento

Nesta etapa, as informações são estruturadas no grafo com base nos atributos mais importantes do conjunto de dados. Na Figura 5.2, observa-se o grafo de conhecimento proposto para *Streaming Platform Dataset*; o grafo contém seis atributos selecionados do *dataset*, que são: *media_ID*, *title_ID*, *category*, *genre*, *playertype* e *deviceGroup*. O atributo *user_ID* não é considerado no grafo porque a recomendação é baseada na informação de histórico de cada usuário; o tratamento desse *feature* foi revisado na seção 5.5.

Figura 5.2: Proposta do grafo de conhecimento para *Streaming Platform Dataset*



Fonte: Elaborado pelos autores

5.5 Criação dos Arquivos de Entrada

Antes de iniciar a etapa de execução, foi necessário adaptar o *dataset* à estrutura de entrada de cada *framework*. Além disso, a Tabela 5.5 apresenta os *frameworks* que compartilham a mesma estrutura nos arquivos de entradas. A seguir, são descritos os arquivos necessários para cada *framework*, de acordo com o grupo que corresponde.

Tabela 5.5: Agrupamento dos *frameworks* em relação aos arquivos de entrada

	Framework
Grupo 1	*RippleNet*, KGCN
Grupo 2	KGAT, ECFKG
Grupo 3	DSKE

- *Adaptação para o grupo 1:* *RippleNet* e KGCN são *frameworks* que utilizam a mesma estrutura porque foram desenvolvidos pelo mesmo autor. A seguir, são descritos os três arquivos essenciais para o trabalho com esses *frameworks*.

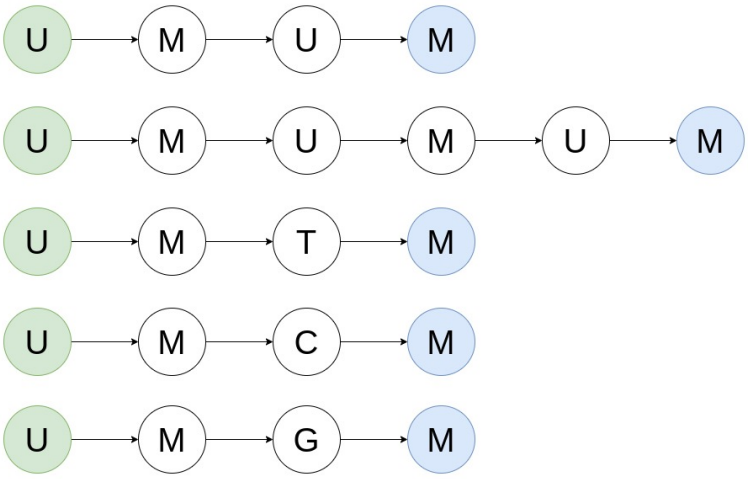
  1. ***Dataset* geral** - este arquivo possui três colunas; a primeira coluna contém o *User_ID*, a segunda contém a *Media_ID*, e a terceira contém o *Rating*.
  2. ***Item_index2entity_id_rehashed*** - este arquivo contém duas colunas; a primeira contém o *Media_ID* e a segunda contém o mapeamento desse *feature*.
  3. ***Kg_rehashed:*** - este arquivo contém as tripletas de entidade-relação estabelecidas a partir do grafo, por exemplo, <*The Office*, temporada, *Season 1*>.

- *Adaptação para o grupo 2:* KGAT e ECFKG são *frameworks* que compartilham a mesma estrutura dos arquivos de entrada. A seguir, são descritos os cinco arquivos essenciais para o trabalho com esses *frameworks*.

    1. ***User_list:*** Este arquivo possui duas colunas, a primeira contém os *user_ID* e a segunda, o mapeamento deles.
    2. ***Entity_list:*** Este arquivo possui duas colunas, a primeira contém todos os itens (exceto *user_list*), e a segunda, contém o mapeamento deles.
    3. ***Item_list:*** Este arquivo possui três colunas: (1) *org_id*, (2) *remap_id* e (3) *freebase_id*. No caso do *dataset* da plataforma de *streaming*, as colunas 1 e 2 são extraídas do *entity_list*, e *freebase_id* é gerada aleatoriamente.
    4. ***Relation_list:*** Este arquivo possui duas colunas, a primeira contém as oito relações estabelecidas no grafo, e a segunda, o mapeamento delas.
    5. ***Kg_final:*** Este arquivo possui as tripletas de entidade-relação do tipo <*head, relation,tail*> que é estabelecido no grafo de conhecimento.
    6. ***train***: Cada linha é um usuário com suas interações positivas com itens, ou seja é relacionado o *user_ID* com a lista de itens que teve interação.
    7. ***test***: Cada linha é um usuário apenas com suas interações positivas com itens, enquanto as interações não observadas são tratadas como instâncias negativas.

- *Adaptação para o grupo 3:* No caso do DSKE, os arquivos de entrada foram adaptados de forma similar ao *dataset* do Yelp. Nesse código fonte, o arquivo de entrada foi lido e cada uma das relações modeladas no grafo de conhecimento foi mapeada para um objeto do tipo *dataframe*. Além, foi estabelecido um arquivo para que cada linha contivesse dois identificadores, um da entidade e outro do *Rating*, todos separados por um espaço em branco. Cada relação foi representada por um arquivo, totalizando quatro diferentes arquivos: *user_media, media_title, media_genre, media_category*. O arquivo que relaciona usuários e mídias tinha os valores da coluna de *Ratings* lidos. Por fim, os meta-caminhos foram modelados manualmente para cada conjunto de dados, que são essenciais para que o modelo consiga oferecer boas recomendações ao usuário. Os meta-caminhos seguiram duas regras principais: (1) deve iniciar em um usuário; e (2) deve acabar em um item. A Figura 5.3 apresenta os meta-caminhos utilizados para o treinamento do DSKE.

Figura 5.3: Meta-caminhos modelados para o DSKE. U = Usuário, M = Mídia, T = Título, C = Categoria e G = Gênero.



Fonte: Elaborado pelos autores

6 EXPERIMENTOS E RESULTADOS

Este Capítulo apresenta os experimentos realizados a fim de (1) avaliar modelos do tipo RNN para prever o comportamento de compra, bem como (2) avaliar os *frameworks* de recomendações explicáveis baseados em grafo de conhecimento. A seguir, são descritos os objetivos, ferramentas e métricas propostas e, por fim, também são incluídos a metodologia e os resultados obtidos a partir de cada um dos objetivos.

6.1 Sobre a Previsão de Comportamento de Compra

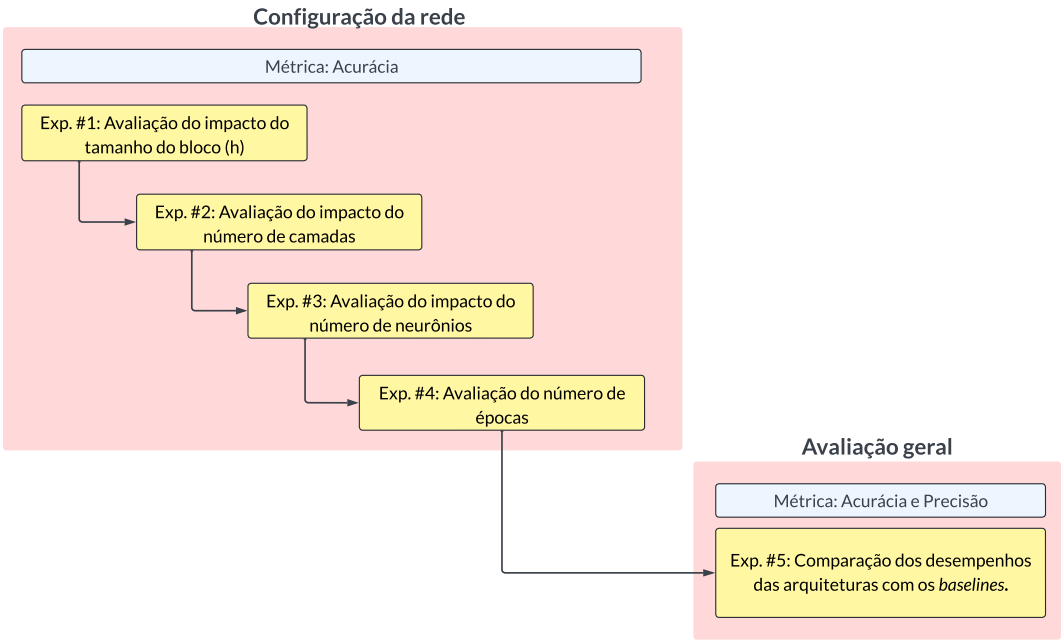
6.1.1 Objetivos

O objetivo principal é prever a próxima transação de compra. Nesse sentido, foram implementados três tipos de arquiteturas baseadas em redes neurais recorrentes, bem como a configuração dos parâmetros para melhorar os resultados. Portanto, cinco experimentos foram propostos para avaliar o desempenho das arquiteturas RNN, LSTM e GRU como se mostra na Figura 6.1. A seguir, são listados cada um dos objetivos propostos:

1. Avaliação do impacto do tamanho do bloco (h).
2. Avaliação do impacto do número de camadas.
3. Avaliação do impacto do número de neurônios.
4. Avaliação do número de épocas ou iterações.
5. Comparação dos desempenhos das arquiteturas com os *baselines*

Primeiramente, 80% dos dados foram destinados ao treinamento e 20% para o teste em todos os experimentos subsequentes. Para a configuração inicial usou-se os seguintes parâmetros: $num\_camadas = 2$, ***num_neurônios*** $= 256$, ***função_ativação*** $= ReLU$, e ***num_épocas*** $= 100$. Outros parâmetros como o $AdamOptimizer$ e $batch\_size = 250$ foram estáticos. Os primeiros três experimentos foram realizados para conseguir as configuração da rede, para isso foram utilizados o *Grupo A* e o *Grupo B*. Porém, para os experimentos 4 e 5, o *Grupo C* foi empregado para conhecer os resultados com o *dataset* completo. Além disso, a acurácia é calculada em todos os experimentos, mas a precisão apenas é considerada no último porque as duas foram estabelecidas como métricas padrão na comparação com os *baselines*. Ressalta-se que os *baselines* foram replicados de acordo com a interpretação da informação contida em cada artigo.

Figura 6.1: Visão geral dos experimentos para a avaliação dos *frameworks*



Fonte: Elaborado pelos autores

6.1.2 Ferramentas e Métricas

Os modelos propostos foram desenvolvidos no *Python* usando *Jupyter Notebook*. Também usou-se *Keras*[1] e *scikitlearn*[2]. Na Equação 6.1, observa-se como é calculada a métrica total, em que α é a acurácia de *CustomerID*, β é a acurácia do tipo de dia, δ é a acurácia da região, γ é a acurácia da categoria, ϵ é a acurácia do *monthly budget*, e λ é a acurácia do pagamento. Esse valores foram definidos de acordo ao nível de importância de cada *feature* que os autores perceberam durante a exploração de dados, nesse sentido, eles poderiam ser modificados em futuras compilações de acordo a outras observações. Por exemplo, o menor peso foi do *feature* região, pelo fato de que a maioria de clientes registravam suas compras em apenas uma região. A acurácia e a precisão foram estabelecidas como as métricas padrão na comparação com os *baselines*.

$$Total_{metric} = 0,15 \cdot \alpha + 0,20 \cdot \beta + 0,10 \cdot \delta + 0,20 \cdot \gamma + 0,20 \cdot \epsilon + 0,15 \cdot \lambda \quad (6.1)$$

[1]https://keras.io/
[2]https://scikitlearn.org/stable/

6.1.3 Avaliação do Impacto do Tamanho do Bloco

Metodologia

Este experimento foi baseado na execução das arquiteturas RNN, LSTM e GRU para *Grupo A* e *Grupo B* com a configuração de rede inicial. Nesse caso, avaliou-se o desempenho definindo dois valores diferentes do tamanho do bloco: $h = \{7.30\}$, que corresponde ao número de dias por uma semana ou um mês. A partir disso, obtém-se a acurácia para cada *feature*, além da acurácia total por meio da Equação 6.1.

Resultados

Na Tabela 6.1, observa-se que os modelos que obtiveram melhores resultados foram as arquiteturas RNN e LSTM. Além disso, um tamanho de bloco menor ($h = 7$) parece ser adequado para o conjunto de dados mais completos, como no caso do *Grupo B*, para o qual o modelo RNN (de menor complexidade) foi suficiente para a análise. Por outro lado, o *Grupo A*, que não possuía históricos completos, obteve bons resultados com $h = 7$ empregando uma arquitetura LSTM. No caso do *feature* Região, considerando $h = 7$ alcançou-se 8,8% de acurácia com o *Grupo A*; já no *Grupo B*, a acurácia obtida foi de 94,4%. Logo, a previsão dos atributos é mais equilibrada nesse último grupo.

Em relação ao tamanho do bloco, observou-se que $h = 7$ tem resultados superiores para *Grupo A* e *Grupo B* com as três arquiteturas propostas. Ademais, como foi observado na seção 4.5, o número de compras não era proporcional entre os dias de semana e os fins de semana, o que indica que um bloco de tamanho maior prejudica os resultados.

Tabela 6.1: Acurácia da avaliação do tamanho do bloco (h)

		RNN	**LSTM**	**GRU**
$h = 7$	Grupo A	60,3%	59,6%	57,3%
	Grupo B	**70,3%**	62,5%	66,5%
$h = 30$	Grupo A	57,3%	**58,4%**	42,7%
	Grupo B	50,7%	48,3%	58,0%

6.1.4 Avaliação do Impacto do Número de Camadas

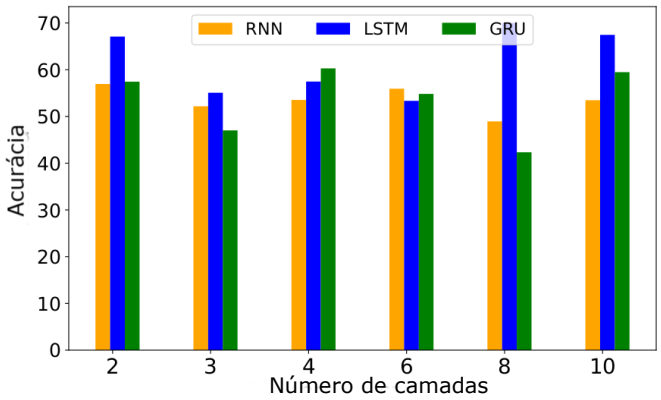
Metodologia

A partir dos resultados obtidos na subseção 6.1.3 definiu-se o valor de $h = 7$. Posteriormente, as arquiteturas RNN, LSTM e GRU foram executadas para o *Grupo A* e *Grupo B* com valores de $num\_camadas = \{2, 4, 6, 8, 10\}$. Além, obteve-se a acurácia para cada *feature*, bem como a acurácia total por meio da Equação 6.1.

Resultados

Na Figura 6.2, observa-se que as melhores acurácias foram obtidas pela arquitetura LSTM para o *Grupo A* com 8 e 10 camadas. A rede LSTM com 8 camadas atingiu quase 70% de acurácia, porém o *feature* Região obteve apenas 15,2% de acurácia. Além disso, LSTM obteve uma acurácia de 60,3% com 10 camadas e 49,3% para o atributo Região, o que indica um melhor desempenho no cálculo desse *feature*. No caso das arquiteturas RNN e GRU, não foram obtidos bons resultados em comparação ao LSTM. Em síntese, o treinamento com maior número de camadas tem um impacto positivo para lidar com o *feature* Região, embora o *dataset* possua históricos de menor tamanho (*Grupo A*).

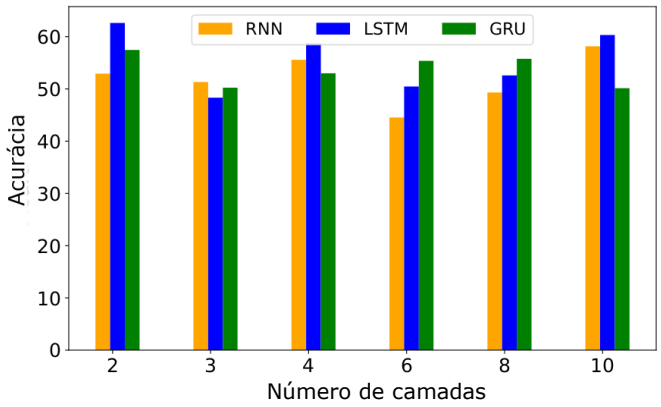
Figura 6.2: Avaliação da acurácia com diferentes valores de camadas para o Grupo A



Fonte: Elaborado pelos autores

Na Figura 6.3, observa-se que a melhor acurácia para o *Grupo B* foi obtida pela arquitetura LSTM com 62,6% com 10 camadas, porém a previsão dos *features* não foi equitativa porque o *feature* Região não conseguiu superar 15,7% de acurácia. Enquanto isso, a arquitetura GRU apenas atingiu uma acurácia de 58,8% de acurácia com duas camadas, o que não foi suficiente para superar os resultados obtidos pelo *Grupo A*.

Figura 6.3: Avaliação da acurácia com diferentes valores de camadas para o Grupo B



Fonte: Elaborado pelos autores

Em relação ao número de camadas, observa-se que um maior número de camadas consegue aprimorar os resultados, porém, uma vez mais o *feature* Região registrou um baixa acurácia pelo fato de não possuir variedade nos dados (seção 4.5). Mesmo assim, a arquitetura LSTM conseguiu aprimorar os resultados para o *Grupo A* e o *Grupo B*.

6.1.5 Avaliação do Impacto do Número de Neurônios

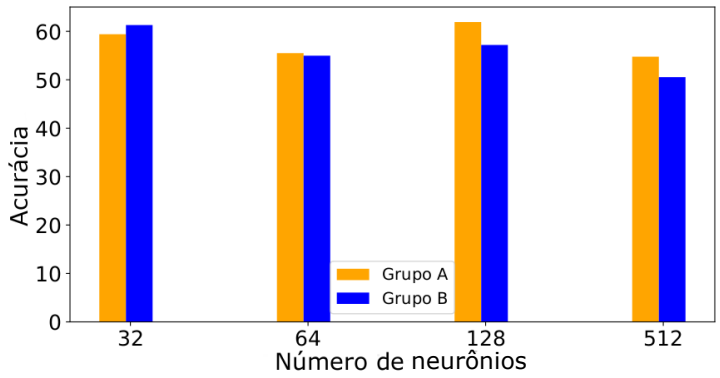
Metodologia

A partir dos resultados obtidos na subseções 6.1.3 e 6.1.4, definiu-se os valores de $h = 7$ e *num_camada = 10*. Posteriormente, apenas as arquiteturas LSTM e GRU são executadas para o *Grupo A* e o *Grupo B*, pois foram as arquiteturas que obtiveram melhores resultados nos experimentos anteriores. Em seguida, definiu-se os valores de *num_neurônios* $= \{32, 54, 128, 512\}$ e depois, obteve-se a acurácia para cada *feature*, além da acurácia total por meio da Equação 6.1.

Resultados

Na Figura 6.4, observa-se que, para o *Grupo A*, a arquitetura LSTM obteve maior acurácia com *num_neurônios* $= 128$, porém o *feature* Região apresentou uma baixa acurácia de 11%. Enquanto isso, para o *Grupo B*, a arquitetura LSTM obteve melhores resultados com *num_neurônios* $= 32$, além de acurácias balanceadas para cada *feature*.

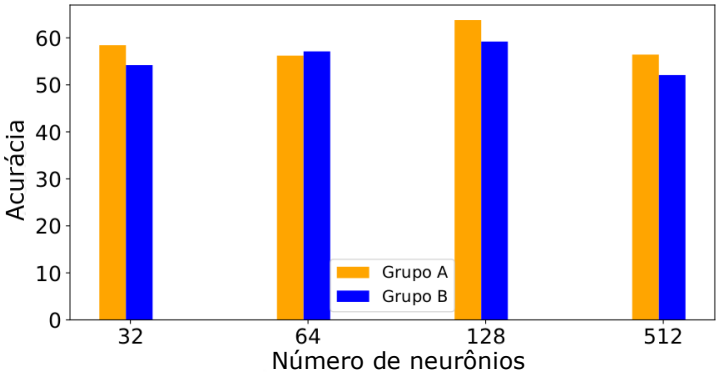
Figura 6.4: Acurácia da arquitetura LSTM com diferentes valores de neurônios



Fonte: Elaborado pelos autores

Na Figura 6.5, observa-se que, para o *Grupo A* e o *Grupo B*, a arquitetura GRU obteve maior acurácia com *num_neurônios* $= 128$. Já o *Grupo A* registrou baixa acurácia para o Região, porém o *Grupo B* obteve acurácias balanceadas para cada *feature*.

Figura 6.5: Acurácia da arquitetura GRU com diferentes valores de neurônios



Fonte: Elaborado pelos autores

Em relação ao número de neurônios, observa-se que o *num_neurônios* $= 128$ nas arquiteturas LSTM e GRU aprimorou os resultados, mas a arquitetura LSTM sempre obteve resultados superiores. O número de neurônios está relacionado à complexidade do modelo, pois favorece a capacidade de aprender padrões no modelo. Nesse sentido, conclui-se que o *num_neurônios* $= 128$ permite à arquitetura LSTM conseguir resultados balanceados nos *features* em relação ao *Grupo B*.

6.1.6 Avaliação do Número de Épocas

Metodologia

Este experimento foi executado com os parâmetros obtidos anteriormente, ou seja $h = 7$, $num\_camadas = 10$ e *num_neurônios* $= 128$. Este experimento apenas considerou as arquiteturas LSTM e GRU, porque foram as arquiteturas com melhores resultados nos experimentos anteriores. Além, apenas foi utilizado o *Grupo C* porque precisava-se conhecer os resultados em relação a históricos completos e incompletos. Em seguida, definiu-se o valor de *num_épocas = {100, 200, 300, 400, 500}*, e depois obteve-se a acurácia para cada *feature*, além da acurácia total por meio da Equação 6.1.

Resultados

Na Tabela 6.3, observa-se que a arquitetura LSTM conseguiu sua maior acurácia com 500 iterações, porém a arquitetura GRU obteve sua maior acurácia apenas com 200 iterações. Mesmo assim, a maior acurácia entre as duas (60,9%), foi obtida pela arquitetura LSTM que também obteve acurácias balanceadas para cada *feature*.

Tabela 6.3: Acurácia de modelos com diferentes valores de épocas

	LSTM	GRU
100 épocas	55,8%	56,9%
200 épocas	57,8%	**59,1%**
300 épocas	57,9%	54,7%
400 épocas	56,1%	54,4%
500 épocas	**60,9%**	50,8%

Dessa forma, conferiu-se que, LSTM conseguiu aprimorar os resultados com maior quantidade de épocas, em comparação à arquitetura GRU. Por outro lado, as melhorias dos resultados ao empregar o *Grupo C* não foram muito significativas, o que poderia significar que os dados incompletos afetam a análise.

6.1.7 Comparação dos Desempenhos das Arquiteturas com os *Baselines*

Metodologia

Este experimento precisou da execução das arquiteturas LSTM e GRU para o *Grupo C* em termos da acurácia e precisão. Esses resultados são considerados como os resultados finais que serão comparados com os *baselines* mencionados na subseção 3.1.4. A Tabela 6.5 mostra em detalhe os resultados obtidos para a previsão de cada atributo.

Resultados

Na Tabela 6.4, são descritos os resultados para cada um dos *features*. Na Tabela 6.5, observa-se que, dos seis *features*, LSTM conseguiu obter os melhores valores para quatro *features* (*CustomerID*, categoria, região e *monthly budget*) e GRU para apenas dois *features* (Tipo do dia e Pagamento). Inusitadamente, a arquitetura LSTM conseguiu uma alta acurácia na previsão do *feature* Região, mas poderia-se tratar de *overfitting*. Em resumo, até o momento os resultados validam que a arquitetura LSTM é promissora para analisar o comportamento de compra no *Online Retail Dataset*, embora o *dataset* não possua históricos com a informação completa e necessária para a análise.

Tabela 6.4: Descrição dos resultados para cada *feature*

Feature	**Descrição dos resultados**
CustomerID	Nenhuma das duas arquiteturas apresenta resultados significativos, pois não ultrapassa 50% da acurácia.
Tipo de dia	Ambas as arquiteturas trazem bons resultados que ultrapassam 50%.
Região	Ambas as arquiteturas ultrapassaram 50%, porém, a arquitetura LSTM melhorou a acurácia significativamente,
Categoria	Ambas as arquiteturas apresentam resultados aceitáveis que ultrapassam pelo menos 50%.
Monthly Budget	Nenhuma das duas arquiteturas apresenta resultados significativos, pois não ultrapassa 50% da acurácia.
Pagamento	Esse *feature* determina o valor total da próxima compra; ambas as arquiteturas passaram de 50% mas não significativamente.

Tabela 6.5: Acurácia para cada *feature* utilizando a configuração final

	CustomerID	**Tipo do dia**	**Categoria**	**Região**	***Monthly Budget***	**Pagamento**
LSTM	**33,7%**	80,1%	**59,7%**	**95,5%**	**46,2%**	60,8%
GRU	23,6%	**83,3%**	59,4%	88,2%	45,0%	**61,0%**

Na Tabela 6.6, observa-se que LSTM obteve 60,9% de acurácia e GRU obteve 59,1%, em que LSTM foi superior. Porém, GRU obteve 66,7% de precisão e LSTM apenas 63,2%. Embora LSTM possuisse uma acurácia superior, o modelo poderia não estar generalizado para os dados de teste, ou seja, poderia se tratar de um caso de *overfitting*. Além disso, a precisão obtida pelo GRU indicou uma melhor generalização do modelo durante a fase de teste, em que pelo menos 376.859 transações estariam sendo corretamente previstas. Ambos apresentaram resultados aceitáveis, porém, nenhum desses dois modelos mostrou boa performance (que superasse os 80%) na previsão, mas obtiveram uma grande melhoria em comparação aos *baselines* selecionados. Conclui-se que, embora LSTM obtivesse a melhor acurácia, GRU possui melhor capacidade de generalização na previsão da próxima transação no histórico de compra empregando *Online Retail Dataset*.

Tabela 6.6: Resultados em comparação aos *baselines*

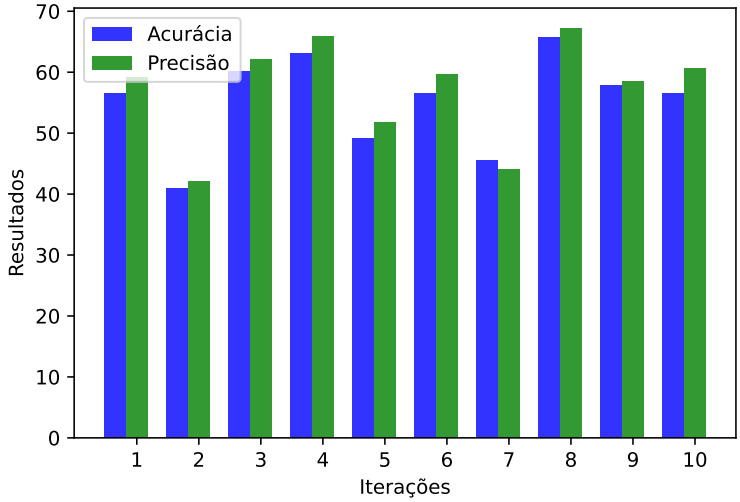
	Acurácia	Precisão
STPC-PGM	10,0%	12,5%
PRED	43,2%	23,6%
LSTM	**60,9%**	**63,2%**
GRU	**59,1%**	**66,7%**

6.1.8 Validação dos Resultados

Nessa etapa foi utilizado o método *hold out* para executar dez iterações do modelo com a configuração final dos parâmetros selecionados anteriormente. Esse método consiste em dividir o conjunto total de dados em dois subconjuntos mutuamente exclusivos (YADAV; SHUKLA, 2016), um para o treinamento (estimação dos parâmetros) e outro para o teste (validação), em que os dados são selecionados aleatoriamente.

Na Figura 6.6, observa-se que para três iterações, os resultados não superaram o 50%, isso pode ser indicativo que nos dados de treinamento não se registrou a informação necessária para que o modelo aprendesse corretamente. No caso da iteração 8, os dados do treinamento conseguiram uma melhor execução do modelo e por isso se obteve os melhores resultados. Além disso, observa-se que a métrica da precisão quase sempre obteve melhores resultados em comparação à acurácia.

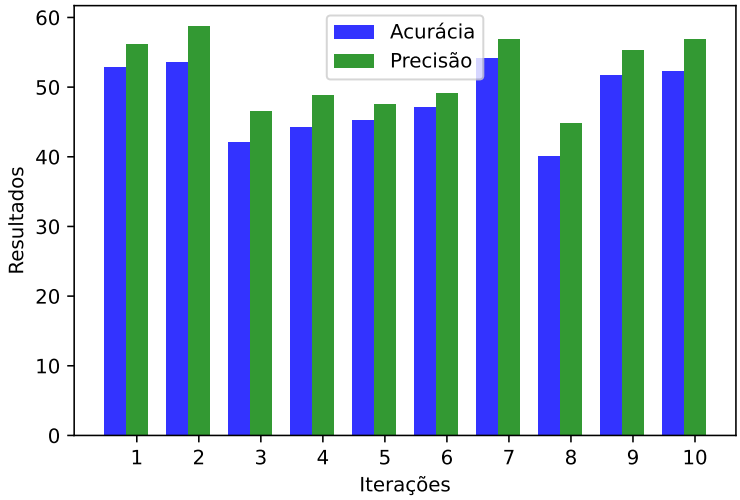
Figura 6.6: Acurácia da arquitetura LSTM com diferentes valores de neurônios



Fonte: Elaborado pelos autores

Na Figura 6.7, observa-se que para a metade das iterações, os resultados não superaram o 50%, isso pode ser indicativo que nos dados de treinamento não se registrou a informação necessária para que o modelo aprendesse corretamente. No caso da iteração 7, os dados do treinamento conseguiram uma melhor execução do modelo e por isso se obteve os melhores resultados. Além disso, observa-se que a métrica da precisão sempre obteve melhores resultados em comparação à acurácia.

Figura 6.7: Acurácia da arquitetura LSTM com diferentes valores de neurônios



Fonte: Elaborado pelos autores

Em resumo, observa-se que os resultados de cada um dos modelos estão ligados com a informação que seja utilizada durante o treinamento, já que os modelos devem possuir a capacidade de generalização certa. Nesse sentido, poderia optar-se por outros métodos de treinamento para aprimorar os resultados. Por outro lado, existem bibliotecas de otimização de hiperparâmetros, que faz descoberta de valoração de parâmetros, o que permitiria ganhar tempo e experimentar outro novos parâmetros.

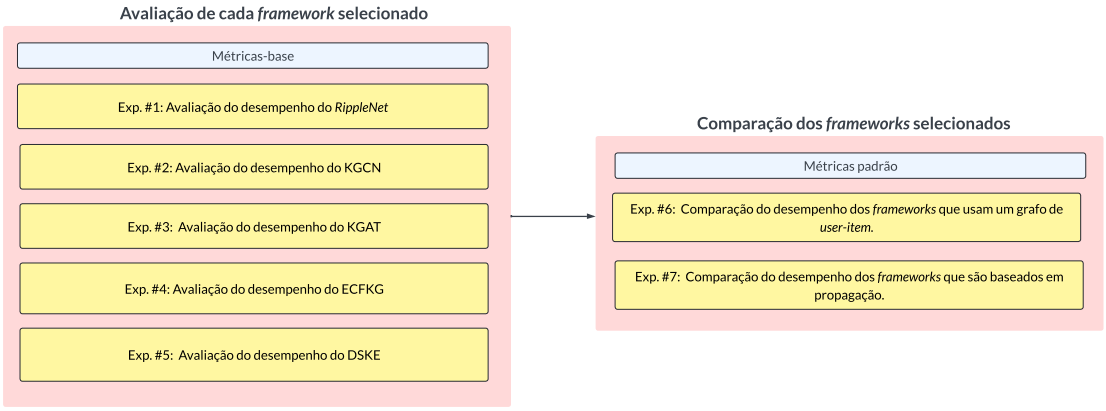
6.2 Sobre a Avaliação de *Frameworks* de Recomendações Explicáveis

6.2.1 Objetivos

De acordo com a visão geral dos experimentos, como se mostra na Figura 6.8, os primeiros cinco experimentos foram propostos para comparar o desempenho de cada *framework*. Para *RippleNet* e KGCN, os conjuntos de dados de quatro versões (*Dataset* v1, *Dataset* v2, *Dataset* v3 e *Dataset* v4) foram usados. Para KGAT, ECFKG e DSKE, apenas foi usado o *Dataset* v4, porque essa versão do *dataset* apresentou os melhores resultados nos experimentos prévios. Ademais, os dois últimos experimentos foram avaliados em termos de uma métrica padrão (*Recall@20*) e também foi empregado o *Dataset* v4. Para conseguir analisar ainda mais os resultados, também foram considerados os *datasets* que acompanham na implementação de cada *framework* (por exemplo, *Movies, Book, LastFM, Yelp* etc.). A seguir, esta descrito o objetivo de cada experimento proposto:

1. Avaliação do desempenho do *RippleNet*.
2. Avaliação do desempenho do KGCN.
3. Avaliação do desempenho do KGAT.
4. Avaliação do desempenho do ECFKG.
5. Avaliação do desempenho do DSKE.
6. Comparação do desempenho dos *frameworks* que usam o *grafo de user-item*.
7. Comparação do desempenho dos *frameworks* que são baseados em propagação.

Figura 6.8: Visão geral dos experimentos para a avaliação dos *frameworks*



Fonte: Elaborado pelos autores

6.2.2 Métricas

A Tabela 3.6 resume as métricas que foram utilizadas para cada *framework*. Nesse contexto, o $Recall@20$ foi estabelecido como a métrica padrão. A Equação 6.2 indica a forma de calcular essa métrica, em que é representada pela fração dos k principais itens recomendados que estão em um conjunto de itens relevantes para o usuário. As outras métricas usadas nos próximos experimentos foram descritas na subseção 3.2.1.

$$Recall@k = \frac{\text{nº das k principais recomendações relevantes}}{\text{nº de todos os itens relevantes}} \tag{6.2}$$

6.2.3 Avaliação do Desempenho do *RippleNet*

Metodologia

O código-fonte oferece uma etapa de pré-processamento, na qual os dados foram organizados com base nos arquivos de entrada descritos na seção 5.5. Assim, foram gerados *(1) kg_final*, arquivo que contém a representação final do grafo, e *(2) ratings_final*, arquivo que transforma o *rating* inicial como 1, se o item foi assistido, e 0 se não.

Antes da execução do *framework*, os hiperparâmetros são definidos da seguinte forma: `dim = 32`, que corresponde ao tamanho de incorporação, `n_memory = 32`, que corresponde ao tamanho de incorporação em cada hop e `n_hop = 3`, que corresponde ao número de saltos. Em segundo lugar, os parâmetros da rede são configurados da seguinte forma: `kge_weight = 0,01`, corresponde ao peso dos *embeddings*, `l2_weight = 1e-7`, corresponde ao peso do termo de regularização, `lr = 0,02`, corresponde à taxa de aprendizagem, `tamanho do lote = 500`, corresponde ao tamanho do lote, e `n_epoch = 100`, corresponde ao número de épocas.

Neste experimento foram incluídos os *datasets Movies* e *Book-Crossing*, assim como as quatro versões do *Streaming Platform Dataset*. A Tabela 6.7 descreve as estatísticas de cada *dataset* usado para a execução do *RippleNet*[3], além disso os resultados são apresentados em termos de acurácia e AUC com base na previsão de CTR.

Resultados

Na Tabela 6.7, observa-se que *Movies* e o *Dataset* v4 obtêm os melhores resultados em termos de acurácia e AUC, isso indica que uma densidade alta nos *datasets* permitem uma análise muito mais completa. Ademais, o uso do *Rating* Implícito, no *Dataset* v4, teve um impacto positivo porque esse *feature* foi gerado a partir de informações contidas no *dataset*. *RippleNet* é um *framework* que utiliza um *grafo de item*, pelo qual o número de itens no *dataset* garantiu uma representação ainda melhor desses itens no grafo.

[3]https://github.com/hwwang55/RippleNet

Tabela 6.7: Estatísticas e resultados de cada conjunto de dados ao analisar *RippleNet*

	Movies	*Book*	*Dataset v1*	*Dataset v2*	*Dataset v3*	*Dataset v4*
#usuários	6.036	17.860	1.141.995	1.141.995	1.141.995	36.397
#itens	2.445	14.962	32.117	32.117	32.117	30.608
#relações	12	25	8	8	8	8
#interações	753.772	139.746	2.195.785	2.195.785	2.195.785	2.000.000
Densidade (int./user)	124,88	7,82	1,91	1,91	1,91	54,95
Acurácia	0,84	0,66	0,75	0,77	0,78	0,85
AUC	0,92	0,73	0,80	0,83	0,85	0,92

6.2.4 Avaliação do Desempenho do KGCN

Metodologia

O código-fonte incluiu uma fase de pré-processamento para definir as representações das entidades, portanto, a versão final do grafo foi construída a partir dos arquivos de entrada descritos na seção 5.5. Como resultado do pré-processamento, foram gerados: *(1) kg_final*, arquivo que contém a representação final do grafo, e *(*2) ratings_final, arquivo que transforma o *rating* inicial como 1, se o item foi assistido, e 0, se não.

Antes de executar o *framework*, os hiperparâmetros foram configurados da seguinte forma: `dim = 32`, corresponde ao tamanho do *embedding*, `neighbor_sample_size = 32`, corresponde ao número de vizinhos, `agregador = soma`, corresponde ao tipo de agregação que será utilizada na conexão dos vizinhos, e `n_iter = 2` é o número de iterações no cálculo da representação. Os parâmetros da rede foram configurados da seguinte forma: `l2_weight = 1e-7`, corresponde ao peso do termo de regularização, `lr = 2e-2`, corresponde à taxa de aprendizagem, `tamanho do lote = 1024`, corresponde ao tamanho do lote, `n_epochs = 50`, corresponde ao número de épocas, e `ratio`, que corresponde ao tamanho do treinamento.

Neste experimento foram incluídos os *datasets Movie* e *LastFM*, assim como as quatro versões do *Streaming Platform Dataset*. A Tabela 6.8 descreve as estatísticas básicas de cada *dataset* utilizado na execução do KGCN[4], bem como os resultados apresentados em termos de F1, AUC e *Recall@20*.

[4]https://github.com/hwwang55/KGCN

Resultados

Os resultados mostraram que *Movies* apresentou melhores resultados em termos de F1, AUC e Recall@20. Ademais, os resultados de *Dataset* v4, em termos de F1 e AUC, indicaram um desempenho superior dessa variação comparada às outras três versões. Assim como no experimento 6.2.3, observou-se resultados melhores se a densidade do *dataset* é maior. Além disso, o uso do *Rating* Implícito parece otimizar os resultados e o número de itens garante uma melhor representação das entidades no *grafo de item*.

A versão do *dataset Movies* era de 20 milhões de interações entre usuários e filmes, disponibilizando 20 vezes mais informações para o treinamento do *framework*, comparado à sua versão com apenas 1 milhão de interações (no *RippleNet*). Essa mudança ocasionou aumento nas métricas de acurácia e AUC, isso indica que KGCN pode aprimorar os resultados ao empregar *datasets* mais extensos.

Tabela 6.8: Estatísticas e resultados de cada conjunto de dados ao analisar KGCN

	Movies	*LastFM*	*Dataset v1*	*Dataset v2*	*Dataset v3*	*Dataset v4*
#usuários	138.159	1.872	1.141.995	1.141.995	1.141.995	36.397
#itens	16.954	3.846	32.117	32.117	32.117	30.608
#relações	32	60	8	8	8	8
#interações	13.501.622	42.346	2.195.785	2.195.785	2.195.785	2.000.000
Densidade (int./user)	97,73	22,62	1,91	1,91	1,91	54,95
F1	0,93	0,69	0,62	0,64	0,66	0,83
AUC	0,98	0,80	0,63	0,65	0,67	0,90
Recall@20	0,16	0,11	0,05	0,05	0,05	0,08

6.2.5 Avaliação do Desempenho do KGAT

Metodologia

Antes da execução, os hiperparâmetros foram configurados da seguinte forma: `embed_size = 64`, corresponde ao tamanho dos *embeddings*, `pretrain = -1`, corresponde ao uso de um modelo treinado anteriormente. Em segundo lugar, `alg_type bi`, específica o tipo de camada convolucional do grafo, `regs = [1e-5,1e-5]`, corresponde ao termo de regularização para os *embeddings* de usuário e item, `layer_size`

= [64,32,16], corresponde ao tamanho das saídas da camada, lr = 0,0001, corresponde à taxa de aprendizagem, época = 1000, corresponde ao número de épocas, batch_size = 1024, corresponde ao tamanho do lote, node_dropout = [0.1], corresponde a manter ou não a probabilidade em cada camada profunda, use_att = True, corresponde a usar ou não o mecanismo de atenção, e use_kge = True, corresponde a usar ou não o *embedding* no grafo de conhecimento.

Nesse experimento foram usados os *datasets AmazonBook*, *LastFM* e *Yelp*, assim como *Dataset* v4 (porque obteve os melhores resultados nos experimentos 6.2.3 e 6.2.4). A Tabela 6.9 descreve as estatísticas básicas de cada *dataset* usado na execução do KGAT[5], além dos resultados apresentados em termos de Recall@20 e NDGC@20.

Resultados

Os resultados mostraram que o *Dataset* v4 obteve melhores resultados para ambas as métricas; desse modo foi verificado, mais uma vez, que um *dataset* com densidade alta e o uso do *Rating implícito* têm impacto positivo nos resultados. Além disso, embora o *LastFM* possuísse maior densidade, ele não conseguiu superar os resultados obtidos por *Dataset* v4 pelo fato de ter menor número de usuários. KGAT é um *framework* de propagação que utiliza o *grafo de user-item*, então, se o *dataset* contém um número balanceado de usuários e itens, é possível explorar relações de maior importância no grafo.

Tabela 6.9: Estatísticas e resultados de cada conjunto de dados ao analisar KGAT

	AmazonBook	*LastFM*	*Yelp*	*Dataset* v4
#usuários	70.679	23.566	45.919	36.397
#itens	24.915	48.123	45.538	30.608
#relações	39	9	42	8
#interações	847.733	3.034.796	1.853.068	2.000.000
Densidade (int./user)	11,99	128,78	25,81	54,95
Recall@20	0,15	0,09	0,07	0,20
NDGC@20	0,10	0,13	0,09	0,25

[5]https://github.com/xiangwang1223/knowledge_graph_attention_network

6.2.6 Avaliação do Desempenho do ECFKG

Metodologia

O código-fonte do ECFKG é encontrado no KGAT, portanto, os arquivos de entrada, os hiperparâmetros e as métricas são os mesmos que foram descritos na subseção 6.2.5. Também as estatísticas dos *datasets* empregadas são encontradas na Tabela 6.9.

Resultados

Nesse quinto experimento, observa-se que ECFKG é o *framework* com os resultados mais baixos em comparação aos outros. Os resultados mostrados na Tabela 6.10 indicam que *Dataset* v4 não obteve os melhores resultados em termos de *Recall@20* e, no caso de NDGC@20, não houve melhoras. Em síntese, um *framework* baseado em *embeddings* não é suficiente para analisar *datasets* de alta densidade, pois é focado no melhoramento das representações, deixando de fora aqueles caminhos que podem evidenciar relações relevantes entre as entidades.

Tabela 6.10: Resultados de cada conjunto de dados ao analisar ECFKG

	AmazonBook	*LastFM*	*Yelp*	*Dataset* v4
Recall@20	0,11	0,07	0,05	0,07
NDGC@20	0,08	0,11	0,06	0,11

6.2.7 Avaliação do Desempenho do DSKE

Metodologia

Neste experimento foram incluídos os *datasets Yelp* e *Douban*, assim como *Dataset* v4. A Tabela 6.11 descreve as estatísticas básicas dos *datasets* empregados na execução do DSKE[6], além disso, os resultados são apresentados em termos de *Recall@20* e NDGC@20, e os meta-caminhos foram definidos manualmente (indicado na seção 5.5).

[6]https://github.com/yuan-pku/Distilling-Structured-Knowledge-into-Embeddings-for-Explainable-and-Accurate-Recommendation

Resultados

Na Tabela 6.11, observa-se que o *Dataset* v4 obteve os melhores resultados para ambas as métricas em comparação aos outros os conjuntos de dados. Mesmo que *Douban* possuía uma densidade mais alta, *Dataset* v4 atingiu melhores resultados pelo fato de ter maior número de usuários e itens porque esse *framework* utiliza um *grafo de user-item*. Em suma, DSKE obteve os melhores resultados em comparação aos outros *frameworks*, porém a definição manual dos meta-caminhos poderia ser o motivo. Mesmo assim, a modelagem manual pode afetar diretamente na escalabilidade se o grafo variasse sua estrutura e fosse necessário considerar novas informações.

Tabela 6.11: Estatísticas e resultados de cada conjunto de dados ao analisar DSKE

	Yelp	*Douban*	*Dataset v4*
#usuários	16.239	13.367	36.397
#itens	14.284	12.677	30.608
#relações	4	7	8
#interações	198.397	1.068.278	2.000.000
Densidade (int./user)	12,22	79,92	54,95
Recall@20	0,10	0,24	0,47
NDGC@20	0,11	0,26	0,46

6.2.8 Comparação do Desempenho dos *Frameworks* que Usam *Grafo de User-Item*

Metodologia

Os *frameworks* que usam o *grafo de user-item* oferecem as explicações das recomendações verificando os caminhos salientes que conectam o usuário-alvo e o item-candidato (GUO et al., 2020). Desse modo, o usuário é incorporado como um tipo de nó e o padrão de conexão pode ser explorado em maior extensão. Nessa perspectiva, foram avaliados os *frameworks* que empregaram um *grafo de user-item* para evidenciar fatores que poderiam ter implicações nos resultados. Logo, ECFKG, DSKE e KGAT foram comparados em termos de *Recall@20* utilizando *Dataset* v4.

Resultados

A Tabela 6.12 mostra que KGAT e DSKE têm melhor desempenho para análise do *Dataset* v4 usando um *grafo de user-item*. ECFKG apresentou os resultados mais baixos, isso indica que o modelo baseado em *embeddings* não é suficiente para capturar conexões de alta ordem, mesmo tenha sido empregado o *grafo de user-item*. DSKE utiliza um modelo diferenciável baseado em conexões para ajudar na interpretação e aprendizagem dos *embeddings*, mas esse *framework* utilizou uma definição manual dos meta-caminhos. Esse seria o motivo pelo que DSKE apresenta melhores resultados comparado ao KGAT, um *framework* que considera diretamente as relações de alta ordem no modelo preditivo. Embora o DSKE tenha apresentado os melhores resultados, ele não é escalável caso o grafo mude sua estrutura ou considere novas informações.

Tabela 6.12: Comparação do desempenho dos *frameworks* que usam o grafo *user-item*

	ECFKG	KGAT	DSKE
Modelo baseado em	*Embeddings*	Propagação	Conexões
Metodo KGE	TransE	TransR	*end-to-end*
Resultados	0,067	0,199	0,474

6.2.9 Comparação do Desempenho dos *Frameworks* Baseados em Propagação

Metodologia

Os métodos baseados em propagação são geralmente custosos em termos computacionais. À medida que o grafo cresce, torna-se difícil para o modelo convergir (GUO et al., 2020). Para melhorar a eficiência, foram propostas a operação convolucional de grafos, assim como a amostragem vizinha em cada camada. Desse modo, foram avaliados os *frameworks* baseados em propagação para evidenciar alguns fatores que poderiam ter implicações nos resultados. Assim, *RippleNet*, KGCN e KGAT foram comparados em termos de *Recall@20* para o *Dataset* v4.

Resultados

A Tabela 6.13 mostra que apesar de *RippleNet* tenha usado *attention mechanism* para atribuir os pesos de acordo com o nível de importância dos nós vizinhos durante a agregação, e KGCN tenha usado um *Graph Convolutional Network* (GCN) para ge-

neralizar a operação convolucional permitindo capturar informações localizadas; o *grafo de item* não foi suficiente para espalhar a informação completamente no grafo. KGAT é um *framework* que usa o *grafo de user-item* com resultados relativamente superiores, isso indica a importância do refinamento desse tipo do grafo nesse *framework*, que propaga recursivamente os *embeddings* dos vizinhos de um nó por meio *graph attention mechanism* para discriminar a importância dos vizinhos.

Tabela 6.13: Comparação de *frameworks* baseados em propagação

	RippleNet	KGCN	KGAT
Tipo do grafo	*grafo de Item*	*grafo de Item*	*grafo de user-item*
Método KGE	*end-to-end*	*end-to-end*	TransR
Resultados	0,017[7]	0,083	0,199

[7]Este valor foi obtido por meio de uma outra implementação do *RippleNet*

7 CONCLUSÃO

Este Capítulo apresenta as conclusões após (1) executar modelos baseados em arquiteturas do tipo RNN (LSTM e GRU) para prever o comportamento de compra , bem como (2) avaliar os *frameworks* de sistemas de recomendações explicáveis baseados em grafo de conhecimento. Além disso, são expostas as conclusões em relação aos experimentos, assim como os possíveis trabalhos futuros.

No caso da previsão do comportamento de compra, enfatiza-se que a ciência de dados foi vital nesse objetivo porque conseguiu-se deixar implementados trés modelos (RNN, LSTM e GRU) que permitam modelar o comportamento sequencialmente. Além disso, os resultados refletem a capacidade desses modelos para trabalhar com um *dataset* que não inclua informações privadas dos clientes, em comparação com os trabalhos relacionados que utilizam dados privados que não podem ser fornecidos sem um acordo de confidencialidade. Assim, LSTM obteve 60,9% em termos de acurácia e 63,2% em termos de precisão, enquanto GRU obteve 59,1% em termos de acurácia e 66,7% em termos de precisão. Os resultados refletiram melhorias em comparação aos *baselines* replicados. Ademais, os modelos apresentados são escaláveis na medida em que for usado outro *dataset*, apenas sendo necessário realizar as etapas descritas. As duas arquiteturas confirmaram seu potencial para esse tipo de análise, mas LSTM teve problemas na generalização nos dados de teste, pelo fato de obter uma menor precisão em comparação ao GRU. Nesse sentido, os resultados podem ter sido influenciados também pelas inconsistências presentes no *dataset*. Como trabalho futuro, poderiam ser empregadas outras métricas para avaliar ainda mais os resultados e identificar outras questões em relação à performance dos modelos (por exemplo, *overfitting*), assim como outros métodos de treinamento (*Bagging* ou *Boosting*) para melhorar o desempenho preditivo. Além disso, também poderiam ser incluído o uso bibliotecas de otimização de hiperparâmetros. Em relação ao *dataset*, os dados poderiam ser complementados e, desse modo, garantir uma melhor analise como é requerido na ciência de dados.

Por outro lado, no caso da avaliação de sistemas de recomendações, conseguiu-se que a ciência de dados ajude no processo de avaliação de cinco *frameworks* (*RippleNet*, KGCN, KGAT, ECFKG e DSKE) baseados no grafo de conhecimento para obter recomendações explicáveis usando um *dataset* de uma plataforma *streaming* que continha dados reais. Apresentou-se a descrição do processo empregado para a execução de cada um dos *frameworks*, assim como os experimentos que foram propostos para avaliar o im-

pacto do tamanho do *dataset*, o tipo de grafo de conhecimento empregado e os *frameworks* baseados em propagação. Uma limitação encontrada nos experimentos foi o custo computacional dos *frameworks* devido à quantidade de dados do *Streaming Platform Dataset*. Apesar das limitações computacionais, parece viável utilizar outros métodos para selecionar os dados que possam ampliar a gama de análises para gerar melhores recomendações. Os melhores resultados em termos de *Recall@20* foram obtidos pelo KGAT com 0,199 e DSKE com 0,474, porém KGAT não usou nenhum tipo de modelagem manual a diferença de DSKE. O fato de empregar um *dataset* extenso requer de um framework com a capacidade prever diretamente as relações de alta ordem, como é o caso de KGAT. Como trabalho futuro, recomenda-se propor uma versão melhorada do grafo de conhecimento que inclua outros *features* do *dataset* que não foram considerados nessa proposta.

Por fim, o uso da ciência de dados na área de *E-commerce* é promissora, mas o cientista de dados deve garantir o tratamento responsável dos dados, e sua ética deve prevalecer nas decisões ao realizar diversas análises.

Um artigo contendo a avaliação desses *frameworks* de SR explicáveis baseados em grafo de conhecimento foi aceito e publicado na *Proceedings of the 29th Brazilian Symposium on Multimedia and the Web* (MARISCAL et al., 2023).

REFERÊNCIAS

AALST, W. V. D.; AALST, W. van der. **Data science in action**. [S.l.]: Springer, 2016.

AGGARWAL, C. C. et al. **Recommender systems**. [S.l.]: Springer, 2016.

AI, Q. et al. Learning heterogeneous knowledge base embeddings for explainable recommendation. **Algorithms**, MDPI, v. 11, n. 9, p. 137, 2018.

ALOM, M. Z. et al. A state-of-the-art survey on deep learning theory and architectures. **Electronics**, Multidisciplinary Digital Publishing Institute, v. 8, n. 3, p. 292, 2019.

AMPHAWAN, K.; LENCA, P.; SURARERKS, A. Efficient mining top-k regular-frequent itemset using compressed tidsets. In: SPRINGER. **Pacific-Asia Conference on Knowledge Discovery and Data Mining**. [S.l.], 2011. p. 124–135.

CAO, Y. et al. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In: **The world wide web conference**. [S.l.: s.n.], 2019. p. 151–161.

CHEN, D.; SAIN, S. L.; GUO, K. Data mining for the online retail industry: A case study of rfm model-based customer segmentation using data mining. **Journal of Database Marketing & Customer Strategy Management**, Springer, v. 19, n. 3, p. 197–208, 2012.

CHICAIZA, J.; VALDIVIEZO-DIAZ, P. A comprehensive survey of knowledge graph-based recommender systems: Technologies, development, and contributions. **Information**, MDPI, v. 12, n. 6, p. 232, 2021.

CHO, K. et al. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. [S.l.], 2014. p. 1724.

DINEV, T.; HART, P. An extended privacy calculus model for e-commerce transactions. **Information systems research**, Informs, v. 17, n. 1, p. 61–80, 2006.

DOLPHIN, R.; SMYTH, B.; DONG, R. A machine learning approach to industry classification in financial markets. In: SPRINGER. **Irish Conference on Artificial Intelligence and Cognitive Science**. [S.l.], 2022. p. 81–94.

ELAHI, E.; HALIM, Z. Graph attention-based collaborative filtering for user-specific recommender system using knowledge graph and deep neural networks. **Knowledge and Information Systems**, Springer, v. 64, n. 9, p. 2457–2480, 2022.

EREVELLES, S.; FUKAWA, N.; SWAYNE, L. Big data consumer analytics and the transformation of marketing. **Journal of business research**, Elsevier, v. 69, n. 2, p. 897–904, 2016.

FENG, D.-C. et al. Machine learning-based compressive strength prediction for concrete: An adaptive boosting approach. **Construction and Building Materials**, Elsevier, v. 230, p. 117000, 2020.

FLEDER, M.; SHAH, D. I know what you bought at chipotle for $9.81 by solving a linear inverse problem. **Proceedings of the ACM on Measurement and Analysis of Computing Systems**, ACM New York, NY, USA, v. 4, n. 3, p. 1–17, 2020.

GAO, C. et al. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. **ACM Transactions on Recommender Systems**, ACM New York, NY, USA, v. 1, n. 1, p. 1–51, 2023.

GUO, Q. et al. A survey on knowledge graph-based recommender systems. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 34, n. 8, p. 3549–3568, 2020.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT press, v. 9, n. 8, p. 1735–1780, 1997.

HUANG, C. et al. Online purchase prediction via multi-scale modeling of behavior dynamics. In: **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. [S.l.: s.n.], 2019. p. 2613–2622.

HUANG, J. et al. Improving sequential recommendation with knowledge-enhanced memory networks. In: **The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval**. [S.l.: s.n.], 2018. p. 505–514.

KOLLER, D.; FRIEDMAN, N. **Probabilistic graphical models: principles and techniques**. [S.l.]: MIT press, 2009.

KOLLER, D.; PFEFFER, A. Probabilistic frame-based systems. In: **AAAI/IAAI**. [S.l.: s.n.], 1998. p. 580–587.

ŁADYŻYŃSKI, P.; ŻBIKOWSKI, K.; GAWRYSIAK, P. Direct marketing campaigns in retail banking with the use of deep learning and random forests. **Expert Systems with Applications**, Elsevier, v. 134, p. 28–35, 2019.

LI, D.; QU, H.; WANG, J. A survey on knowledge graph-based recommender systems. In: IEEE. **2023 China Automation Congress (CAC)**. [S.l.], 2023. p. 2925–2930.

LI, J. et al. A machine learning based method for customer behavior prediction. **Tehnički vjesnik**, Strojarski fakultet u Slavonskom Brodu; Fakultet elektrotehnike, računarstva . . . , v. 26, n. 6, p. 1670–1676, 2019.

LI, Q.; CHEN, Z.; ZHAO, H. V. Prima++: A probabilistic framework for user choice modelling with small data. **IEEE Transactions on Signal Processing**, IEEE, v. 69, p. 1140–1153, 2021.

LIMA, B. S. M. d. Análise de aplicabilidade de recomendações baseadas em grafos para conteúdos multimídia do globoplay. 2022.

MA, W. et al. Jointly learning explainable rules for recommendation with knowledge graph. In: **The world wide web conference**. [S.l.: s.n.], 2019. p. 1210–1221.

MARISCAL, C. S. et al. Assessing explainable recommendations from knowledge graph-based in an international streaming platform. In: **Proceedings of the 29th Brazilian Symposium on Multimedia and the Web**. [S.l.: s.n.], 2023. p. 213–220.

MARTÍNEZ, A. et al. A machine learning framework for customer purchase prediction in the non-contractual setting. **European Journal of Operational Research**, Elsevier, v. 281, n. 3, p. 588–596, 2020.

MONTEIRO, R. L. Existe um direito à explicação na lei geral de proteção de dados do brasil. **Artigo estratégico**, n, v. 39, p. 1–14, 2018.

MULHOLLAND, C. S. Dados pessoais sensíveis e a tutela de direitos fundamentais: uma análise à luz da lei geral de proteção de dados (lei 13.709/18). **Revista de Direitos e Garantias Fundamentais**, Faculdade de Direito de Vitória, v. 19, n. 3, p. 159–180, 2018.

PINHEIRO, P. P. **Proteção de dados pessoais: Comentários à lei n. 13.709/2018-lgpd**. [S.l.]: Saraiva Educação SA, 2020.

RENDLE STEFFEN, F.; SCHMIDT-THIEME. Factorizing personalized markov chains for next-basket recommendation. In: **Proceedings of the 19th international conference on World wide web**. [S.l.: s.n.], 2010. p. 811–820.

RUIZ, F. J.; ATHEY, S.; BLEI, D. M. Shopper: A probabilistic model of consumer choice with substitutes and complements. 2020.

SAFARA, F. A computational model to predict consumer behaviour during covid-19 pandemic. **Computational Economics**, Springer, v. 59, n. 4, p. 1525–1538, 2022.

SARKAR, M.; BRUYN, A. D. Lstm response models for direct marketing analytics: Replacing feature engineering with deep learning. **Journal of Interactive Marketing**, SAGE Publications Sage CA: Los Angeles, CA, v. 53, n. 1, p. 80–95, 2021.

SAURA, J. R. Using data sciences in digital marketing: Framework, methods, and performance metrics. **Journal of Innovation & Knowledge**, Elsevier, v. 6, n. 2, p. 92–102, 2021.

SHRESTHA, A.; MAHMOOD, A. Review of deep learning algorithms and architectures. **IEEE access**, IEEE, v. 7, p. 53040–53065, 2019.

TABIANAN, K.; VELU, S.; RAVI, V. K-means clustering approach for intelligent customer segmentation using customer purchase behavior data. **Sustainability**, MDPI, v. 14, n. 12, p. 7243, 2022.

TATTI, N.; MOERCHEN, F.; CALDERS, T. Finding robust itemsets under subsampling. **ACM Transactions on Database Systems (TODS)**, ACM New York, NY, USA, v. 39, n. 3, p. 1–27, 2014.

VASUPULA, N.; MUNNANGI, V.; DAGGUBATI, S. Modern privacy risks and protection strategies in data analytics. In: SPRINGER. **Soft Computing and Signal Processing: Proceedings of 3rd ICSCSP 2020, Volume 2**. [S.l.], 2022. p. 81–89.

VELICKOVIC, P. et al. Graph attention networks. **stat**, v. 1050, n. 20, p. 10–48550, 2017.

VOIGT, P.; BUSSCHE, A. Von dem. The eu general data protection regulation (gdpr). **A Practical Guide, 1st Ed., Cham: Springer International Publishing**, Springer, v. 10, p. 3152676, 2017.

WADE, C.; GLYNN, K. **Hands-On Gradient Boosting with XGBoost and scikit-learn: Perform accessible machine learning and extreme gradient boosting with Python**. [S.l.]: Packt Publishing Ltd, 2020.

WANG, H. et al. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In: **Proceedings of the 27th ACM international conference on information and knowledge management**. [S.l.: s.n.], 2018. p. 417–426.

WANG, H. et al. Dkn: Deep knowledge-aware network for news recommendation. In: **Proceedings of the 2018 world wide web conference**. [S.l.: s.n.], 2018. p. 1835–1844.

WANG, H. et al. Multi-task feature learning for knowledge graph enhanced recommendation. In: **The world wide web conference**. [S.l.: s.n.], 2019. p. 2000–2010.

WANG, H. et al. Knowledge graph convolutional networks for recommender systems. In: **The world wide web conference**. [S.l.: s.n.], 2019. p. 3307–3313.

WANG, S. et al. Graph learning based recommender systems: A review. **arXiv preprint arXiv:2105.06339**, 2021.

WANG, W. et al. A user purchase behavior prediction method based on xgboost. **Electronics**, MDPI, v. 12, n. 9, p. 2047, 2023.

WANG, X. et al. A survey on heterogeneous graph embedding: methods, techniques, applications and sources. **IEEE Transactions on Big Data**, IEEE, v. 9, n. 2, p. 415–436, 2022.

WANG, X. et al. Kgat: Knowledge graph attention network for recommendation. In: **Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining**. [S.l.: s.n.], 2019. p. 950–958.

WANG, X. et al. Learning intents behind interactions with knowledge graph for recommendation. In: **Proceedings of the web conference 2021**. [S.l.: s.n.], 2021. p. 878–887.

WEN, Y.-T. et al. Customer purchase behavior prediction from payment datasets. In: **Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining**. [S.l.: s.n.], 2018. p. 628–636.

WIERINGA, J. et al. Data analytics in a privacy-concerned world. **Journal of Business Research**, Elsevier, v. 122, p. 915–925, 2021.

WU, S. et al. Graph neural networks in recommender systems: a survey. **ACM Computing Surveys**, ACM New York, NY, v. 55, n. 5, p. 1–37, 2022.

XIAN, Y. et al. Reinforcement knowledge graph reasoning for explainable recommendation. In: **Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval**. [S.l.: s.n.], 2019. p. 285–294.

YADAV, S.; SHUKLA, S. Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In: IEEE. **2016 IEEE 6th International conference on advanced computing (IACC)**. [S.l.], 2016. p. 78–83.

YANG, Z.; DONG, S. Hagerec: Hierarchical attention graph convolutional network incorporating knowledge graph for explainable recommendation. **Knowledge-Based Systems**, Elsevier, v. 204, p. 106194, 2020.

YU, X. et al. Recommendation in heterogeneous information networks with implicit user feedback. In: **Proceedings of the 7th ACM conference on Recommender systems**. [S.l.: s.n.], 2013. p. 347–350.

YU, X. et al. Personalized entity recommendation: A heterogeneous information network approach. In: **Proceedings of the 7th ACM international conference on Web search and data mining**. [S.l.: s.n.], 2014. p. 283–292.

YU, Y. et al. A review of recurrent neural networks: Lstm cells and network architectures. **Neural computation**, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 31, n. 7, p. 1235–1270, 2019.

YUAN, Q. et al. Pred: Periodic region detection for mobility modeling of social media users. In: **Proceedings of the Tenth ACM International Conference on Web Search and Data Mining**. [S.l.: s.n.], 2017. p. 263–272.

ZHANG, J.-C. et al. A review of recommender systems based on knowledge graph embedding. **Expert Systems with Applications**, Elsevier, p. 123876, 2024.

ZHANG, Y. et al. Learning over knowledge-base embeddings for recommendation. **arXiv preprint arXiv:1803.06540**, 2018.

ZHANG, Y.; CHEN, X. et al. Explainable recommendation: A survey and new perspectives. **Foundations and Trends® in Information Retrieval**, Now Publishers, Inc., v. 14, n. 1, p. 1–101, 2020.

ZHANG, Y. et al. Distilling structured knowledge into embeddings for explainable and accurate recommendation. In: **Proceedings of the 13th international conference on web search and data mining**. [S.l.: s.n.], 2020. p. 735–743.

ZHAO, J. et al. Intentgc: a scalable graph convolution framework fusing heterogeneous information for recommendation. In: **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. [S.l.: s.n.], 2019. p. 2347–2357.

ZHOU, M. et al. Micro behaviors: A new perspective in e-commerce recommender systems. In: **Proceedings of the eleventh ACM international conference on web search and data mining**. [S.l.: s.n.], 2018. p. 727–735.

ZHU, B. et al. Location-based hybrid deep learning model for purchase prediction. In: IEEE. **2020 5th International Conference on Computational Intelligence and Applications (ICCIA)**. [S.l.], 2020. p. 161–165.