Universidade Federal de São Carlos– UFSCar
Centro de Ciências Exatas e de Tecnologia– CCET
Departamento de Computação– DC
Programa de Pós-Graduação em Ciência da Computação– PPGCC

Rafael Tofoli Sereicikas

Representação Vetorial Consciente de Tempo para Recomendadores Incrementais

Sorocaba
2026

Rafael Tofoli Sereicikas

Representação Vetorial Consciente de Tempo para Recomendadores Incrementais

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências Exatas e de Tecnologia da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Metodologias e Técnicas de Computação

Orientador: Prof. Dr. Tiago A. Almeida

Sorocaba
2026

Sereicikas, Rafael Tofoli

Representação vetorial consciente de tempo para recomendadores incrementais / Rafael Tofoli Sereicikas - - 2026.
120f.

Dissertação (Mestrado) - Universidade Federal de São Carlos, campus Sorocaba, Sorocaba
Orientador (a): Tiago Agostinho de Almeida
Banca Examinadora: Marcelo Garcia Manzato, Alan Demétrius Baria Valejo
Bibliografia

1. Modelagem temporal. 2. Sistemas de recomendação. 3. Representação vetorial. I. Sereicikas, Rafael Tofoli. II. Título.

Ficha catalográfica desenvolvida pela Secretaria Geral de Informática (SIn)

DADOS FORNECIDOS PELO AUTOR

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

Folha de Aprovação

Defesa de dissertação de mestrado do(a) candidato(a) Rafael Tofoli Sereicikas, realizada em 01/04/2026

Comissão Julgadora

Prof(a) Dr(a) Tiago Agostinho de Almeida (UFSCar)

Prof(a) Dr(a) Marcelo Manzato (USP)

Prof(a) Dr(a) Alan Valejo (UFSCar)

O relatório de defesa assinado pelos membros da comissão Julgadora encontra-se arquivado junto ao Programa de Pós-Graduação em Ciência da Computação

Dedico este trabalho aos meus pais, Ricardo e Rosana, com todo o meu amor e gratidão.

Agradecimentos

Sou imensamente grato a toda minha família, em especial aos meus pais, Ricardo e Rosana, pelo apoio inestimável em todas as etapas da minha vida.

À Anna Julia, minha parceira de mais de dois anos, por todo o carinho, a companhia e a paciência durante todo este processo.

Aos meus amigos sorocabanos, agradeço pelos anos de companheirismo e por sempre me fazerem sentir tão bem acolhido. Aos meus amigos ituanos, agradeço pela amizade de uma vida inteira, que tornou este trabalho muito mais leve.

A todos os integrantes do laboratório de pesquisas da UFSCar, por tornarem o ambiente de trabalho tão agradável. Em especial ao Pedro, Gregório e Pietro, meus colegas de projeto do LaSID, pelas inúmeras reuniões, discussões e apoio nos experimentos, este projeto não teria a mesma qualidade, e nem seria possível, sem vocês.

Ao meu orientador, Tiago Almeida, que me incentivou a fazer o mestrado e que depositou todo o apoio e confiança em mim durante este projeto.

À UFSCar, onde tive o privilégio de concluir tanto a graduação quanto esta pesquisa, por me proporcionar uma formação acadêmica de excelência.

Por fim, este trabalho foi realizado com o apoio da CAPES. Agradeço especialmente à concessão da bolsa nº 88887.854357/2023-00, que viabilizou a dedicação integral a esta pesquisa por dois anos.

"Ideias são como peixes. Se quiser pescar peixes pequenos, pode ficar na água rasa. Mas se quiser pescar o peixe grande, precisará ir mais fundo. No fundo, os peixes são mais poderosos e mais puros. São enormes e abstratos. E eles são muito bonitos."

(David Lynch)

Resumo

Com o crescimento do volume de dados, os sistemas de recomendação evoluíram de abordagens estáticas em lote para modelos incrementais, permitindo que agentes inteligentes tratem a recomendação como um processo dinâmico de decisão e aprendam de forma contínua com o feedback do usuário. Contudo, ao priorizar etapas como a otimização de política e a modelagem de recompensas, a literatura atual negligencia importantes aspectos destes sistemas, como representação de contexto e dinâmicas temporais de consumo, fatores intrinsecamente relacionados ao cenário da recomendação contínua. Com base nessa lacuna, este trabalho introduz o TAI2Vec, um modelo de geração de embeddings de itens que integra variáveis temporais para criar representações latentes adaptáveis a diferentes ritmos de consumo. Através de uma extensa avaliação experimental, abrangendo recomendações por similaridade e algoritmos de aprendizado incremental baseados em *bandits*, demonstra-se que o TAI2Vec supera métodos tradicionais em tarefas de recomendação, especialmente em cenários com escassez de dados, evidenciando a capacidade que informações temporais têm em enriquecer o aprendizado contextual em diferentes abordagens da recomendação. O código-fonte do TAI2Vec está disponível publicamente em: https://github.com/UFSCar-LaSID/tai2vec.

Palavras-chave: Sistemas de recomendação. Aprendizado por reforço. Consciente do tempo. Dados temporais. Representação Distribuída de Vetores. Ranqueamento.

Abstract

With the growth of data volume, recommendation systems have evolved from static batch approaches to incremental models, allowing intelligent agents to treat recommendation as a dynamic decision process and learn continuously from user feedback. However, by prioritizing stages such as policy optimization and reward modeling, the current literature neglects important aspects of these systems, such as context representation and temporal consumption dynamics, factors intrinsically related to the continuous recommendation scenario. Based on this gap, this work introduces TAI2Vec, an item embedding generation model that integrates temporal variables to create latent representations adaptable to different consumption rhythms. Through an extensive experimental evaluation, covering from similarity recommendations to incremental learning algorithms based on bandits, it is demonstrated that TAI2Vec surpasses traditional methods in recommendation tasks, especially in scenarios with data scarcity, evidencing the capacity that temporal information has in enriching contextual learning in different recommendation approaches. The source code for the TAI2Vec methods is publicly available at: https://github.com/UFSCar-LaSID/tai2vec.

Keywords: Recommender systems. Reinforcement learning. Time-aware. Temporal data. Distributed vector representation. Ranking.

Lista de ilustrações

Lista de tabelas

Sumário

Capítulo 1

Introdução

Com o surgimento da era digital e o crescimento acelerado do volume de dados disponíveis, sistemas de recomendação se tornaram uma ferramenta cada vez mais presente nos meios digitais. Desempenhando um papel fundamental na personalização das experiências dos usuários em diversos ambientes, desde serviços de *streaming* até comércio eletrônico, sistemas de recomendação são um conjunto de algoritmos capazes de identificar padrões de comportamento de um usuário, com o objetivo de fornecer sugestões personalizadas de novos itens, produtos, serviços ou qualquer tipo de informação que possa ser de seu interesse.

As recomendações costumam funcionar sob o princípio de que existem relações significativas entre interações centradas em usuários e produtos. Por exemplo, um usuário que está interessado em documentários de história pode estar mais pendente de assistir a outros documentários do que a um filme de ação. Um outro exemplo seria a suposição de que usuários com comportamentos similares possuem gostos similares, sendo assim, possível realizar predições a partir de conexões com os usuários considerados mais semelhantes (AGGARWAL, 2016).

As abordagens tradicionais de recomendação dividem-se majoritariamente entre filtragem colaborativa, que explora padrões de similaridade entre usuários para prever preferências, e recomendação baseada em conteúdo, que sugere itens com atributos semelhantes aos consumidos anteriormente pelo usuário (EKSTRAND, 2011; KO et al., 2022). Embora eficazes, estas técnicas enfrentam limitações estruturais: enquanto a filtragem colaborativa sofre com a alta esparsidade e dimensionalidade das matrizes de interação, a baseada em conteúdo tende a reduzir excessivamente o escopo das sugestões, limitando a descoberta de itens divergentes do padrão histórico do usuário e reduzindo a novidade nas recomendações. Como resposta, diferentes abordagens e métodos foram propostos, como os sistemas híbridos, que combinam ambas as estratégias para melhorar a qualidade das sugestões, e as técnicas de fatoração de matrizes, que decompõem as interações em espaços latentes de menor dimensão para contornar a

alta demanda de memória de matrizes esparsas (TRABELSI; KHTIRA; ASRI, 2021; KOREN; BELL; VOLINSKY, 2009). Mais recentemente, a evolução da área permitiu a inclusão da filtragem com reconhecimento de contexto, que extrapola a relação binária entre usuários e itens ao integrar variáveis dinâmicas do ambiente, como localização, momento do dia e atividade atual do usuário (MA et al., 2011).

Mesmo com avanços promissores, a maioria dos sistemas de recomendação convencionais ainda é estática e não consegue se adaptar a desvios comportamentais e tendências emergentes em tempo real. Diferentemente dessa abordagem, sistemas interativos e técnicas de recomendação incremental, como o aprendizado por reforço ou os *multi-armed bandits*, permitem que o sistema aprenda de forma sequencial ao longo da interação do usuário (BAGHI et al., 2021). Nos últimos anos, esta área de pesquisa presenciou um crescimento exponencial de estudos (AFSAR; CRUMP; FAR, 2021).

Nesse paradigma incremental, a eficácia do sistema depende da precisão com que o ambiente e o comportamento do usuário são codificados. No entanto, enquanto a maioria dos estudos foca no design de algoritmos, a estrutura de representação do estado é frequentemente negligenciada (LIU et al., 2020b). Modelos convencionais, baseados em matrizes de relacionamento, tornaram-se inviáveis diante do crescimento exponencial de dados, enfrentando sérios gargalos de alta dimensionalidade, esparsidade e falta de escalabilidade (LINDEN; SMITH; YORK, 2003; ZARZOUR; AL-SHARIF; JARARWEH, 2019).

Para superar essas limitações, consolidou-se na área o uso de *embeddings* para transformar objetos em representações de baixa dimensionalidade em um espaço vetorial contínuo (BENGIO et al., 2003). A utilização dessas representações densas é fundamental em algoritmos de Aprendizado por Reforço para lidar com espaços de estados complexos. Em sistemas de recomendação, essa técnica permite que o agente opere em espaços de ação vastos e contínuos, mapeando itens para vetores latentes que capturam semelhanças semânticas de forma eficiente (DULAC-ARNOLD et al., 2015).

Uma forma bem estabelecida de gerar essas embeddings é por meio de redes neurais que processam os últimos itens consumidos (ZHAO et al., 2020). Contudo, embora esses modelos respeitem a ordem de entrada, eles raramente utilizam outras informações temporais importantes durante o treinamento (LATHIA; HAILES; CAPRA, 2009; VINAGRE; JORGE; GAMA, 2015). Mesmo em arquiteturas de aprendizado incremental, a dimensão temporal costuma ser reduzida a janelas deslizantes (CHEN et al., 2019b), ignorando intervalos exatos entre ações ou atributos contextuais como o momento do dia, elementos que, embora promissores (CAMPOS; DÍEZ; CANTADOR, 2014; PAVLOVSKI et al., 2020), raramente são integrados ao treinamento dos vetores latentes.

Essa limitação persiste mesmo quando o fator tempo é explicitamente considerado. Trabalhos anteriores tentaram integrar a dinâmica temporal por meio de representações de estado específicas (PIRES; PASCON; ALMEIDA, 2021), mas geralmente dependem de heurísticas globais, como janelas fixas ou funções de decaimento universais. O problema central dessas

abordagens reside na negligência da heterogeneidade comportamental: um intervalo de 24 horas pode significar o fim de uma sessão para um usuário, mas apenas uma breve pausa para outro. Ao desconsiderar esses ritmos individuais, os modelos geram representações que falham em refletir fielmente a evolução das preferências do usuário no tempo.

1.1 Lacunas de Pesquisa

Com base no cenário atual da literatura de sistemas de recomendação incrementais, foram identificadas as seguintes lacunas de pesquisa:

L1. Limitação de Conhecimento Temporal em *Embeddings*: Modelos tradicionais de representação vetorial, como o Item2Vec e o Prod2Vec, focam na ordem sequencial e na coocorrência das iterações, mas ignoram informações temporais explícitas, como o intervalo de tempo exato entre as interações. Assim, o tempo é tratado apenas como uma sequência ordenada, e não como um contexto rico em informação.

L2. Rigidez de Regras Temporais Fixas: O uso de janelas deslizantes ou funções de decaimento universais falha ao ignorar a heterogeneidade comportamental dos usuários, não diferenciando ritmos de consumo distintos e falhando em oferecer uma experiência personalizada.

L3. Dificuldade de Representação de Estados Dinâmicos: A maioria das pesquisas em sistemas de recomendação incrementais foca na proposta de novos algoritmos ou na otimização da política de decisão e exploração, negligenciando o desenvolvimento de estados que melhor capturem as preferências do usuário em um espaço latente.

L4. Ausência de um protocolo experimental consolidado para recomendadores incrementais: Não há padronização na forma como algoritmos de recomendação incrementais devem ser avaliados de forma *offline*. Adicionalmente, o uso de simuladores não preencheu de forma satisfatória a lacuna entre a segurança da avaliação *offline* e a fidelidade dos testes *online*, permanecendo complexos e restritos a domínios específicos.

L5. Viés de Avaliação contra a Exploração: Protocolos de avaliação *offline* penalizam estratégias exploratórias por dependerem de registros históricos estáticos, o que dificulta a validação de benefícios de longo prazo como diversidade e descoberta.

1.2 Hipótese e objetivos

Este projeto de pesquisa tem como hipótese principal: "*A integração de fatores temporais na representação de estados de recomendação baseados em aprendizagem incremental pode resultar em recomendações mais adaptativas e personalizadas, melhorando a qualidade das recomendações ao longo do tempo.*"

O objetivo geral deste trabalho é desenvolver e avaliar novos modelos de representação vetorial que integrem informações temporais explícitas para alimentar sistemas de recomendação incrementais. Nesse sentido, este trabalho introduz o Time-Aware Item-to-Vector (TAI2Vec), uma família de modelos de *embeddings* leves que incorpora a proximidade temporal adaptativa ao usuário diretamente no processo de aprendizado de representação. Diferente de métodos que impõem limiares globais arbitrários, o TAI2Vec ajusta dinamicamente a definição de "contexto" ao ritmo comportamental de cada indivíduo. Desta forma, esta pesquisa propõe duas frentes metodológicas:

- **Segmentação Discreta (TAI2Vec-Disc):** método que utiliza limiares estatísticos personalizados para detectar fronteiras naturais de sessão e segmentar o histórico de usuário em diversos grupos temporais, garantindo que a coocorrência seja reforçada dentro de itens pertencentes a grupos contextualmente relevantes.

- **Decaimento Contínuo (TAI2Vec-Cont):** método que utiliza o desvio padrão individual para modelar o decaimento suave do interesse ao longo do tempo, ponderando as relações semânticas de acordo com o afastamento estatístico de cada interação em relação ao ritmo habitual de consumo do usuário.

Como suporte a essas propostas, o trabalho realiza uma extensa avaliação experimental fundamentada no desenvolvimento de protocolos experimentais *offline*. Como objetivos secundários, são estabelecidos fluxos de avaliação que padronizam o treinamento de algoritmos de *bandits* em um contexto de dados sequenciais e contínuos. Complementarmente, são propostas adaptações na plataforma de simulação KuaiSim para avaliar a qualidade dos modelos sob métricas de feedback imediato, demonstrando o potencial desses simuladores e advogando por sua utilização como alternativa para mitigar os vieses de registros históricos estáticos.

1.3 Perguntas de pesquisa

Com base nas lacunas de pesquisa e nos objetivos apresentados, este trabalho busca responder quatro perguntas de pesquisa relacionadas à etapa de geração de *embeddings* e ao ambiente experimental de recomendadores incrementais:

P1. Em que medida a integração explícita de intervalos temporais entre interações durante o treinamento de modelos de embeddings supera o desempenho de modelos puramente estáticos em métricas de recomendação Top-N?

P2. Como a modelagem dinâmica de contextos temporais personalizados ao comportamento individual dos usuários impacta a qualidade das recomendações em comparação a métodos gerais que utilizam janelas de tempo globais e estáticas?

P3. Protocolos experimentais *offline* tradicionalmente utilizados na literatura para avaliação de recomendadores incrementais são capazes de mensurar com assertividade a evolução do aprendizado sob um fluxo contínuo de dados?

P4. Como a adaptação de simuladores de larga escala permite a integração de novas representações vetoriais para uma avaliação versátil e livre de vieses de seleção?

1.4 Resultados e Contribuições

Esta dissertação oferece contribuições tanto no campo de *embeddings* temporais quanto na metodologia de avaliação para sistemas de recomendação incrementais. Os principais resultados e contribuições podem ser sintetizados nos seguintes pontos:

- ❑ **Proposta do Modelo TAI2Vec:** A principal contribuição técnica foi o desenvolvimento de uma nova família de modelos de embeddings que superam a rigidez de janelas temporais fixas. Ao introduzir o desvio padrão individual e limiares estatísticos personalizados para definir "contexto", o modelo apresentou maior robustez em cenários de dados esparsos e de consumo orgânico, superando os *baselines* Item2Vec e Seq2Vec em métricas de acerto e ordenação na maioria das bases de dados avaliadas.

- ❑ **Integração entre *Embeddings* temporais e Multi-Armed Bandits:** O trabalho avalia a utilização de representações vetoriais latentes enriquecidas com informação temporal como contexto para algoritmos de *contextual multi-armed bandits* (tais como o LinUCB e o LinGreedy). Resultados obtidos indicam que a qualidade das *embeddings* atua como um facilitador para a convergência do aprendizado incremental, permitindo que o agente identifique preferências de curto prazo com maior agilidade.

- ❑ **Protocolo de Avaliação Incremental Reutilizável:** Frente à carência de protocolos padronizados, este trabalho detalha uma metodologia offline baseada no regime *test-then-train* com janelas deslizantes, oferecendo uma boa alternativa para pesquisadores que buscam validar algoritmos incrementais antes da implantação em produção.

- ❑ **Adaptação de Simuladores Orientados a Dados:** O simulador KuaiSim foi reestruturado para suportar a troca de arquiteturas de embeddings utilizadas na construção do estado. Ao treinar o simulador e o agente sob o mesmo espaço semântico, este trabalho demonstra como nuances comportamentais, como a diferença entre feedback imediato e retenção de longo prazo, que seriam invisíveis em avaliações estáticas tradicionais.

- ❑ **Análise da Integridade Temporal:** O estudo identifica e documenta como a eficácia de modelos sensíveis ao tempo é dependente da natureza da coleta de dados. A análise comparativa entre bases como MovieLens (ruidosa) e Amazon Reviews (orgânica) fornece um guia sobre as condições nas quais a modelagem temporal dinâmica é indicada, alertando para os limites de causalidade em conjuntos de dados específicos.

1.5 Organização do Trabalho

Este trabalho está estruturado em sete capítulos. O Capítulo 2 estabelece a fundamentação teórica sobre sistemas de recomendação e aprendizado incremental. O Capítulo 3 contextualiza o estado-da-arte em recomendações conscientes de tempo e técnicas de representação vetorial de estados. O Capítulo 4 detalha a proposta metodológica, descrevendo o desenvolvimento das abordagens TAI2Vec-Disc e TAI2Vec-Cont para geração de embeddings temporais. O Capítulo 5 apresenta a avaliação experimental, confrontando os métodos propostos em cenários estáticos, incrementais e simulados para permitir uma análise comparativa abrangente. Por fim, o Capítulo 6 encerra o trabalho com as conclusões, reflexões finais, e sugestões para pesquisas futuras.

Capítulo 2

Fundamentação teórica

Neste capítulo, serão abordados os conceitos fundamentais de duas áreas da inteligência artificial nas quais este projeto se concentra: sistemas de recomendação e aprendizado incremental. As próximas seções buscam fornecer uma compreensão abrangente dos princípios básicos dessas áreas, que serão explorados nos capítulos subsequentes.

2.1 Sistemas de recomendação

Os sistemas de recomendação emergiram como uma área de pesquisa e desenvolvimento proeminente, impulsionando significativamente a capacidade de oferecer sugestões personalizadas aos usuários (BOBADILLA et al., 2013). Desde o seu surgimento em meados dos anos 90, proveniente das pesquisas na área de filtragem colaborativa (HERLOCKER; KONSTAN; RIEDL, 2002), sistemas de recomendação se tornaram uma ferramenta fundamental para a tomada de decisões mais informativas, eficientes e eficazes em praticamente todos os aspectos diários de nossas vidas, como trabalho, estudos, entretenimento e socialização (WANG et al., 2021).

Em linhas gerais, os recomendadores atuam sobre a chamada matriz de interação, que registra o histórico de conexões entre dois grupos principais: os usuários e os itens (produtos, filmes, músicas, etc.), podendo ser representada de forma esparsa ou densa. Essa matriz funciona como um mapa de consumo que, quando um usuário acessa ou avalia um item, a relação é registrada; caso contrário, o espaço permanece vazio. O objetivo central do sistema é justamente "prever" o que deveria estar nesses espaços vazios, identificando quais itens teriam maior probabilidade de agradar cada perfil de usuário.

As recomendações também podem atuar sobre cenários explícitos e implícitos. No cenário explícito, o sistema lida com feedbacks diretos, como notas de 1 a 5 ou curtidas, onde

a opinião do usuário é claramente manifestada. Já no cenário implícito, o sistema analisa comportamentos indiretos, como o histórico de cliques, tempo de visualização ou compras realizadas, sem que haja uma avaliação formal. Com isso, dependendo da natureza desses dados e das necessidades do negócio, o sistema geralmente se orienta por dois objetivos principais: a predição de nota ou a geração de listas Top-N (WANG et al., 2021).

No primeiro caso, comum em cenários de feedback explícito, o sistema tenta prever com a maior precisão possível qual seria a nota de um usuário sobre um item que ele ainda não consumiu, onde o objetivo é minimizar o erro desta predição para todos os itens no catálogo. Por outro lado, a abordagem Top-N é frequentemente a escolha para cenários de feedback implícito, como no caso desta dissertação. Em vez de se preocupar com o valor exato de uma nota, o objetivo é a classificação e o ordenamento. O sistema processa todo o catálogo disponível para selecionar os N itens com maior probabilidade de consumo e os apresenta em uma sequência de relevância decrescente, garantindo que as sugestões de maior impacto ocupem as primeiras posições na interface apresentada ao usuário.

Os sistemas de recomendação podem ser classificados em várias categorias com base em seus algoritmos e abordagens. Os principais tipos de sistemas de recomendação incluem filtragem colaborativa, filtragem baseada em conteúdo e abordagens híbridas que combinam várias técnicas (BURKE, 2002).

Os algoritmos de vizinhança, pioneiros da filtragem colaborativa, atuam sob um processo de três etapas (HERLOCKER; KONSTAN; RIEDL, 2002): *(i)* o sistema compara o perfil de consumo do usuário alvo com outros usuários que interagiram com os mesmos itens através de uma matriz esparsa que armazena todos os relacionamentos usuário-item, calculando semelhanças entre esses usuários a partir de métricas pré-estabelecidas; *(ii)* os usuários mais parecidos são agrupados em um subconjunto chamado de "vizinhança" do usuário alvo. Essa vizinhança é essencialmente um grupo de usuários com preferências semelhantes às do usuário alvo; *(iii)* com base nas informações obtidas sobre os usuários desta vizinhança, um algoritmo é usado para criar uma lista top-N de recomendações, que consiste em N itens ordenados que sejam relevantes para o usuário. De forma similar, é possível gerar recomendações com base em uma vizinhança construída de acordo com itens similares (LINDEN; SMTITH; YORK, 2003).

Abordagens baseadas em fatoração de matrizes (do inglês, *Matrix Factorization* – MF) se mostraram precisas e escaláveis ao lidar com problemas de filtragem colaborativa (LUO et al., 2014). A ideia por trás de um método de MF funciona através da aproximação de uma matriz R como o produto de duas matrizes:

$$\mathbf{R} \approx \mathbf{P} \cdot \mathbf{Q} \tag{1}$$

onde $\mathbf{P}$ é uma matriz $N \times K$ e $\mathbf{Q}$ é uma matriz $K \times M$, nas quais N representa a quantidade de usuários, M a quantidade de itens, e K a dimensionalidade dos atributos de cada matriz.

O objetivo principal de modelos de fatoração é gerar uma representação numérica de baixa dimensionalidade de ambos os usuários e itens (TAKÁCS et al., 2008). Um modelo desse

tipo está intimamente relacionado à Decomposição de Valor Singular (do inglês, *Singular Value Decomposition* – SVD), uma técnica bastante estabelecida para identificar fatores semânticos latentes na recuperação de informações. No entanto, a aplicação deste método frequentemente apresenta dificuldades devido à alta quantidade de valores ausentes causados pela dispersão na matriz de avaliações usuário-item. O SVD convencional não é aplicável quando a matriz possui valores ausentes, além disso, lidar apenas com as entradas conhecidas é altamente propenso a *overfitting* (KOREN; BELL; VOLINSKY, 2009).

Embora a área tenha experimentado melhorias significativas com métodos de fatoração de matrizes em seus estágios iniciais, a ascensão das redes neurais artificiais (do inglês, *Artificial Neural Networks* – ANN) trouxe mudanças consideráveis para os sistemas de recomendação (ZHANG et al., 2019). As redes neurais profundas, em particular, aprimoraram a capacidade de aprendizado de representações de características, permitindo a captura de interações complexas entre os itens e os usuários. Além disso, estes métodos podem lidar com dados mais complexos, diversificados (ZHANG et al., 2019), e representar usuários e itens como vetores distribuídos de baixa dimensionalidade (BARKAN; KOENIGSTEIN, 2016).

Nos últimos anos, avanços significativos vêm sendo realizados em métodos de aprendizagem profunda para sistemas de recomendação, superando as limitações de modelos tradicionais e resultando em recomendações de alta qualidade. A aprendizagem profunda possibilita capturar relações não lineares entre usuários e itens, sendo também capaz de identificar as relações intrínsecas dos próprios dados, considerando informações contextuais, textuais e visuais disponíveis (CHENG et al., 2016).

Tradicionalmente, os sistemas de recomendação lidam com aplicações que têm apenas dois tipos de entidades – usuários e itens – e não os inserem em um contexto ao fornecer recomendações (ADOMAVICIUS; TUZHILIN, 2015). Desta forma, recomendações de filtragem baseada em contexto (do inglês, *Context-aware*) ganharam destaque e vem apresentando um progresso significativo. As principais evoluções têm sido direcionadas para a integração eficaz de informações contextuais no processo de recomendação (ADOMAVICIUS; TUZHILIN, 2015). Vários fatores contextuais, como hora, local, clima, contexto social e comportamento do usuário, são levados em consideração para estimar as preferências do usuário e a relevância do item. A importância de incluir contexto no processo de recomendação pode ser observada a partir de um ponto de vista prático por meio de um exemplo simples: embora um usuário possa adorar esquiar, recomendar a ele um resort de esqui durante o verão é ineficaz. Isso pode ser prejudicial para a confiança do usuário no sistema, se interpretado como uma recomendação "fora de contexto" (CAMPOS; DÍEZ; CANTADOR, 2014).

Abordagens avançadas de modelagem, incluindo a fatoração de matrizes com informações contextuais, Redes Bayesianas e Processos de Decisão de Markov (do ingês, *Markov Decision Process* – MDPs) foram utilizadas para capturar a dinâmica do contexto e as interações entre o item e o usuário, permitindo a geração de recomendações contextualmente relevantes (KULKARNI; RODD, 2020; LIU; ABERER, 2013; SHANIGU; HECKERMAN; BRAF-

MAN, 2005).

2.2 Aprendizado incremental

Mesmo com a grande quantidade de avanços e resultados comercialmente promissores na área, a maioria dos sistemas de recomendação convencionais ainda considera o ato de recomendação como um procedimento estático, independente da estratégia de recomendação adotada. Nesta abordagem, o sistema de recomendação não realiza nenhum tipo de interação com os usuários, e consequentemente não consegue aprimorar o seu modelo de recomendação através do feedback imediato e de longo prazo do usuário. O mesmo não ocorre com sistemas de recomendação interativos, que são capazes de adaptar os seus modelos às preferências do usuário (BAGHI et al., 2021). Dentro do espectro de algoritmos capazes de realizar esse aprendizado dinâmico, duas vertentes principais se destacam na literatura: o Aprendizado por Reforço (*Reinforcement Learning* – RL) e os *Multi-Armed Bandits* (MAB).

2.2.1 Aprendizado por reforço para recomendação

Segundo Gama, Sebastião e Rodrigues (2012), a maioria dos sistemas de recomendação presentes na literatura é tradicionalmente construída e avaliada sobre um cenário estático e em lotes, incluindo técnicas estado-da-arte de fatoração de matrizes e redes neurais. Dessa forma, um algoritmo de recomendação é frequentemente treinado usando uma grande parte das interações entre usuários e itens que foram previamente armazenadas. Sua eficácia em fornecer recomendações é avaliada usando outra parte desse mesmo conjunto de dados. Isto implica que o sistema de recomendação não realiza nenhum tipo de iteração com os usuários, e consequentemente não consegue aprimorar o seu modelo de recomendação através do feedback imediato e de longo prazo do usuário (BAGHI et al., 2021). Em aplicações práticas, um recomendador deve continuamente gerar recomendações de itens para usuários, que imediatamente geram um feedback sobre o que lhes foi apresentado. Sistemas não incrementais, que precisam ser completamente retreinados ao receber novas informações, enfrentam grandes obstáculos para sua aplicação em cenários reais com grande fluxo contínuo de dados (GAMA; SEBASTIÃO; RODRIGUES, 2012). Neste contexto, duas abordagens têm se destacado e sido amplamente empregadas para lidar com o desafio da recomendação interativa: Aprendizado por reforço e *Multi-armed Bandits*.

Aprendizado por reforço (do inglês, *Reinforcement Learning* – RL) é uma área de estudo de aprendizado de máquina que estuda problemas no qual agentes, através de suas interações com o ambiente, aprendem a maximizar uma recompensa numérica. No contexto de sistemas de recomendação, o RL pode ser usado para treinar um agente a realizar recomendações personalizadas a um usuário com base em suas preferências e comportamentos (AFSAR; CRUMP; FAR, 2021). Na maioria das aplicações, para ser solucionado por meio de RL, um problema deve ser formalizado como um MDP (PUTERMAN, 2007), sendo assim uma tupla $M = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$,

na qual:

- $\mathcal{S}$ corresponde a um conjunto de estados;
- $\mathcal{A}$ corresponde a um conjunto de ações;
- $\mathcal{R}$ corresponde a uma função de recompensa $\mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, que gera um ganho negativo, neutro ou positivo para uma ação a tomada em um estado s;
- $\mathcal{P}$ corresponde a uma função de probabilidade $\mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$, que calcula a probabilidade de transição do estado s para o s' dada uma ação a;
- γ corresponde a um fator de desconto, que ajusta a importância de recompensas a curto e longo prazo.

Neste formato de aprendizado, um agente deve aprender uma política $\pi(s)$ que execute uma ação a dentro de um estado s com o objetivo de maximizar a recompensa esperada, descontada por γ, ao longo de um episódio composto por τ passos, sendo calculada da maneira descrita nas Equações 2 e 3 (SUTTON; BARTO, 1998):

$$\max \mathbb{E}[\mathcal{R}(\tau)] \qquad (2) \qquad\qquad \mathcal{R}(\tau) = \sum_{t=0}^{\tau} \gamma^t r(a_t, s_t) \qquad (3)$$

Normalmente, agentes operam por meio da descoberta de uma função-valor Q^π, que calcula um valor para uma tupla de estado e ação e pode ser representado de forma recursiva por meio da equação de Bellman (Equação 4):

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s'} \mathcal{P}(s'|s,a) Q^\pi(s', \pi(s')) \qquad (4)$$

Após o treinamento do agente, a escolha correta de ações que serão realizadas em cada estado com o intuito de resolver a Equação 2 passa a ser a descrita na Equação 5:

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^\pi(s,a) \qquad (5)$$

O paradigma de RL tem sido aplicado com sucesso em várias áreas da inteligência artificial. Recentemente, esse conceito começou a ser explorado na área de recomendação, dando origem ao campo de pesquisa conhecido como Sistemas de Recomendação com Aprendizado por Reforço (do inglês, *Reinforcement Learning Recommender Systems* – RLRS) (AFSAR; CRUMP; FAR, 2021). Os aspectos que distinguem diferentes abordagens de RLRS, além dos algoritmos utilizados, estão na forma em como os estados, ações e recompensas são definidos. Ainda assim, todos são construídos de maneira similar à formulação de Shani et al. (2005). Nela, os estados ($\mathcal{S}$) representam o comportamento do usuário no momento em que a recomendação é gerada, construído com base em interações passadas. As ações ($\mathcal{A}$) correspondem às recomendações

feitas pelo sistema. A probabilidade de transição ($\mathcal{P}$) reflete o feedback do usuário, indicando se a recomendação foi ou não consumida, tornando o ambiente não-determinístico. A função de recompensa ($\mathcal{R}$) atribui um ganho positivo se a recomendação for aceita e 0 caso contrário. Finalmente, o estado seguinte é construído com base no feedback do usuário: $s' = s$ caso o usuário não consuma nenhum item, ou $s' = s + i$, caso o usuário consuma um item i, independente de ter sido recomendado ou não.

O *Q-learning* e o SARSA (SUTTON; BARTO, 1998) são dois algoritmos populares no campo da aprendizagem por reforço que podem ser aplicados em sistemas de recomendação. Ambos modelos representam o estado do sistema através de conjuntos de características dos usuários e seus históricos de interações com itens, enquanto o espaço de ação é definido como o conjunto de itens a serem recomendados. A principal distinção entre essas abordagens reside na forma como atualizam os valores da função Q, que expressa o valor esperado de uma ação em um determinado estado, associando valores a cada par estado-ação possível (SUTTON; BARTO, 1998). Uma evolução notável nesse contexto é o Deep Q Network (DQN) (MNIH et al., 2013), reconhecido como um dos algoritmos de aprendizado por reforço mais bem-sucedidos. O DQN utiliza um *buffer* de replay de experiência para armazenar interações entre o agente e o ambiente. Além disso, os valores Q são aproximados por meio de um modelo de aprendizado profundo, também conhecido como *Q-network*.

No contexto de sistemas de recomendação, recomendar um conjunto de itens por meio de métodos de RL não é uma adaptação tão trivial quanto em outros tipos de algoritmos de recomendação. Por realizarem apenas uma ação, algoritmos tradicionais de RL devem mapear cada possível conjunto de itens como uma ação a ser seguida, resultando em um espaço de ações combinatorial, o que pode ser computacionalmente custoso e muitas vezes impraticável (SUNEHAG et al., 2015). Uma estratégia possível é a de agrupar e recomendar K itens pertencentes a uma mesma vizinhança, como é o caso da arquitetura Wolpertinger (DULAC-ARNOLD et al., 2015), uma técnica que combina RL com algoritmos de vizinhança. Nela, os itens são mapeados para um espaço dimensional de baixa dimensionalidade, e o agente, inspirado no *Deep Deterministic Policy Gradient* (DDPG) de Lillicrap et al. (2015), gera uma "proto-ação", um ponto no espaço vetorial que não corresponde diretamente a uma ação específica. Esta proto-ação é então utilizada como ponto de busca para recomendar os K itens mais similares.

O Wolpertinger apresentou bons resultados comparativos e foi um dos primeiros trabalhos na área a considerar RL dentro de um vasto espaço de ações. No entanto, ainda que alivie o problema de atuar sobre um espaço de ação combinatorial, esta estratégia gera o problema de baixa diversidade das recomendações (IE et al., 2019b). Itens próximos no espaço vetorial tendem a ser similares, e em cenários reais, recomendar itens muito semelhantes pode prejudicar a experiência do usuário (ZHAO et al., 2020). Uma maneira gulosa de contornar este problema é assumir que as recomendações são independentes, e selecionar os K itens que obtiverem maior valor em $Q^{\pi}(s, a)$, como formalmente proposto por Ie et al. (2019b) e empregado em outros estudos (CHEN et al., 2019a). Entretanto, ainda que não restrinja a recomendação apenas

a uma vizinhança de itens, esta abordagem também apresenta a tendência de possuir baixa diversidade. Além disso, a suposição de que recomendações são independentes pode não ser verdadeira em todos os cenários (ZOU et al., 2019).

É importante notar que uma das premissas fundamentais do paradigma de RL tradicional é que a execução de uma ação pelo agente provoca necessariamente uma transição para um novo estado. No entanto, em sistemas de recomendação, essa causalidade nem sempre é observada, especialmente em protocolos de avaliação offline. Mesmo em ambientes dinâmicos, a mudança no estado do usuário pode estar muito mais associada a fatores externos ou ruídos aleatórios do que a recomendação em si. Nesses casos, assumir que o próximo estado s' é uma consequência direta da ação a pode levar a modelos com baixa capacidade de generalização e dificuldades de convergência.

A dificuldade em modelar como um estado transita para outro reforça a escolha pelos Multi-Armed Bandits. Enquanto o RL convencional assume que cada ação altera o estado futuro, os MABs simplificam esse processo ao focar na otimização das recompensas imediatas. Essa característica torna o modelo mais robusto para lidar com a aleatoriedade típica das interações dos usuários.

2.2.2 *Multi-armed bandits* para recomendação

Apresentando-se como uma subclasse dos problemas de aprendizado por reforço, os algoritmos baseados em *Multi-Armed Bandits* (MABs) também se mostraram proeminentes em cenários de recomendação iterativa (BOUNEFFOUF; RISH, 2019). Eles se especializam na seleção de opções entre várias alternativas, conhecidas também como "braços de uma máquina caça-níqueis", ideia de onde surge sua nomenclatura. Cada "braço" representa uma ação possível que o sistema pode realizar, e para cada um deles, é associada uma probabilidade de recompensa desconhecida pelo agente. O objetivo é encontrar a melhor estratégia para maximizar a recompensa total, explorando diferentes opções enquanto aproveita as melhores ações conhecidas até o momento (ZHAO et al., 2019). Formalmente, MABs são um modelo de decisão representados pela tupla $M = \langle \mathscr{A}, \mathscr{R}, Q \rangle$ onde:

- $\mathscr{A}$ corresponde a um conjunto de ações;
- $\mathscr{R}$ corresponde a uma função de recompensa $\mathscr{A} \to \mathbb{R}$ para cada ação;
- Q corresponde a uma função de distribuição de probabilidade.

Nele, o agente deve escolher uma ação $a \in \mathscr{A}$ durante o percurso de T episódios de maneira a maximizar a recompensa cumulativa $\sum_{t=1}^{T} r_t$, sendo que $r_t = \mathscr{R}_t(a_t)$ representa a recompensa adquirida da ação a dentro de um episódio t. O agente opera sobre uma função de distribuição de probabilidade Q sob cada ação a, Q sendo responsável pela escolha de uma ação a ao calcular a sua recompensa esperada.

Para superar as limitações dos algoritmos de MAB tradicionais, que tratam todas as situações de tomada de decisão de forma idêntica, foi proposta a extensão conhecida como *Contextual Bandits* (ou CMAB) (ZHOU, 2016). Nesta abordagem, busca-se melhorar a qualidade das recomendações introduzindo informações que vão além do simples histórico de iterações passadas entre agente e ambiente. O "contexto" é formalizado como um vetor de características n-dimensional, disponível no momento da decisão, que encapsula dados como o perfil do usuário, atributos dos itens e variáveis do ambiente, como o tempo da iteração. Em termos de modelagem, uma das abordagens mais utilizadas é a *Linear Contextual Bandits*. Nela, assume-se a hipótese de que a recompensa esperada de uma ação (ou braço) a depende linearmente do contexto $\mathbf{x}_{t,a}$. Essa relação é expressa por:

$$\mathbb{E}[r_{t,a} \mid \mathbf{x}_{t,a}] = \mathbf{x}_{t,a}^{\top}\boldsymbol{\theta}_a, \tag{6}$$

onde o vetor de parâmetros desconhecido $\boldsymbol{\theta}_a \in \mathbb{R}^d$ é específico para cada braço a. Essa parametrização desacopla a estimativa entre os braços, permitindo estruturas de recompensa heterogêneas. Tipicamente, os parâmetros $\boldsymbol{\theta}_a$ são estimados utilizando mínimos quadrados regularizados (*ridge regression*) independentemente para cada braço.

Para cada $a \in \mathscr{A}$, mantém-se uma matriz $\mathbf{A}_a$ de dimensão $d \times d$ e um vetor acumulador $\mathbf{b}_a \in \mathbb{R}^d$. Inicialmente, $\mathbf{A}_a = \lambda \mathbf{I}_d$ e $\mathbf{b}_a = \mathbf{0}_d$, onde $\lambda > 0$ é o parâmetro de regularização. Quando um braço a_t é selecionado na iteração t com contexto $\mathbf{x}_{t,a_t}$ e a recompensa r_{t,a_t} é observada, as estatísticas são atualizadas da seguinte forma:

$$\mathbf{A}_{a_t} \leftarrow \mathbf{A}_{a_t} + \mathbf{x}_{t,a_t}\mathbf{x}_{t,a_t}^{\top}, \tag{7}$$

$$\mathbf{b}_{a_t} \leftarrow \mathbf{b}_{a_t} + r_{t,a_t}\mathbf{x}_{t,a_t}. \tag{8}$$

A estimativa pontual do vetor de parâmetros para o braço a é então dada pela solução do problema de regressão:

$$\hat{\boldsymbol{\theta}}_a = \mathbf{A}_a^{-1}\mathbf{b}_a. \tag{9}$$

No entanto, o cálculo dessas estimativas não dita a escolha da ação por si só, tal decisão depende da estratégia adotada. Nesse contexto, a qualidade de um algoritmo MAB ou CMAB está diretamente relacionada à política utilizada para tratar o dilema de exploração e aprofundamento (SLIVKINS, 2019). Se o agente realiza apenas exploração, ele vai escolher sempre uma ação aleatória, ignorando o conhecimento adquirido em etapas passadas. Por outro lado, caso ele realize apenas aprofundamento, ele vai escolher apenas a ação que maximiza a recompensa esperada, ou seja, seleciona $a_t = \text{argmax}_{a \in \mathscr{A}}(\mathbf{x}_{t,a}^{\top}\hat{\boldsymbol{\theta}}_a)$, apresentando o risco de ficar preso em um ótimo-local (SILVA et al., 2022).

Em propostas mais simples, como o LinGreedy, inspirado no ε-*greedy* (AUER; CESA-BIANCHI, 2002), o equilíbrio entre exploração e aprofundamento é feito a partir de uma esti-

mativa de probabilidade ϵ, que se mantém fixa durante o aprendizado do agente. Neste método, ao selecionar uma ação, há uma probabilidade ϵ de escolher uma ação aleatória (exploração) e uma probabilidade $1-\epsilon$ de escolher a ação com a melhor recompensa estimada até o momento (aprofundamento). Matematicamente, a ação a_t é selecionada como:

$$a_t = \begin{cases} \underset{a \in \mathscr{A}}{\text{argmax}}\ \mathbf{x}_{t,a}^{\top} \hat{\boldsymbol{\theta}}_a, & \text{com probabilidade } 1-\varepsilon, \\ \text{seleção uniforme em } \mathscr{A}, & \text{com probabilidade } \varepsilon. \end{cases} \tag{10}$$

Outra abordagem possível é a utilização da técnica *Upper Confidence Bound* (UCB) sobre o retorno da regressão, como é feito pelo método LinUCB (AUER, 2003). Nele, a exploração é integrada diretamente na estratégia do agente. Ao calcular a recompensa estimada para uma ação, o UCB também leva em conta a incerteza associada à verdadeira média de recompensa dessa ação, proveniente da falta de experiências passadas. O UCB adota uma postura otimista, sempre considerando o valor máximo possível para a média da recompensa dentro do intervalo de incerteza. À medida que mais dados são observados sob uma certa ação, a incerteza diminui gradualmente, e o método se concentra cada vez mais nas ações que parecem ter as maiores recompensas estimadas. O algoritmo seleciona a ação que maximiza o score $p_{t,a}$:

$$p_{t,a} = \mathbf{x}_{t,a}^{\top} \hat{\boldsymbol{\theta}}_a + \alpha \sqrt{\mathbf{x}_{t,a}^{\top} \mathbf{A}_a^{-1} \mathbf{x}_{t,a}}. \tag{11}$$

onde $\alpha \geq 0$ representa um hiperparâmetro que controla o grau de exploração, servindo como um limite superior de confiança na estimativa de recompensa.

Adicionalmente, o LinTS, construído sobre o método *ThompsonSampling* (AGRAWAL; GOYAL,), aplica princípios Bayesianos à exploração. A cada rodada, para cada braço, amostra-se um vetor de parâmetros $\tilde{\boldsymbol{\theta}}_a$ de sua distribuição posterior:

$$\tilde{\boldsymbol{\theta}}_a \sim \mathscr{N}\left(\hat{\boldsymbol{\theta}}_a, \nu^2 \mathbf{A}_a^{-1}\right). \tag{12}$$

O braço a_t é então escolhido de forma gulosa em relação a esses parâmetros amostrados: $a_t = \text{argmax}_{a \in \mathscr{A}}\ \mathbf{x}_{t,a}^{\top} \tilde{\boldsymbol{\theta}}_a$. O hiperparâmetro ν^2 controla a quantidade de exploração, permitindo que o algoritmo balanceie exploração e aprofundamento com base na incerteza atual dos parâmetros.

Levando em consideração todas as suas diferentes estratégias, a utilização de CMABs no contexto de sistemas de recomendação consiste na seleção de itens de um catálogo para recomendar aos usuários de forma a maximizar uma recompensa, esta podendo ser representada por quantidades de cliques ou satisfação do usuário (SILVA et al., 2022). Como algoritmos baseados em CMABs tipicamente envolvem modelos simples, eles funcionam bem em cenários onde o processo de decisão possa ser modelado como uma série de escolhas independentes entre si. Apresentando resultados promissores em cenários como recomendações de artigos (LI et al., 2010a) e seleção de propagandas online (WU; IYER; WANG, 2018).

Estratégias de *Multi-Armed Bandits* e Aprendizado por reforço podem ser consideradas complementares entre si. Os algoritmos baseados em MABs são especialmente vantajosos em cenários onde há a necessidade de tomar decisões rápidas e eficientes diante de múltiplas opções, sendo que eles se concentram na escolha de ações ótimas para maximizar a recompensa imediata, sem a necessidade de considerar o impacto de longo prazo das decisões (SLIVKINS, 2019). Por outro lado, os algoritmos baseados e aprendizado por reforço são mais adequados para situações mais complexas, onde as ações tomadas têm consequências a longo prazo, buscando maximizar a recompensa cumulativa ao longo do tempo (AFSAR; CRUMP; FAR, 2021).

2.2.3 Avaliação de algoritmos incrementais

A avaliação de algoritmos incrementais no cenário de recomendação pode ser conduzida de duas maneiras principais: avaliação online e offline. A avaliação online, frequentemente realizada via Testes A/B, testa a eficácia do sistema interagindo diretamente com usuários em produção, o que garante a métrica mais precisa de performance real (LI et al., 2010b). Porém, a utilização deste método é custosa e sua implementação complexa, além de correr o risco de comprometer a confiança dos usuários no caso do *deploy* de um sistema de recomendação inferior ou falho.

Do outro lado, há a avaliação offline (também conhecida como *replay*), que consome registros (ou *logs*) de interações passadas para reproduzir de forma estática um cenário de avaliação real. Comparada ao teste online, ela apresenta uma implementação mais simples, segura, e versátil–considerando a possibilidade de testar o algoritmo com diferentes fontes de dados. Por essas vantagens práticas, ela acaba sendo a escolha mais comum para quem desenvolve e estuda novos algoritmos de MABs. No entanto, a situação é outra ao analisarmos a fidelidade dos resultados. A predição de próximo item, maneira mais tradicional de se avaliar recomendadores offline, está sujeita a certos vieses por se restringir a *logs* históricos provenientes de uma política anterior intervencional, isto é, que interferiu nas recomendações e escolhas dos usuários. Métodos de *Replay* (LI et al., 2011; ZENG et al., 2016) adotam uma abordagem baseada em rejeição: eles filtram os *logs* históricos, retendo apenas as instâncias onde a ação da nova política coincide com a da política original, fazendo com que apenas os dados que foram observados pelo usuário no momento da recomendação possam ser avaliados.

Essa limitação torna a validação do novo modelo mais desafiadora, especialmente no comportamento exploratório, que requer feedback de ações ausentes nos *logs*. Esse problema é parcialmente contornado pelo uso de *off-policy evaluation* (OPE) (WANG; AGARWAL; DUDIK, 2016), que considera o comportamento passado para quantificar a qualidade de novos modelos. Há diversas propostas na literatura para reduzir o viés em avaliações *off-policy*, como *Inverse Propensity Scoring* (IPS) (WANG; AGARWAL; DUDIK, 2016) e sua extensão *Doubly Robust* (DR) (DUDIK; LANGFORD; LI, 2011), que corrigem a distribuição dos dados históricos com base na razão de importância entre políticas, atribuindo maior peso a eventos raros.

A aplicação desses métodos enfrenta o dilema viés-variância: quanto mais rigorosa é a eliminação do viés, seja pela aplicação de pesos elevados ou pelo descarte de dados, menor se torna a base de informações válida para a avaliação. Ou seja, uma redução agressiva do viés frequentemente infla a variância, resultando em estimativas mais instáveis. Além disso, mesmo em avaliações de larga escala onde a variância não é necessariamente um problema, ainda existe um viés estrutural no dilema aprofundamento-exploração, evidenciado pelo fato de algoritmos puramente gulosos apresentarem desempenho quase indistinguível de estratégias focadas em exploração (PIRES et al., 2025). O que acaba gerando uma equivalência problemática, pois privilegia o retorno imediato e acaba ocultando benefícios importantes da exploração, como a diversidade e a descoberta de novos itens.

Simuladores oferecem uma alternativa robusta às formas de avaliação tradicionais, criando ambientes com usuários simulados que geram feedback em tempo real baseados em regras comportamentais ou dados históricos (SAITO et al., 2021). Alguns exemplos incluem o RecSim (IE et al., 2019a), uma plataforma configurável que permite criar ambientes simulados com flexibilidade na sua modelagem, incluindo ferramentas como um modelo de transição capaz de simular mudanças no comportamento do usuário ao longo do tempo. O RecoGym (ROHDE et al., 2018), focado em um contexto mais específico de recomendação de produtos em publicidade online, modela o comportamento do usuário através da alternância entre sessões de navegação orgânica (e-commerce) e sessões do tipo *bandit* (interação com anúncios). E, mais recentemente, o KuaiSim (ZHAO et al., 2023), que propõe uma abordagem orientada a dados, fundamentada no dataset KuaiRand, para suportar três níveis hierárquicos de tarefas: a recomendação por lista, a otimização da recompensa em uma única seção e, distintamente, a otimização entre sessões, modelando o tempo de retorno e a retenção do usuário. Simuladores preenchem a lacuna entre a segurança offline e a fidelidade online, permitindo validar efeitos de longo prazo e estratégias sequenciais que logs estáticos falham em capturar.

No entanto, Pires et al. (2025) observam que o uso de simuladores ainda é uma alternativa em maturação, limitada pela alta complexidade de modelar fielmente o comportamento humano real. Como as soluções atuais tendem a ser específicas de domínio, a avaliação offline tradicional permanece como a prática dominante, apesar de comprovadamente penalizar estratégias exploratórias e estar sujeita a vieses dos registros históricos.

2.2.4 Síntese

Em resumo, a fundamentação teórica apresentada demonstra a transição dos sistemas de recomendação de modelos estáticos para paradigmas dinâmicos de aprendizado contínuo. Enquanto técnicas tradicionais como filtragem colaborativa e fatoração de matrizes consolidaram a personalização das recomendações, o uso de Aprendizado por Reforço e Multi-Armed Bandits permite que o sistema se adapte em tempo real às mudanças de comportamento do usuário. No entanto, essa evolução exige métodos de avaliação mais sofisticados que superem as limitações provenientes dos vieses de registros históricos estáticos.

Capítulo 3

Contextualização bibliográfica

Este capítulo busca fornecer uma base sólida para o entendimento da literatura referente ao contexto em que o projeto está inserido. A estrutura deste capítulo está organizada em duas seções: uma introdução a sistemas conscientes de tempo e uma apresentação dos conceitos fundamentais relacionados à representação vetorial de estados.

3.1 Recomendação consciente de tempo

É comum na área de recomendação o uso de abordagens estáticas, onde o ajuste do modelo de recomendação não leva em consideração mudanças temporais. No entanto, essa estratégia tem limitações importantes, uma vez que os interesses dos usuários evoluem ao longo do tempo e valiosas informações podem ser perdidas ao desconsiderá-lo (LATHIA; HAILES; CAPRA, 2009; VINAGRE; JORGE; GAMA, 2015). De forma análoga, é sabido também que considerar aspectos temporais pode agregar melhorias na recomendação e gerar uma maior confiança por parte do usuário (BALTRUNAS; AMATRIAIN, 2009). Sistemas de recomendação conscientes de tempo podem ser considerados um tipo especializado de Sistemas Baseados em Contexto, nos quais sua principal característica é o uso de informações de contexto temporal durante o processo de previsão de avaliações. Desta forma, são capazes de fornecer recomendações diferenciadas dependendo do momento da recomendação, de acordo, por exemplo, com as preferências expressas pelos usuários em contextos temporais semelhantes no passado (CAMPOS; DÍEZ; CANTADOR, 2014).

O uso de atributos temporais em sistemas de recomendação foi introduzido nos estudos de Adomavicius e Tuzhilin (2001), que adotaram uma abordagem de *data warehouse* na qual o tempo foi tratado como uma dimensão da recomendação. No mesmo ano, Zimdars, Chickering e Meek (2001) modelaram o problema da recomendação para uma previsão de séries

temporais. No entanto, o assunto só ganhou popularidade a partir de 2009, com o método timeSVD++ (KOREN; BELL; VOLINSKY, 2009) vencendo o desafio da Netflix Prize[1]. O método utilizou informações temporais para melhorar o já existente SVD++, e se consolidou como o estado-da-arte na época. No mesmo ano, Lathia, Hailes e Capra (2009) concluíram que resultados obtidos por algoritmos estáticos em cenários offline podem sofrer queda de qualidade quando os mesmos são aplicados em cenários online e temporais. Consequentemente, diversas propostas temporais surgiram nos anos subsequentes, como, por exemplo, os trabalhos de Matuszyk et al. (2015) e Vinagre, Jorge e Gama (2015).

Existem duas abordagens principais para incorporar atributos temporais em um sistema de recomendação: usar o tempo como sequência, como é feito nos sistemas dependentes de tempo (do inglês, *Time-Dependent Recommender Systems* – TDRS); ou usar o tempo como contexto, como é feito nos sistemas conscientes de tempo (do inglês, *Time-Aware Recommender Systems* – TARS) (VINAGRE; JORGE; GAMA, 2015).

Os TDRS incorporam a informação temporal como uma sequência cronologicamente ordenada, com os comportamentos temporais de um usuário inseridos de forma implícita de acordo com a ordem como os dados são consumidos (VINAGRE; JORGE; GAMA, 2015). Nos TDRS, duas metodologias comuns de interpretação de dados são adotadas (PIRES; PASCON; ALMEIDA, 2021): a primeira é o uso de funções de decaimento, que atribuem, durante a fase de aprendizado do sistema, mais peso às interações mais recentes do que às interações mais antigas (PAVLOVSKI et al., 2020); já a segunda se baseia no uso de janelas deslizantes, nas quais as interações são ordenadas cronologicamente, e apenas uma fatia pré-definida dos dados é utilizada durante o treinamento, com as informações restantes recebendo um peso menor ou sendo ignoradas completamente (VINAGRE; JORGE; GAMA, 2015). Como consequência, é importante observar que interações realizadas consecutivamente, mas entre longos intervalos de tempo, podem não ser relacionadas (PIRES; PASCON; ALMEIDA, 2021), demandando o uso de atributos temporais para alcançar resultados melhores (BALTRUNAS; AMATRIAIN, 2009).

Ao contrário dos TDRS, os TARS consomem características temporais de forma explícita durante o treinamento e recomendação (CAMPOS; DÍEZ; CANTADOR, 2014). Esse tipo de sistema utiliza informações temporais apoiando-se na periodicidade de fenômenos. Dessa maneira, consideram que um usuário possui um padrão de consumo que é repetido, e a recomendação é influenciada pelo momento em que é realizada, com base em atributos como hora, dia de semana, mês do ano, entre outros. As duas estratégias principais dos TARS consistem em utilizar atributos temporais para alimentar a recomendação (BALTRUNAS; AMATRIAIN, 2009; OKU et al., 2006; RENDLE, 2012) e aplicar funções de pesos com base nos intervalos entre interações (DING; LI, 2005). Outro aspecto importante no consumo de atributos temporais é a capacidade de aumentar a diversidade das recomendações, através de etapas adicionais de filtragem de acordo com o período em que a recomendação é realizada, como

[1] Importante competição na área de sistemas recomendação, organizada pela empresa *Netflix* durante os anos de 2006 a 2009: <www.netflixprize.com/rules.html>

mostrado em Panniello, Tuzhilin e Gorgoglione (2014). Encontrar um ponto de equilíbrio entre diversidade e acurácia das recomendações pode melhorar a experiência do usuário com o sistema (ZHAO et al., 2020).

Em resumo, ao contrário das abordagens estáticas, os sistemas conscientes do tempo (TARS) e dependentes do tempo (TDRS) utilizam explicitamente informações temporais para aprimorar a precisão das recomendações e oferecer uma experiência mais personalizada aos usuários. Enquanto os TDRS se concentram na ordem em que as interações ocorreram, os TARS integram aspectos temporais diretamente no processo de recomendação, levando em conta fatores como periodicidade e intervalos entre as interações. Ambas as abordagens são complementares entre si, oferecendo diferentes perspectivas para lidar com a temporalidade dos dados e proporcionar recomendações mais precisas e relevantes (Figura 1).

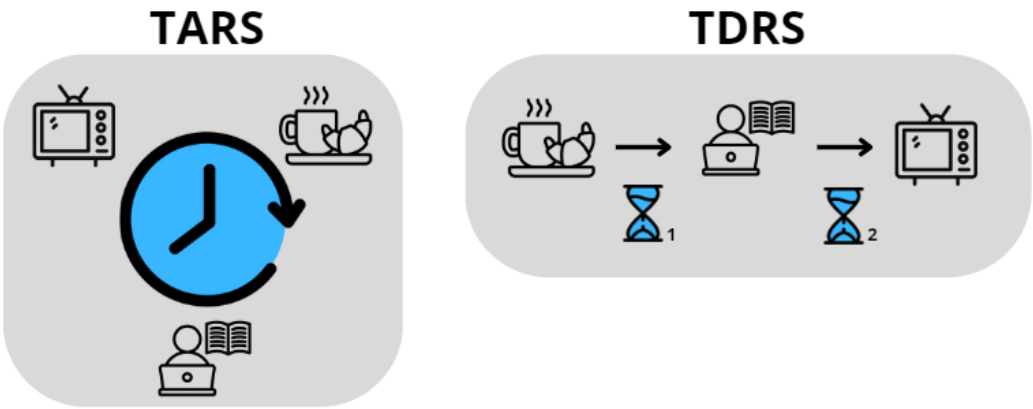


Figura 1 – Comparação entre TARS e TDRS

Embora incorporem conceitos temporais de forma natural em sua formulação (SHANI et al., 2005), a grande maioria dos algoritmos de recomendação baseados em aprendizado por reforço utiliza apenas abordagens sequenciais, aplicando janelas deslizantes sobre as interações dos usuários para construir os estados (AFSAR; CRUMP; FAR, 2021). Com o avanço no uso de redes neurais para aprendizado por reforço e na representação de estados dentro de espaços contínuos, é possível utilizar técnicas já existentes na literatura para se agregar informações temporais nas *embeddings* de representação (PIRES; PASCON; ALMEIDA, 2021).

3.2 Representação vetorial de estados

A maioria dos trabalhos existentes no cenário de recomendação incremental concentra-se no *design* de políticas e algoritmos de aprendizado do agente recomendador, mas raramente se preocupam com a representação do estado do ambiente, que é essencial para a tomada de decisões em recomendações (LIU et al., 2020b). Em recomendadores convencionais, como os sistemas baseados em vizinhança, os estados, podendo ser itens ou usuários, são representados por matrizes de relacionamento M, de dimensão $|U| * |I|$, onde um valor não-nulo na posição M_{ui} indica a ocorrência de interação entre um usuário u e um item i.

Como consequência do aumento exponencial de informações disponíveis, sistemas modernos têm que lidar com uma vasta quantidade de usuários e itens, fazendo com que o espaço vetorial de representação do problema alcance uma alta dimensionalidade, o que prejudica a acurácia de diversos modelos de recomendação (LINDEN; SMITH; YORK, 2003). Além disso, é comum o problema de alta esparsidade, que ocorre devido ao fato de que apenas alguns poucos itens recebem avaliações dos usuários na matriz de relacionamento, resultando em valores na sua maioria nulos (ZARZOUR; AL-SHARIF; JARARWEH, 2019). Por fim, surge também o problema de escalabilidade, quando um sistema opera bem em conjuntos de dados pequenos, mas enfrenta dificuldades para ser eficaz em conjuntos grandes de dados (KHUSRO; ALI; ULLAH, 2016).

Diante desses desafios, o conceito de *embedding* se destaca como uma solução promissora. O termo "*embedding*", cunhado originalmente por Bengio et al. (2003), refere-se a uma representação de baixa dimensionalidade de objetos, como palavras ou documentos, em um espaço vetorial contínuo. Em um cenário de processamento de linguagem natural, por exemplo, cada palavra em um vocabulário pode ser representada como um vetor contínuo de números reais. Esses vetores são construídos de forma a capturar informações semânticas e relacionamentos entre as palavras.

A principal característica das *embeddings* é que itens semanticamente similares são mapeados para vetores próximos uns dos outros no espaço dimensional. Isso significa que palavras com significados semelhantes ou que geralmente aparecem no mesmo contexto apresentam representações vetoriais próximas umas das outras (JATNIKA; BIJAKSANA; SURYANI, 2023). Elas são aprendidas automaticamente a partir dos dados, normalmente de forma não-supervisionada, e permitem que modelos compreendam e manipulem dados de forma mais eficaz ao transformar a informação em uma forma densa e contextualizada.

Uma vez geradas, essas *embeddings* superam as limitações dos métodos de codificação tradicionais, podendo ser utilizadas para fins como: encontrar vizinhos mais próximos, visualizações de dados, análise de relações entre categorias, ou como entrada para algum outro em outro modelo de aprendizado de máquina. Um recomendador baseado em *embeddings* torna mais fácil para que um usuário visualize intuitivamente similaridades entre itens quando comparados a outros métodos de implementação. Além disso, é possível destacar a sua natureza de construção simples e de alta eficiência (JUN et al., 2020).

Técnicas de fatoração de matrizes mais tradicionais como o SVD, elaborado anteriormente no capítulo 2.1, *Alternating Least Squares* (ALS) (TAKÁCS; TIKK, 2012) e *Bayesian Personalized Ranking* (BPR) (RENDLE et al., 2009), são exemplos de métodos não-supervisionados que retornam uma matriz de *embeddings* para usuários e itens a partir de dados históricos de iteração. Estes métodos são capazes de lidar com os problemas citados anteriormente, ao mesmo tempo que geram recomendações de boa qualidade. No entanto, ainda demandam a necessidade de passar por etapas custosas de fatoração de matrizes (SU; KHOSHGOFTAAR, 2009).

3.2.1 *Embeddings* para Processamento de Linguagem Natural

As primeiras pesquisas relacionadas à aplicação de redes neurais para geração de *embeddings* surgiram na área de processamento de linguagem natural, onde a utilização de métodos de codificação de palavras, como *one-hot encoding* e técnicas baseadas em contagem de palavras eram comuns (SEZERER; TEKIR, 2021). Similarmente aos problemas encontrados em sistemas de recomendação, o aumento do vocabulário implicava também em obstáculos substanciais, como dimensionalidade e esparsidade elevada, e a incapacidade de capturar nuances semânticas.

Dentre as diversas soluções propostas na área, as abordagens de representação vetorial de baixa dimensionalidade de palavras (ou *word embeddings*) destacaram-se como uma das mais promissoras (BENGIO et al., 2003). O termo *word embeddings* engloba um conjunto de técnicas voltadas para modelagem de linguagem e extração de características, nas quais palavras ou frases são representadas por vetores de números reais. Esses vetores podem ser empregados em diversas tarefas no processamento de linguagem natural, proporcionando uma representação mais eficiente e significativa. Ainda que já houvessem propostas anteriores, foi com a apresentação do Word2Vec (MIKOLOV et al., 2013a) que as técnica de *embeddings* geradas com redes neurais ganharam notoriedade e se tornaram competitivas com os métodos tradicionais de Fatoração de Matrizes (SEZERER; TEKIR, 2021).

Os modelos Word2Vec produzem representações vetoriais que capturam relações semânticas e são amplamente utilizados em tarefas como recuperação de informações e classificação de texto. Com base no contexto (palavras que se encontram antes ou depois da palavra-chave em um texto), o Word2Vec realiza previsões usando uma dentre duas arquiteturas de rede neural: *(i)* o *Continuous Bag of Words* (CBOW), que recebe como entrada as palavras do contexto (vizinhança) e tenta prever a palavra central alvo; e *(ii)* o *Skip-gram* que opera de maneira inversa: utiliza a palavra central atual como entrada para prever as probabilidades das palavras de contexto dentro de uma janela específica (MIKOLOV et al., 2013b), ambas as arquiteturas podem ser visualizadas na Figura 2.

O treinamento do modelo Word2Vec se baseia na formação de pares compostos pela palavra alvo (w_t) e suas vizinhas de contexto (w_c), que constituem os exemplos positivos ou *ground truth*. A arquitetura opera sobre duas matrizes de pesos inicializadas aleatoriamente: a matriz de entrada W, representando os vetores das palavras alvo, e a matriz de saída W', representando os vetores de contexto, ambas de dimensão $|V| \times d$, onde $|V|$ é o tamanho do vocabulário e d a dimensão do embedding. Nesse processo, a similaridade entre uma palavra alvo e um contexto é determinada pelo produto escalar de seus vetores, $v_{w_t} \cdot u_{w_c}$, valor que é transformado em uma probabilidade por meio da função sigmoide $\sigma(x)$. Um par positivo é definido quando o termo de contexto w_c se encontra dentro da janela de tamanho fixo ao redor do termo alvo w_t na sequência original do texto. Para cada par positivo identificado, a posição vetorial do item alvo é atualizada na direção do vetor de contexto, guiada pelo erro do gradiente. Em contrapartida, as demais palavras que não pertencem ao contexto são tratadas

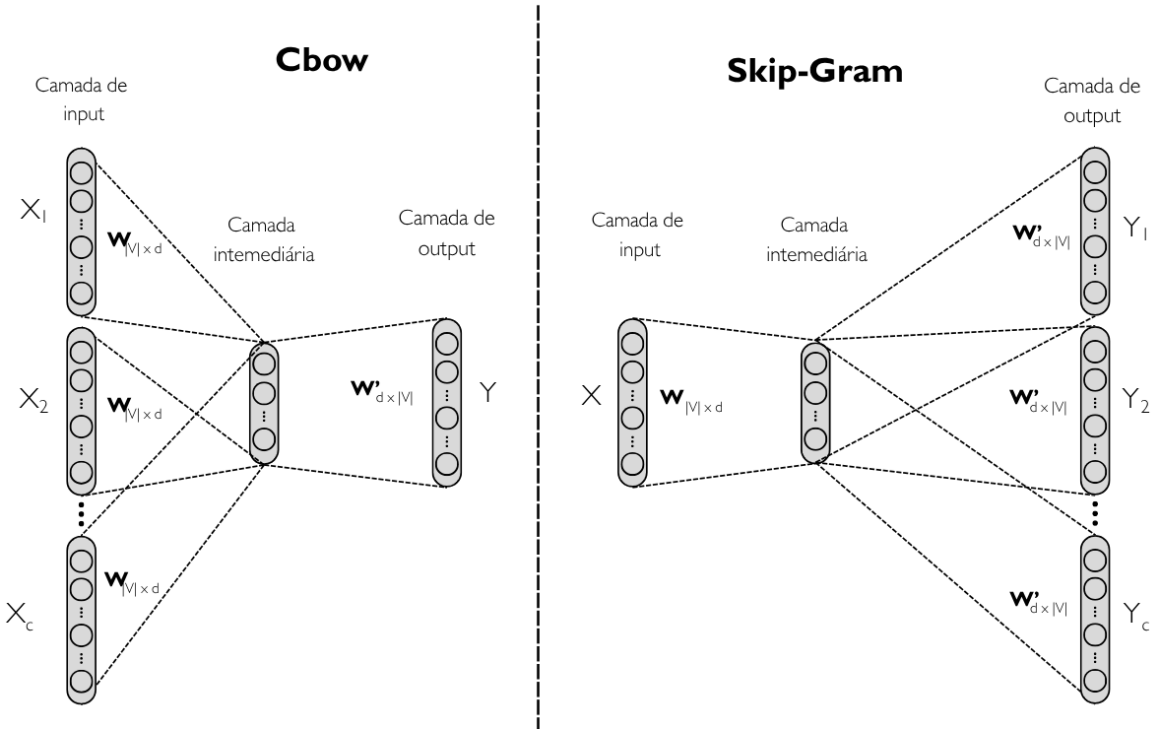


Figura 2 – Diagrama das arquiteturas Cbow e Skip-Gram do Word2Vec.

como exemplos negativos. Nesses casos, o objetivo é minimizar a probabilidade predita de co-ocorrência, ajustando os pesos para penalizar a similaridade e "afastar" o vetor alvo de palavras não relacionadas no espaço vetorial.

Na formulação padrão, a camada de saída utiliza a função *Softmax*. Contudo, sua aplicação direta torna-se computacionalmente intensiva em grandes vocabulários, pois exige o processamento de todos os termos $|V|$ (que podem chegar a milhões) a cada atualização de pesos. Para contornar essa complexidade e viabilizar o treinamento, são implementadas duas estratégias de otimização: *subsampling* e *negative sampling*.

O método de *subsampling* aborda o desequilíbrio na frequência das palavras. Palavras muito frequentes (como artigos e preposições) carregam menos conteúdo semântico revelante do que palavras raras. Para equilibrar o treinamento, palavras frequentes são descartadas probabilisticamente. A probabilidade $P(w_i)$ de descartar uma palavra w_i é dada pela fórmula:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

na qual $f(w_i)$ representa a frequência da palavra no corpus e t é um limiar pré-definido. Isso força o modelo a focar em palavras raras e significativas, acelerando o treinamento e melhorando a qualidade das *embeddings* (SEZERER; TEKIR, 2021).

Visto que o *Softmax* tradicional requer a atualização de pesos de todas as palavras, resultando em uma complexidade proporcional a um vocabulário, o *negative sampling* simplifica esse problema transformando-o em uma série de classificações binárias. O objetivo passa a ser distinguir a palavra de contexto positiva de k palavras de ruído (amostras negativas) que não estão no contexto. A função de custo J para um par de palavra-alvo (w_t) e contexto

(w_c) é definida como a maximização da probabilidade da classe correta e a minimização da probabilidade das amostras negativas:

$$J = \underbrace{\log \sigma(\mathbf{v}_{w_c}^\top \mathbf{v}_{w_t})}_{\text{Termo Positivo}} + \underbrace{\sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)}[\log \sigma(-\mathbf{v}_{w_i}^\top \mathbf{v}_{w_t})]}_{\text{Termo Negativo}}$$

sendo $\sigma(x) = \frac{1}{1+e^{-x}}$ a função sigmoide, $\mathbf{v}_{w_t}$ e $\mathbf{v}_{w_c}$ os vetores da palavra-alvo e do contexto real, e $\mathbf{v}_{w_i}$ os vetores das palavras negativas amostradas da distribuição de ruído $P_n(w)$.

Essa abordagem reduz drasticamente o custo computacional, pois atualiza apenas os pesos da palavra positiva e das k palavras negativas (onde k é tipicamente pequeno), resultando em representações vetoriais de alta qualidade e treinamento veloz (KOREN; BELL; VOLINSKY, 2009). Essas técnicas combinadas não apenas tornam o Word2Vec rápido e eficiente, mas também melhoram a qualidade das representações (KOREN; BELL; VOLINSKY, 2009).

3.2.2 *Embeddings* para sistemas de recomendação

O emprego de redes neurais também se mostrou vantajoso para a área da recomendação, principalmente por possibilitarem a representação de ações e estados em um espaço contínuo, apresentando excelentes resultados em diferentes domínios de aplicação (LILLICRAP et al., 2015). Diversos modelos de sistemas de recomendação usam modelos *Word2Vec* – ou arquiteturas inspiradas nele – para criar representações compactas para os itens de um catálogo. Ao usar esses vetores em métodos de vizinhança, o sistema pode recomendar itens semelhantes com base na proximidade de seus vetores no espaço dimensional reduzido (JUN et al., 2020).

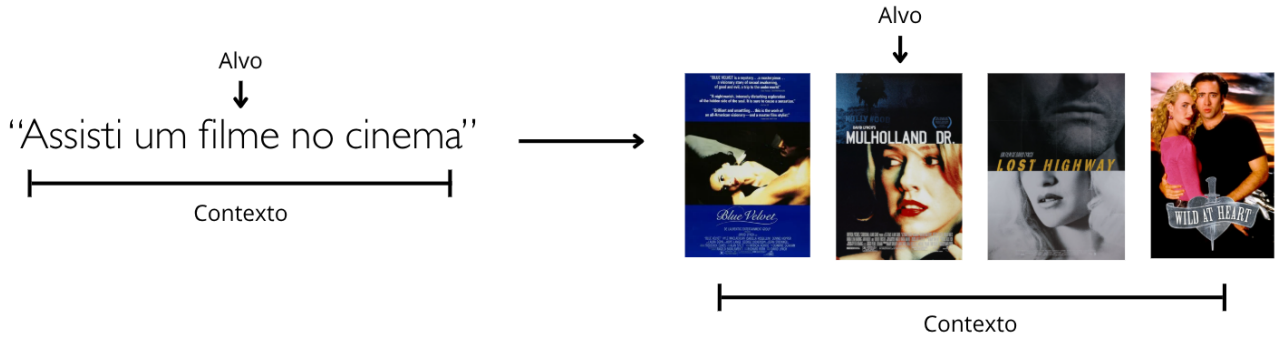


Figura 3 – Da mesma forma que podemos gerar relações semânticas entre palavras vizinhas na mesma frase, também podemos gerar para itens que compartilhem o mesmo histórico de consumo.

O modelo Prod2Vec (GRBOVIC et al., 2015), um dos pioneiros na área, envolve o aprendizado de representações vetoriais de produtos em um ambiente de comércio eletrônico a partir de registros de recibos de e-mail. Emprestando a terminologia do Word2Vec, o modelo considera uma sequência de compras como uma "frase", e produtos dentro da sequência como "palavras", ilustrado na Figura 3. Mais especificamente, o Prod2Vec aprende representações

de produtos usando o modelo Skip-gram. Ele calcula a similaridade de cosseno com todos os outros produtos no catálogo e recomenda os produtos mais semelhantes. O modelo considera todos os produtos comprados como independentes entre si e modela o contexto das sequências de produtos, onde produtos com contextos semelhantes (ou seja, com compras vizinhas semelhantes) terão representações vetoriais semelhantes (GRBOVIC et al., 2015). Em uma extensão deste trabalho, Vasile, Smirnova e Conneau (2016) introduziram o Meta-Prod2Vec, que, durante o cálculo de erro da fase de *backpropagation* do treinamento da rede, também leva em consideração as informações de metadados dos itens, assim adicionando implicitamente essas informações nas *embeddings* resultantes.

No ano seguinte, Barkan e Koenigstein (2016) propuseram o Item2Vec, uma rede neural também baseada no método Word2Vec, capaz de gerar recomendações apenas com o histórico de iterações do usuário. O Item2Vec é capaz de inferir relações item-item mesmo quando informações do usuário não estão disponíveis e produz *embeddings* para itens em um espaço latente. Nessa abordagem, a informação espacial e temporal é descartada ao embaralhar o histórico de consumo no início do treino, assumindo-se que itens compartilhados em um mesmo conjunto são similares independentemente da ordem em que foram gerados. O modelo consistentemente retorna representações melhores para itens menos populares em comparação com o SVD. Além dessas abordagens, Valcarce et al. (2019) propuseram o prefs2vec, que foca na representação de itens para recomendações baseadas em filtragem colaborativa que utiliza a arquitetura Continuous Bag-of-Words ao invés do Skip-gram.

Em um cenário industrial, Grbovic e Cheng (2018) aplicaram embeddings para personalização em tempo real no mecanismo de busca da Airbnb, utilizando também o modelo Skip-gram com amostragem negativa para aprender representações de anúncios a partir de sessões de cliques, integrando-as diretamente ao sistema de ranqueamento. E, mais recentemente, Pires e Almeida (2025) apresentaram o Interact2Vec, um modelo neural aprende simultaneamente representações de usuários e itens no mesmo espaço vetorial, utilizando apenas interações implícitas. Inspirado em arquiteturas como o Item2Vec, o modelo se destaca pela eficiência computacional.

3.2.3 *Embeddings* com conhecimento temporal

No contexto dos trabalhos que propuseram a incorporação do tempo (LATHIA; HAILES; CAPRA, 2009; VINAGRE; JORGE; GAMA, 2015), aprender representações vetoriais que preservam características temporais, independentemente de uma tarefa específica, economizaria esforço redundante e poderia resultar em *embeddings* com qualidade mais alta. Apesar disso, cada aplicação depende de uma engenharia de características especializada que exige muito esforço e geralmente ignora a natureza variável do comportamento do usuário ao longo do tempo.

Diversos métodos foram propostos para a integração de fatores temporais em *embeddings*, apresentando resultados positivos, como é o caso do algoritmo timeSVD++, proposto

por Koren (2010). Ele apresenta uma extensão do algoritmo SVD++ (KOREN, 2008) ao incluir informações temporais nos cálculos dos fatores latentes e utilizar um mecanismo de decaimento temporal para ponderar a importância das interações passada. Outro exemplo é o algoritmo TMF, proposto por Lo et al. (2018), que adapta modelos tradicionais de fatoração de matrizes ao introduzir uma matriz de transição específica do usuário que rastreia a mudança de conceito da preferência do usuário em cada etapa de tempo.

Em relação a modelos baseados em redes neurais, o uso de informações temporais para a segmentação de sessões tem se mostrado uma estratégia eficaz para enriquecer a representação latente. O SeqI2V (PIRES; PASCON; ALMEIDA, 2021) adapta o Item2Vec aplicando janelas deslizantes temporais para separar o histórico do usuário em sessões distintas durante o treinamento, reportando resultados superiores em quase todos os conjuntos de dados avaliados. Nesta linha, o framework STAR (YEGANEGI; HARATIZADEH; EBRAHIMI, 2024) fornece contexto adicional a uma RNN ao utilizar os intervalos de tempo entre interações para identificar descontinuidades e quebrar sessões longas em sub-sessões conceitualmente independentes.

Seguindo por outra estratégia, diversas abordagens buscam capturar a dinâmica temporal de forma contínua, sem necessariamente fragmentar o histórico em sessões discretas. Nesta vertente, Zhang et al. (2023) introduziram o método TAT4SRec, que utiliza informações contínuas de *timestamps* e histórico de itens para criar *embeddings* temporais, os quais são capazes de prever o próximo item que um usuário irá consumir. Mais recentemente, Park et al. (2025) propuseram o TALE, que aplica um sistema de pesos que decai conforme o intervalo de tempo real entre as interações aumenta e utiliza uma normalização dinâmica, que considera a popularidade dos itens apenas em períodos recentes, diminuindo o viés de itens que foram populares apenas no passado.

3.2.4 *Embeddings* para recomendação incremental

Devido à necessidade de estados que sejam densos e de baixa dimensionalidade para um treinamento efetivo, a utilização de *embeddings* tem se destacado como uma escolha popular na modelagem de algoritmos de RLRS (AFSAR; CRUMP; FAR, 2021). Muitos trabalhos de RLRS utilizam seus próprios modelos de geração de *embeddings* acoplados aos agentes de aprendizado em suas abordagens. Em Chang et al. (2019), é proposto um sistema de recomendação híbrido de músicas, que usa uma abordagem baseada em conteúdo para a representação de estados e uma abordagem de aprendizado por reforço para a recomendação em si. Nele, o treinamento das *embeddings* é feito a partir de metadados específicos em cada música, onde informações de áudio são extraídas a partir de um modelo *Wavenet* e informações líricas extraídas a partir de um modelo *Word2Vec*, sendo a representação final do item uma concatenação de ambos os vetores resultantes. É possível citar também o trabalho de Liu et al. (2020c), onde é proposto um método para ajustar dinamicamente os tamanhos de *embeddings* de usuários e itens para melhorar o desempenho e reduzir o consumo de memória. A abordagem envolve uma política de ajuste de tamanhos que utiliza aprendizado por reforço para ajustar adaptativamente os

tamanhos de *embeddings* com base nas frequências de usuário e item durante as iterações no sistema. A Tabela 1 apresenta uma comparação entre os diversos métodos de *embeddings*.

Tabela 1 – Comparação entre diversos algoritmos de *embeddings*

Algoritmo	Modelo	Tipo de Recomendação	Uso de Metadados	Atributos Temporais	Recomendação por Reforço	Referência
SVD	Fat. de Matrizes	Filtragem Colaborativa				Koren, Bell e Volinsky (2009)
ALS	Fat. de Matrizes	Filtragem Colaborativa				Takács e Tikk (2012)
BLR	Fat. de Matrizes	Filtragem Colaborativa				Rendle et al. (2009)
Word2Vec	Rede Neural	Baseada em Conteúdo	✓			Mikolov et al. (2013a)
Prod2Vec	Rede Neural	Filtragem Colaborativa				Grbovic et al. (2015)
Meta-prod2vec	Rede Neural	Híbrido	✓			Vasile, Smirnova e Conneau (2016)
Item2Vec	Rede Neural	Filtragem Colaborativa				Barkan e Koenigstein (2016)
TMF	Fat. de Matrizes	Baseada em Contexto		✓		Lo et al. (2018)
TimeSVD++	Fat. de Matrizes	Baseada em Contexto		✓		Koren (2010)
TASA	Rede Neural	Baseada em Contexto		✓		Pavlovski et al. (2020)
SeqI2V	Rede Neural	Baseada em Contexto		✓		Pires, Pascon e Almeida (2021)
ESAPN	Rede Neural	Aprendizado por Reforço			✓	Liu et al. (2020c)
RPMRS	Rede Neural	Baseada em Conteúdo	✓		✓	Chang et al. (2019)
EDRR	Rede Neural	Aprendizado por Reforço			✓	Liu et al. (2024)

Nesse contexto, as abordagens de aprendizado por reforço propostas que incorporam *embeddings* podem ser generalizadas em um *framework* unificado composto por três módulos principais (LIU et al., 2024). No primeiro módulo, conhecido como Módulo de *Embeddings*, são geradas as representações de baixa dimensionalidade a partir do histórico de interações do usuário. O segundo módulo, Módulo de Representação de Estados, emprega essas representações para modelar o ambiente em um estado contínuo. Nele, os vetores resultantes das interações históricas são integrados em uma rede neural profunda, que explicitamente captura nuances nas interações entre usuários e itens, oferecendo uma descrição mais informativa do estado do usuário (LIU et al., 2020b). Por fim, o Módulo de Aprendizado otimiza as políticas de recomendação do agente ao incorporar a representação contínua do estado do usuário desenvolvida pelo Módulo de Representação de Estados. Dentro do corpo de pesquisa que emprega esse *framework*, é possível citar também os estudos de Liu et al. (2020a) e Zhao et al. (2018), que estruturam suas redes neurais para a geração de *embeddings* de maneira análoga ao *Item2Vec* (BARKAN; KOENIGSTEIN, 2016). Uma representação gráfica do *framework* pode ser observada na Figura 4.

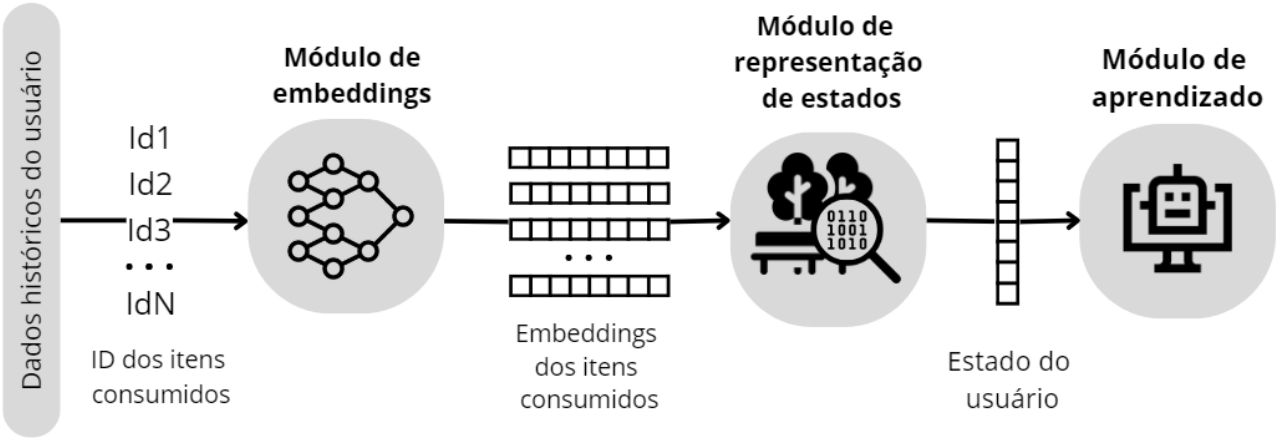


Figura 4 – Representação em alto nível, de autoria própria, do *framework* unificado proposto por Liu et al. (2024)

É importante ressaltar que, embora seja possível treinar *embeddings* de maneira simultânea com o aprendizado do agente recomendador, as abordagens que utilizam *embeddings* pré-treinadas com interações históricas de usuários e itens, mantendo-as inalteradas durante o treinamento do modelo, demonstram, em geral, desempenho superior (*e.g.*, Liu et al. (2024)). Essa constatação ressalta a importância da estabilidade e consistência nas representações geradas, indicando que *embeddings* pré-treinadas podem oferecer benefícios significativos em comparação com métodos que ajustam suas *embeddings* durante o treinamento do agente.

3.3 Considerações finais

Em resumo, os sistemas de recomendação baseados em aprendizado por reforço surgem como uma área de pesquisa promissora, apresentando um crescimento exponencial de estudos nos últimos anos (AFSAR; CRUMP; FAR, 2021). Apesar disso, são poucos os trabalhos que exploram o emprego de dados temporais no treinamento desses recomendadores (VINAGRE; JORGE; GAMA, 2015). Além disso, é consenso que negligenciar informações temporais pode comprometer a qualidade da recomendação final (LATHIA; HAILES; CAPRA, 2009).

Com os avanços no uso das redes neurais artificiais para aprendizado por reforço e na representação de estados em espaços contínuos, torna-se possível empregar técnicas consolidadas na literatura para incorporar informações temporais nas *embeddings* (PIRES; PASCON; ALMEIDA, 2021), proporcionando um valioso acréscimo de informação ao agente recomendador. O aprofundamento de técnicas que preencham estes *gaps* de pesquisa poderão aprimorar a qualidade da recomendação e abrir caminho para o desenvolvimento de sistemas de recomendação mais robustos, eficientes e confiáveis, capazes de se adaptar às preferências dinâmicas dos usuários ao longo do tempo.

Capítulo 4

Embeddings de itens conscientes de tempo

Muitos algoritmos no estado-da-arte em sistemas de recomendação baseados em aprendizado por incremental representam estados como *embeddings*, isto é, vetores contínuos que carregam significado intrínseco e são posteriormente alimentados ao modelo neural responsável pela recomendação. A estratégia mais comum para a criação das *embeddings* consiste no uso de redes neurais, que recebem como entrada os últimos itens consumidos e recomendados ao usuário (ZHAO et al., 2020; ZOU et al., 2019). Entretanto, na maioria dos casos, com exceção da ordem de entrada, nenhuma outra informação temporal é utilizada durante o treinamento.

No método *Skip-gram* do Word2Vec (MIKOLOV et al., 2013a), as *embeddings* são geradas por uma rede neural que recebe a representação *one-hot* de uma palavra como entrada, visando prever o contexto ao seu redor. Para definir este contexto, utiliza-se uma janela deslizante de tamanho N, considerando apenas N palavras à esquerda e à direita do termo em questão (SEZERER; TEKIR, 2021). Como detalhado no Capítulo 3.2.1, para cada palavra dentro dessa janela, geram-se pares alvo-contexto positivos, que são contrastados com uma subseleção de pares negativos, técnica conhecida como amostragem negativa. O objetivo é aproximar vetorialmente palavras semelhantes e distanciar aquelas não relacionadas.

Este modelo pode ser adaptado para a recomendação de itens, como já foi feito através do método Item2Vec (BARKAN; KOENIGSTEIN, 2016). Inspirado pelo sucesso do *Skip-gram* em processamento de linguagem natural, o método trata o histórico de consumo de um usuário como uma "frase" e cada item individual como uma "palavra". No entanto, diferentemente do Word2Vec, essa abordagem descarta a informação sequencial, assumindo um ambiente estático onde todos os itens que compartilham o mesmo conjunto são considerados pares positivos de contexto, independentemente da ordem. Uma visualização da interpretação de "contexto" pode

Figura 5 – No modelo Item2Vec, todos os itens dentro do mesmo histórico de consumo possuem o mesmo peso para o item alvo

ser visto na Figura 5

Desta forma, a incorporação do tempo durante o treinamento baseia-se em uma motivação análoga à adotada por Pires, Pascon e Almeida (2021), onde desenvolveu-se uma estratégia de janela temporal que considera não apenas a ordem, mas o momento das interações, obtendo resultados superiores em diversos conjuntos de dados. Contudo, aquela abordagem apresentou limitações de flexibilidade ao trabalhar com cortes nos dados baseados em um intervalo temporal T fixo, ignorando diferentes comportamentos de consumo entre usuários.

Diante dessas limitações, este trabalho explora a hipótese de que a incorporação explícita de intervalos temporais no treinamento de *embeddings* resulta em vetores mais robustos para recomendação. Para verificar essa hipótese, foram formulados dois métodos: o *Temporal Aware Item2Vec Discrete* (TAI2Vec-Disc), de natureza **Discreta**, baseado na segmentação do histórico em sessões de consumo; e o *Temporal Aware Item2Vec Continuous* (TAI2Vec-Cont), de natureza **Contínua**, que modela o decaimento da influência temporal como uma função suave.

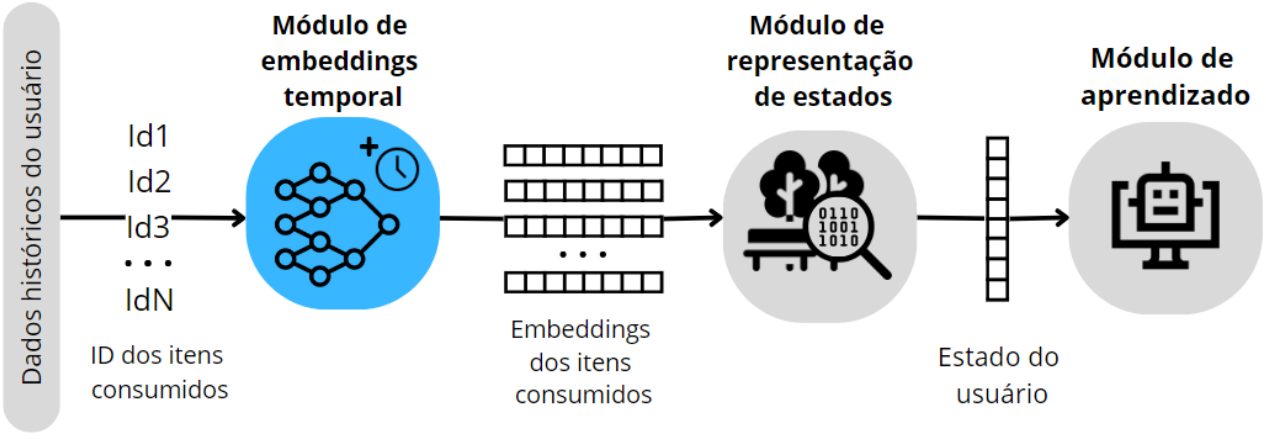


Figura 6 – Representação em visual do *framework* proposto com a introdução do tempo na geração conteúdo das *embeddings*

É importante notar que, como as modificações propostas intervêm exclusivamente na etapa de formação dos pares de treinamento, o custo computacional do TAI2Vec permanece equivalente ao do modelo base. A alteração reside unicamente no critério de seleção do contexto:

Enquanto a abordagem tradicional de janelas ou conjuntos pode exigir uma complexidade de $O(N^2)$ para gerar todos os pares de relação dentro de um histórico de usuário, os métodos TAI2Vec baseiam-se apenas no cálculo de distâncias entre interações, mantendo uma complexidade de $O(N)$ na operação. Consequentemente, a complexidade assintótica final do treinamento não é impactada, garantindo a escalabilidade do modelo.

4.1 Abordagem discreta

A proposta se baseia na segmentação do histórico de consumo de um usuário em diversos "Grupos Temporais". Estes grupos consistem em conjuntos de itens consumidos sob um intervalo de tempo considerado curto, relativo ao comportamento habitual do próprio usuário. A premissa do método baseia-se na constatação de que um usuário pode assistir continuamente uma série de ficção científica e, após um intervalo anormal de comportamento, buscar documentários culinários. Tratar esses dois momentos como um fluxo contínuo e uniforme introduz ruído na modelagem, especialmente em cenários voltados para o aprendizado de representação vetorial de itens. Portanto, esta abordagem busca identificar as fronteiras naturais dessas sessões, assumindo que a relevância semântica é máxima entre itens que compartilham o mesmo bloco temporal e incerta entre blocos distintos. Uma visualização destes grupos pode ser encontrada na Figura 7

Figura 7 – No modelo TAI2Vec-Disc, itens que pertencem ao mesmo grupo temporal possuem maior influência.

Para viabilizar essa análise, o primeiro passo é a ordenação cronológica das interações. Seja H_u o histórico de consumo de um usuário u, composto por uma sequência de N tuplas $(item, tempo)$:

$$H_u = \{(i_1, t_1), (i_2, t_2), \ldots, (i_N, t_N)\}$$

onde a ordenação garante que $t_k \leq t_{k+1}$ para todo k. A partir dessa sequência organizada, define-se o intervalo temporal Δt_k como a diferença de tempo entre uma interação k e sua antecessora imediata:

$$\Delta t_k = t_k - t_{k-1}, \quad \forall k \in \{2, \ldots, N\}$$

O conjunto global de intervalos de transição do usuário, sobre o qual as análises estatísticas serão realizadas, é denotado por $\mathscr{T} = \{\Delta t_2, \Delta t_3, \ldots, \Delta t_N\}$.

Ao contrário de Pires, Pascon e Almeida (2021), a definição das fronteiras dos grupos temporais abandona o uso de limiares estáticos em favor de uma modelagem dinâmica adaptativa. A premissa é que cada usuário (ou domínio de dados) possui um ritmo de consumo intrínseco: o que constitui uma "pausa longa" para um usuário ativo pode ser o intervalo padrão para um usuário esporádico.

Dessa forma, a segmentação é tratada como um problema de detecção de anomalias na distribuição dos intervalos de tempo. O algoritmo considera que a primeira interação inaugura o grupo inicial e monitora o fluxo contínuo de consumo. Um novo grupo é criado somente quando ocorre uma quebra significativa nesse fluxo, ou seja, quando o intervalo Δt entre dois itens excede um limiar de corte personalizado. Esse limiar é determinado estatisticamente pela distribuição dos próprios intervalos do usuário, utilizando uma variação do método de Intervalo Interquartil (IQR) para identificar outliers superiores:

$$\tau = Q_3 + \lambda \cdot \underbrace{(Q_3 - Q_1)}_{IQR}$$

onde Q_1 e Q_3 representam, respectivamente, o primeiro e o terceiro quartil dos intervalos de tempo do usuário, e λ é um hiperparâmetro configurável que controla o rigor do corte. Esta formulação permite adaptar a criação dos grupos temporais tanto em nível de usuário quanto em domínio, podendo acomodar diferentes perfis de consumo. Desta forma, pares de relação seriam formados exclusivamente entre itens pertencentes ao mesmo grupo temporal.

No entanto, aplicar essa regra cegamente pode ser problemático. Em cenários onde ocorrem muitas interações quase instantâneas, como um usuário pulando faixas musicais rapidamente ou logs gerados em lote pelo sistema, a distribuição dos dados fica poluída com valores próximos a zero. Isso distorce os quartis para baixo, criando limiares artificiais muito baixos que, embora façam sentido na formulação matemática, não modelam bem comportamentos no mundo real. Para corrigir essa distorção, é definido formalmente um subconjunto de intervalos válidos, $\mathscr{S}$, aplicando um filtro de ruído T_{min}. Seja $\mathscr{T}$ o conjunto de todos os intervalos de tempo de um usuário, o conjunto válido para cálculo estatístico é dado por:

$$\mathscr{S} = \{\Delta t \in \mathscr{T} \mid \Delta t > T_{min}\}$$

Dessa forma, garante-se que apenas pausas perceptíveis sejam consideradas. Com base na distribuição deste conjunto refinado $\mathscr{S}$, são calculados os quartis Q_1 e Q_3. O limiar de corte (τ) que define o fim de uma sessão é, então, determinado pela cerca superior da distribuição:

$$\tau = Q_3(\mathscr{S}) + \lambda \cdot \underbrace{(Q_3(\mathscr{S}) - Q_1(\mathscr{S}))}_{IQR}$$

Uma vez determinado o limiar τ, o histórico H_u é particionado em um conjunto de grupos temporais $\mathscr{G} = \{G_1, G_2, \ldots, G_K\}$. Um grupo temporal G_m é definido formalmente como uma subsequência contígua de itens onde a continuidade temporal é preservada:

$$G_m = \{(i_j, t_j) \in H_u \mid \Delta t_j \leq \tau\}$$

Consequentemente, a fronteira entre dois grupos distintos ocorre sempre que a condição de continuidade é violada ($\Delta t > \tau$), indicando uma mudança de contexto de consumo.

Embora a restrição dos pares aos grupos G_m seja muito eficaz para capturar a intenção imediata do usuário e garantir uma alta coerência semântica, o isolamento estrito impõe alguns riscos. Restringir o aprendizado apenas ao seu contexto temporal ignora conexões entre interesses manifestados em momentos distintos. Além disso, a redução no volume de pares traz problemas de escassez de dados. Essa diminuição excessiva do número de amostras tende a elevar a variância do modelo, comprometendo sua estabilidade e capacidade de generalização.

Visando superar essa limitação, se desenvolveu uma versão aprimorada das *embeddings* discretas utilizando uma estratégia de *data augmentation*. Nesta abordagem híbrida, o treinamento ocorre em duas frentes dentro da mesma época de treinamento: inicialmente, o modelo de *embeddings* é treinado sem considerar os atributos temporais, gerando pares independentemente de pertencerem ao mesmo grupo ou não. Em seguida, o modelo é realimentado, mas restrito apenas às relações que pertencem ao mesmo grupo temporal. O processo de treinamento é modificado para operar da seguinte maneira:

1. O modelo processa pares gerados a partir do histórico global, garantindo a captura de correlações de longo prazo e a estabilidade vetorial.

2. Simultaneamente, aplica-se um reforço aos pares que coocorrem dentro do mesmo grupo temporal (definido pelo limiar τ).

Uma implementação ingênua de *data augmentation*, que consistisse em duplicar fisicamente os pares intra-grupo no conjunto de dados, resultaria em um aumento significativo no custo computacional por época. Para manter a eficiência, a lógica de reforço foi incorporada diretamente na função de perda do modelo através de um esquema de ponderação.

Define-se um peso escalar ω associado a cada par de treinamento $(i_{centro}, i_{contexto})$. Seja $\mathscr{G}$ o conjunto de todos os grupos temporais identificados, o peso para um par em específico é determinado por:

$$\omega_{i,j} = \begin{cases} 2 & \text{se } i, j \in G_m \text{ para algum } G_m \in \mathscr{G} \\ 1 & \text{caso contrário} \end{cases} \tag{13}$$

Dessa forma, durante a etapa de *backpropagation*, os pares que respeitam a proximidade temporal exercem o dobro de influência na atualização dos gradientes. Esta estratégia permite que a rede neural intensifique o aprendizado de relações temporais fortes, sem negligenciar as conexões de longo prazo presentes no histórico global do usuário.

4.2 Abordagem contínua

Na abordagem discreta, a janela deslizante de tempo funciona como um limite decisivo para escolher quais itens, com base em um item-chave, devem ser considerados como contexto ao fazer previsões com a rede neural. A ideia de uma proposta contínua é que o esquecimento ou a mudança de interesse do usuário não ocorra em degraus abruptos, mas sim como um decaimento gradual. Um item consumido há 15 minutos tem uma relação forte com o atual; um consumido há 24 horas, uma relação média; e um há 30 dias, uma relação fraca. A abordagem contínua busca modelar essa suavidade, eliminando a necessidade de definir fronteiras rígidas de corte.

Na abordagem contínua, de maneira similar aos estudos de Pavlovski et al. (2020) e Vasile, Smirnova e Conneau (2016), o peso é introduzido como um parâmetro W de valor flutuante, onde $W \in [0, 1]$, que atua diretamente com a etapa de aprendizado da rede neural. Durante a fase de *backpropagation*, ao calcular o erro de previsão e ajustar os pesos da rede neural, o parâmetro W é levado em consideração. Itens consumidos com uma maior distância temporal em relação ao item-chave recebem valores menores de W, resultando em uma influência reduzida durante o ajuste da matriz de pesos. Essa abordagem visa aliviar proporcionalmente a contribuição de itens mais distantes temporalmente, mantendo sua relevância no processo. É definido e avaliado um processo de parametrização da função que realizará o decréscimo do valor de W de acordo com o tempo. Na Figura 8 é possível observar a intuição do problema.

Figura 8 – No modelo TAI2Vec-Cont, itens distantes ao item alvo possuem menos influência

Para modelar esse decaimento de forma a ser adaptável a diferentes ritmos de consumo, desenvolveu-se uma função de ponderação híbrida que combina duas métricas baseadas no

histórico do usuário: a distância temporal normalizada (foco global) e o desvio estatístico do comportamento do usuário (foco local).

4.2.1 Pré-processamento e Estatísticas do Usuário

O pré-processamento inicial segue a mesma ordenação cronológica estabelecida no método discreto. No entanto, ao invés de buscar pontos de corte, calculam-se estatísticas descritivas sobre os intervalos de tempo (Δt) para caracterizar o "ritmo" habitual do usuário. Para o cálculo das estatísticas em si, o mesmo filtro de ruído T_{min} mencionado anteriormente é aplicado para controlar iterações muito próximas. Adicionalmente, implementa-se um tratamento de outliers para evitar que intervalos excessivamente longos (e.g., meses de inatividade) inflem artificialmente a média e o desvio padrão. Caso um intervalo exceda o limite superior (*Upper Bound*), ele é substituído por esse valor teto antes do cálculo das estatísticas finais.

Desta forma, seja $\mathscr{S}$ o conjunto de intervalos de tempo válidos após a filtragem de ruídos e *outliers*, se calculam a média (μ_u) e o desvio padrão (σ_u) dos intervalos de cada usuário u:

$$\mu_u = \text{mean}(\mathscr{S}), \quad \sigma_u = \text{std}(\mathscr{S})$$

Além disso, para situar cada interação no tempo global, calcula-se o tempo acumulado $t_{cum}^{(k)}$ para o k-ésimo item da sequência e sua respectiva normalização $t_{norm}^{(k)} \in [0, 1]$:

$$t_{cum}^{(k)} = \sum_{j=2}^{k} \Delta t_j, \quad t_{norm}^{(k)} = \frac{t_{cum}^{(k)} - \min(t_{cum})}{\max(t_{cum}) - \min(t_{cum})}$$

4.2.2 Modelagem da função de decaimento local

A limitação do uso do tempo absoluto (segundos) reside na generalização do comportamento, ao tratar usuários com ritmos distintos de forma homogênea, quando, na verdade, cada um possui seu próprio relógio interno de consumo. A abordagem contínua resolve isso ao utilizar a métrica *Z-score*, ajustando o peso do tempo com base no histórico pessoal de cada um, desta forma, o sistema passa a olhar para o desvio do comportamento individual. Em termos estatísticos, o *Z-score* quantifica exatamente quantos desvios-padrão um intervalo específico se afasta da média histórica do usuário, permitindo medir a anormalidade de cada pausa.

Seja $d_{i,j} = |t_{cum}^{(i)} - t_{cum}^{(j)}|$ a distância temporal absoluta em segundos entre o item central i e um item de contexto j. O afastamento estatístico padronizado $z_{i,j}$ é calculado por:

$$z_{i,j} = \frac{d_{i,j} - \mu_u}{\sigma_u + \epsilon} \tag{14}$$

onde μ_u e σ_u representam a média e o desvio padrão dos intervalos do usuário (previamente calculados e ajustados contra *outliers*), e ϵ é uma constante minúscula para garantir estabilidade numérica em casos de variância nula. Vale notar que, em um cenário ordinário, espera-se que

um *Z-score* permaneça sempre baixo (menor que três). No entanto, como $d_{i,j}$ é a soma de todos os intervalos entre os itens i e j, o valor final acumula-se com a soma das distâncias. Consequentemente, não há um teto para o crescimento de $z_{i,j}$ à medida que a janela de contexto se expande.

Uma vez obtido o valor de $z_{i,j}$, é necessário convertê-lo em um peso de influência $W_{stat} \in (W_{min}, 1]$. Para isso, se emprega uma função de decaimento não-linear baseada em uma transformação racional, controlada por um hiperparâmetro de decaimento α:

$$W_{stat}(i,j) = \max\left(W_{min}, \quad 1 - \left(\frac{z_{i,j}}{z_{i,j}+1}\right)^{\alpha}\right) \tag{15}$$

Nesta configuração, valores de $z_{i,j}$ próximos a zero preservam o peso integral ($W_{stat} \approx 1$), privilegiando interações imediatas. À medida que a distância temporal cresce além do desvio padrão típico, o peso diminui suavemente. Enquanto o termo $\frac{z}{z+1}$ atua como uma função de esmagamento (*squashing function*), mapeando o domínio $[0, \infty)$ para o intervalo $[W_{min}, 1)$, o termo W_{min} atua como um controle para garantir que a influência de itens distantes não seja completamente anulada.

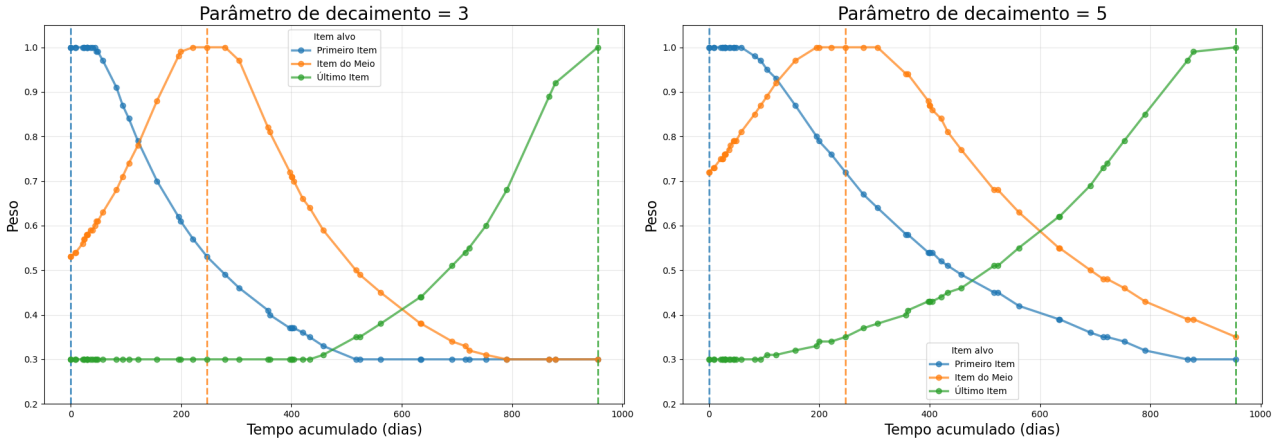


Figura 9 – Comportamento da curva de decaimento do peso a partir de diferentes parâmetros temporais

A Figura 9 ilustra o comportamento da função de decaimento para um usuário da base Amazon Books, com piso $W_{min} = 0.3$. As curvas representam a distribuição de pesos relativa a três itens-alvo, posicionados no início (azul), meio (laranja) e fim (verde) quando comparadas a todos os outros itens consumidos do histórico. Comparativamente, observa-se que o aumento do parâmetro de decaimento (de $\alpha = 3$ para $\alpha = 5$) resulta em uma penalização mais leve para iterações distantes, enquanto o valor menor fornece uma transição mais rígida.

4.2.3 Modelagem da função de decaimento global

A modelagem local baseada em estatísticas apresenta limitações diante de comportamentos considerados como *outliers*. Em históricos muito curtos, a média e o desvio padrão se tornam instáveis devido à escassez de amostras. Por outro lado, em históricos muito longos,

como muitas iterações acumuladas ao longo de anos, o desvio tende a crescer excessivamente, levando a rede neural a interpretar preferências distantes de forma mais irrelevante que o esperado. Desta forma, foi introduzida uma visão global que atua como uma base estável. Diferente da abordagem estatística, que foca na variação dos intervalos, esta etapa considera a posição relativa do item na posição histórica do item na sequência do usuário.

Para a abordagem global, utiliza-se o tempo normalizado t_{norm} variando entre 0 (primeira interação) e 1 (última interação):

$$W_{linear}(i,j) = 1 - (1 - W_{min}) \cdot |t_{norm}^{(i)} - t_{norm}^{(j)}|$$

onde $|t_{norm}^{(i)} - t_{norm}^{(j)}|$ representa a distância temporal normalizada entre o item alvo i e o item de contexto j, variando entre 0 e 1; W_{min} é o hiperparâmetro que define a relevância mínima permitida para interações antigas; e o termo $(1 - W_{min})$ atua como um fator de escalonamento, ajustando a inclinação do decaimento para garantir que a penalidade máxima aplicada pela distância nunca viole o piso de relevância estabelecido.

Na Figura 10, é possível perceber como a modelagem linear oferece uma perspectiva complementar de decaimento. O exemplo dado se refere a um usuário com um histórico inconvencionalmente grande para um *e-commerce* de livros (1.700 registros ao longo de 5 anos). Nele, fica evidenciada uma limitação da abordagem local: o peso mínimo é atingido em interações distantes aproximadamente 100 dias do item alvo. A abordagem global, por outro lado, considera a distância relativa de toda a trajetória temporal, evitando o possível descarte prematuro de preferências antigas do usuário.

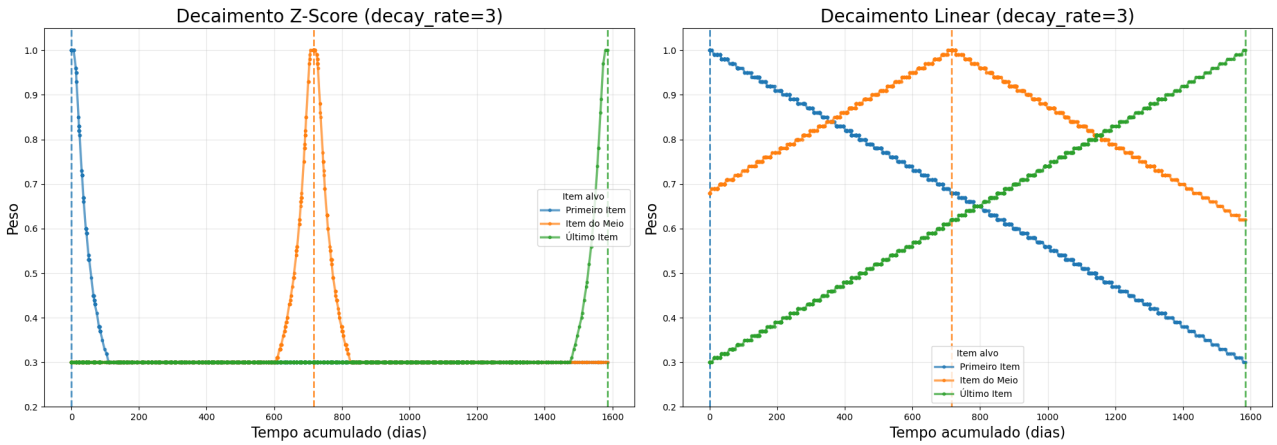


Figura 10 – Comparação entre a curva de decaimento local e global sob o histórico de um usuário inconvencional, ambos com $W_{min} = 0.3$.

É importante reforçar que o decaimento baseado em uma visão global do histórico do usuário é usado apenas para aliviar a curva de decaimento em usuários *outliers* dentro da rede. A aplicação isolada desse método não é suficiente para capturar as nuances do comportamento de consumo, pois sua linearidade global não reflete as variações de ritmo próprias de cada usuário, justificando assim sua composição com a abordagem anterior.

4.2.4 Unificação dos Pesos

O peso final $\omega_{i,j}$, utilizado para escalar o gradiente durante o treinamento, é a média aritmética das duas abordagens:

$$\omega_{i,j} = \frac{W_{stat}(i,j) + W_{linear}(i,j)}{2} \tag{16}$$

Na Figura 11, é possível observar uma modificação da curva resultante da Figura 9 que se aproveita do método híbrido ao invés de utilizar apenas o foco local.

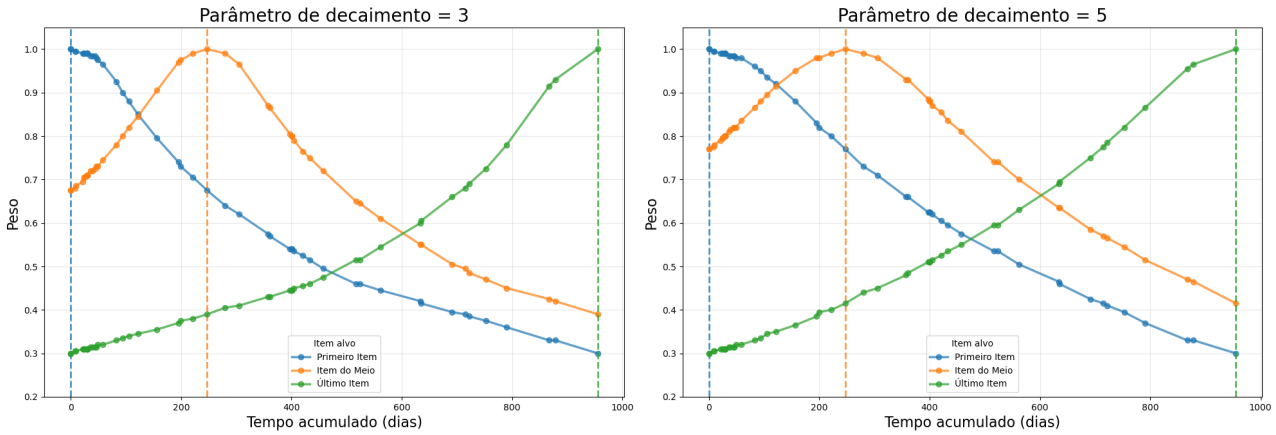


Figura 11 – Comportamento da curva de decaimento do peso a partir de diferentes parâmetros temporais após a média aritmética dos escopos locais e globais.

A eficácia da inserção do contexto global na modelagem de decaimento é evidenciada na Figura 12, que apresenta uma avaliação da assertividade do modelo de recomendação em quatro bases de dados amplamente utilizadas na literatura. A comparação contrasta o desempenho da abordagem com e sem a inclusão do contexto global, utilizando a métrica NDCG. Detalhes aprofundados sobre o protocolo experimental podem ser consultados no Capítulo 5.

A modelagem híbrida, baseada na inclusão do contexto global, apresenta melhorias em assertividade, especialmente em bases menores, justificando o seu uso na proposta final.

Uma vez calculado, o escalar $\omega_{i,j}$ é integrado diretamente à função de perda da rede neural. Diferente da abordagem discreta, que utiliza uma máscara binária para incluir ou excluir itens do contexto, a abordagem contínua utiliza $\omega_{i,j}$ como um multiplicador do gradiente. O erro de predição para pares distantes no tempo é escalado para baixo, reduzindo a magnitude da atualização dos pesos da rede para esses casos específicos.

4.3 Extração das embeddings para recomendação

Arquiteturas do tipo Item2Vec produzem dois conjuntos distintos de representações vetoriais no espaço latente: a matriz de embeddings de itens $\mathbf{W}$ e a matriz de pesos de saída $\mathbf{W}'$. Apesar de ambas capturarem relações estruturais durante o aprendizado, apenas a matriz $\mathbf{W}$ é tipicamente retida para a fase de inferência.

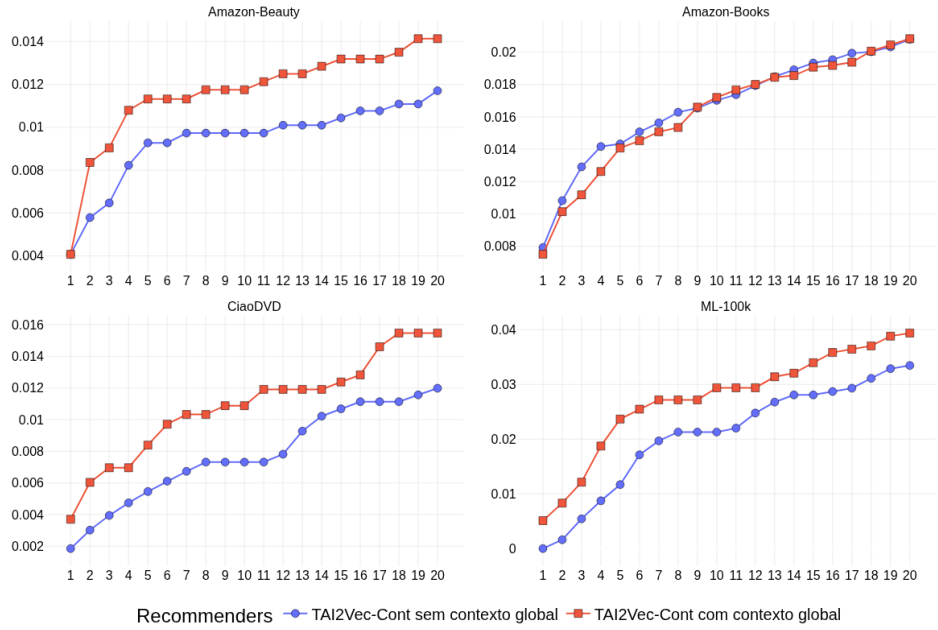


Figura 12 – Comparação entre modelagens de decaimento pela métrica NDCG

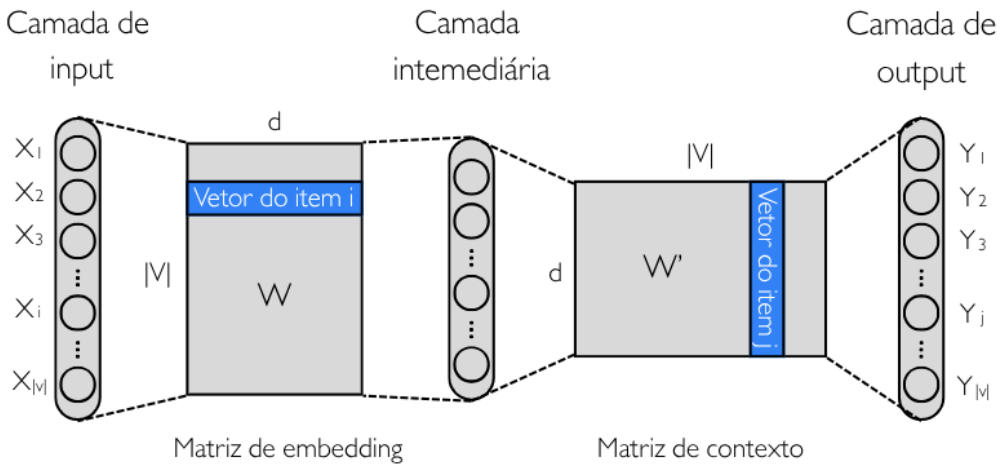


Figura 13 – Comparação entre a curva de decaimento local e global sob o histórico de um usuário inconvencional, ambos com $W_{min} = 0.3$.

Como já demonstrado em estudos anteriores, como Ahn, Rhee e Lee (2022) e Kvernadze et al. (2022), o descarte da matriz de pesos de saída $\mathbf{W}'$ resulta na perda de informações semânticas valiosas que complementam e estabilizam a matriz de entrada $\mathbf{W}$. Para mitigar essa perda, este trabalho adota a metodologia de *Dual Embedding* para a extração destes vetores para as tarefas de recomendação.

Para a construção do vetor de características $\mathbf{x}_{t,a}$, o processo de integração das matrizes de *embeddings* passa por duas etapas:

1. Normalização Individual: Cada vetor extraído das matrizes de pesos de entrada $\mathbf{W}$ e de saída $\mathbf{W}'$ é submetido à normalização L_2 para que ambos contribuam com a mesma escala para a representação final.

2. Combinação: Os vetores normalizados correspondentes ao mesmo item são somados elemento a elemento, combinando as informações de ambos os espaços latentes em um único vetor de mesma dimensionalidade.

Uma vez gerada a matriz de *embeddings* combinada, cabe ao agente recomendador construir a representação do estado do usuário a partir do seu histórico de interações. Dado que as arquiteturas TAI2Vec são centradas na representação de itens, a modelagem das preferências do usuário ocorre de forma dinâmica durante a fase de inferência. As estratégias específicas para essa agregação, assim como a lógica de recomendação, serão discutidas com maior profundidade no capítulo de experimentos.

Capítulo 5

Experimentos e resultados

Como mencionado no Capítulo 1, o principal objetivo deste trabalho é a inclusão de informações temporais de contexto para alimentar *embeddings* em sistemas de recomendação incrementais. Para atingir seu objetivo, este trabalho busca responder às seguintes perguntas de pesquisa:

P1. Em que medida a integração explícita de intervalos temporais entre interações durante o treinamento de modelos de embeddings supera o desempenho de modelos puramente estáticos em métricas de recomendação Top-N?

P2. Como a modelagem dinâmica de contextos temporais personalizados ao comportamento individual dos usuários impacta a qualidade das recomendações em comparação a métodos gerais que utilizam janelas de tempo globais e estáticas?

P3. Protocolos experimentais offline tradicionalmente utilizados na literatura para avaliação de recomendadores incrementais são capazes de mensurar com assertividade a evolução do aprendizado sob um fluxo contínuo de dados?

P4. Como a adaptação de simuladores de larga escala permite a integração de novas representações vetoriais para uma avaliação versátil e livre de vieses de seleção?

Para avaliar de forma adequada a modelagem dinâmica temporal proposta, os experimentos foram estruturados com o objetivo de quantificar a qualidade das *embeddings* em diferentes cenários de recomendação e regimes de feedback. Assim, foram conduzidas as seguintes abordagens avaliativas.

- **Avaliação estática**: Utiliza *logs* históricos e partições globais fixas. A recomendação é feita através de métodos de similaridade usuário-item e item-item, com o objetivo de

avaliar o poder representacional das embeddings TAI2vec em um cenário de recomendação tradicional.

- **Avaliação incremental**: Construída sobre *logs* históricos, mas executada de forma sequencial utilizando agentes *Multi-Armed Bandits*. O objetivo é simular a adaptação contínua do modelo a novos dados e validar a hipótese central deste trabalho quanto à evolução das representações.

- **Avaliação por simulador**: Emprega uma adaptação do simulador KuaiSim para avaliar o desempenho em um ciclo de feedback fechado. A avaliação por simulador possui o objetivo de contornar os vieses inerentes encontrados em avaliações offline ao retornar *feedbacks* imediatos baseados em regras de comportamento previamente estabelecidas.

O objetivo desta tríade experimental é validar como a dimensão temporal impacta a qualidade das representações em três etapas progressivas. Primeiro, se estabelece a eficácia semântica das embeddings em cenários estáticos. Em seguida, é utilizado um protocolo incremental para comprovar a adaptação a fluxos contínuos de dados. Por fim, avalia-se a robustez do modelo em simulações de interações reais. A estrutura proposta testará o potencial *model-agnostic* do TAI2Vec-Disc e do TAI2Vec-Cont, investigando sua resiliência e capacidade de generalização.

5.1 Materiais e métodos

Nesta seção, são detalhadas as configurações comumente compartilhadas entre todos os experimentos conduzidos.

Por serem inspirados em uma mesma arquitetura *Skip-gram*, todos os algoritmos de *benchmark* baseados em *embeddings* neurais e as variantes propostas do TAI2vec compartilham uma mesma implementação base. Detalhes técnicos e arquiteturais dos modelos são discutidos na Seção 5.1.2. Para o ajuste dos modelos, os hiperparâmetros foram definidos através de uma busca em grade exaustiva, seguindo o protocolo detalhado na Seção 5.1.3. As combinações ideais de hiperparâmetros para cada algoritmo podem ser consultadas no Apêndice A.

Os experimentos offline, tanto em suas variações estáticas quanto incrementais, compartilham os mesmos conjuntos de dados apresentados na Seção 5.1.1. Para a avaliação via simulador, é necessário construir as *embeddings* com base no mesmo conjunto de dados utilizado para definir as regras de comportamento do simulador. Demais particularidades metodológicas exclusivas de cada formato de experimento encontram-se abordadas em suas respectivas seções.

5.1.1 Conjuntos de dados

Os conjuntos de dados que serão utilizados nos experimentos offline são apresentados na Tabela 2. A seleção foi feita por se tratar de bases de dados frequentemente usadas em

benchmarks na literatura, abordando diferentes áreas de aplicação e contendo as informações necessárias para execução dos métodos, isto é, interações usuário-item com atributos temporais.

Amazon Beauty, Amazon Books e Amazon Games correspondem ao conjunto de dados Amazon Reviews'23[1], contendo uma vasta quantidade de dados de consumo e avaliação em diferentes setores na empresa de comércio eletrônico Amazon. BestBuy[2] é uma base de dados utilizada em competições organizadas pela Association for Computing Machinery (ACM). CiaoDVD[3] contém interações colectadas do web site britânico *dvd.ciao.co.uk* e utilizadas como *benchmark* para o popular *framework* de recomendação LibRec. Os conjuntos MovieLens-100k e MovieLens-1M (diferenciando-se apenas pela quantidade de itens usuários e iterações) fazem parte do repositório de bases de dados do GroupLens[4], referência e pioneiros na área de recomendação. Finalmente, RetailRocket[5] é a base de dados do sistema de recomendação RetailRocket, utilizado por mais de mil lojas de comércio eletrônico.

Dataset	$\|U\|$	$\|I\|$	$\|R\|$	S	**Período (meses)**	$\overline{\Delta t}$ **(dias)**
Amazon Beauty[1]	631,986	112,565	693,929	99.99%	234.00	422.9
Amazon Books[1]	1,008,954	206,710	2,115,793	99.99%	195.36	485.7
Amazon Games[1]	757,210	80,316	2,579,662	99.99%	286.0	435.5
BestBuy[2]	1,268,702	69,858	1,862,782	99.99%	2.64	3.7
CiaoDVD[3]	16,923	14,800	65,038	99.97%	151.64	81.4
MovieLens-100k[4]	943	1,682	100,000	93.70%	7.08	0.4
MovieLens-1M[4]	6,039	3,628	836,478	96.18%	34.08	1.1
RetailRocket[5]	11,719	21,270	21,270	99.98%	4.56	4.6

Tabela 2 – Estatísticas dos conjuntos de dados utilizados nos experimentos.

Os conjuntos de dados exibem características temporais variadas, contendo bases com interações esparsas e irregulares como a Amazon Beauty, ou padrões de consumo densos, como o MovieLens. Nas colunas da Tabela 2, $|U|$, $|I|$ e $|R|$ representam, respectivamente, o número de usuários, itens e interações. A coluna S descreve a esparsidade do conjunto de dados (a relação entre $|R|$ e $|U| \times |I|$). Adicionalmente, a coluna Período indica o tempo total de abrangência dos dados em meses, enquanto $\overline{\Delta t}$ representa o intervalo médio de tempo entre interações consecutivas de um mesmo usuário, medido em dias.

Durante a etapa de pré-processamento, as interações duplicadas foram removidas, garantindo a retenção de apenas uma ocorrência para cada par usuário-item. Adicionalmente, interações inconsistentes (isto é, casos em que o mesmo par usuário-item apresentava avaliações distintas) foram excluídas. Para os conjuntos de dados com feedback explícito, foram consideradas como interações válidas apenas aquelas com avaliações superiores à avaliação intermediária

[1] Amazon Reviews'23. Disponível em: <https://amazon-reviews-2023.github.io>. Acesso em 17 de junho de 2026.

[2] Data Mining Hackathon on BIG DATA (7GB) Best Buy mobile web site. Disponível em: <kaggle.com/c/acm-sf-chapter-hackathon-big>. Acesso em 17 de junho de 2026.

[3] CiaoDVD. Disponível em: <https://guoguibing.github.io/librec/datasets.html>. Acesso em 17 de junho de 2026.

[4] GroupLens. Disponível em: <grouplens.org/datasets/>. Acesso em 17 de junho de 2026.

[5] Retailrocket recommender system dataset. Disponível em: <kaggle.com/retailrocket/ecommerce-dataset>. Acesso em 17 de junho de 2026.

(R_{mid}), a qual pode ser calculada conforme apresentado na Equação 17, considerando R como o conjunto de todas as avaliações dos usuários em uma dada base de dados.

$$R_{mid} = min(R) + \frac{max(R) - min(R)}{2} \tag{17}$$

Apenas as interações com avaliações $r > R_{mid}$ foram retidas para as etapas de treinamento e avaliação.

5.1.2 Implementação e *Benchmark*

Para a condução dos experimentos e avaliação do desempenho dos métodos TAI2Vec-Disc e TAI2Vec-Cont, foram selecionados quatro algoritmos de geração de *embeddings* consolidados na literatura como *baselines*. São eles: *Alternating Least Squares* (ALS) (TAKÁCS; TIKK, 2012) e *Bayesian Personalized Ranking* (BPR) (RENDLE et al., 2009), baseados em fatoração de matrizes, o Item2Vec (BARKAN; KOENIGSTEIN, 2016), fundamentado em aprendizado de representações para captura de relações entre itens, e o SeqI2V (PIRES; PASCON; ALMEIDA, 2021), uma adaptação do Item2Vec que usa limiares globais para definir o escopo do contexto entre itens.

Os métodos escolhidos oferecem abordagens distintas para a geração de representações vetoriais. Enquanto o ALS e o BPR otimizam funções de custo específicas para minimizar erros de predição ou maximizar ranqueamentos personalizados, o Item2Vec e o SeqI2V utilizam janelas de contexto para extrair relações semânticas a partir de sequências de consumo.

Toda a infraestrutura experimental, incluindo o desenvolvimento dos modelos propostos, foi implementada em linguagem `Python`. As ferramentas e bibliotecas utilizadas no desenvolvimento deste projeto são:

- **Fatoração de Matrizes:** Os algoritmos ALS e BPR foram implementados por meio da biblioteca `implicit`[6], otimizada para conjuntos de dados implícitos.
- **Redes Neurais e *Embeddings*:** Os modelos Item2Vec, SeqI2V, TAI2Vec-Disc e TAI2Vec-Cont foram desenvolvidos integralmente utilizando o *framework* `PyTorch`.
- **Ambiente de Simulação:** Utilizou-se o simulador `KuaiSim` para a execução dos experimentos de recomendação em tempo real.
- **Recomendação Não-Incremental:** A busca por vizinhos próximos via *K-Nearest Neighbors* (HERLOCKER; KONSTAN; RIEDL, 2002) foi implementada com a biblioteca `Faiss`, a qual fornece alta eficiência computacional.

[6] Fast Python Collaborative Filtering for Implicit Datasets. Disponível em <implicit.readthedocs.io/en/latest/>.

- **Recomendação Incremental:** Os algoritmos de *Multi-Armed Bandits* (MABs), especificamente os métodos LinGreedy e LinUCB, foram executados através da biblioteca `Mab2Rec`.

- **Processamento e Visualização:** As bibliotecas `NumPy`, `pandas` e `Scikit-learn` foram empregadas para manipulação de dados e processamento auxiliar, enquanto a `Plotly` foi utilizada para a geração de gráficos.

5.1.3 Parametrização

Para cada formato de avaliação, foi executada uma etapa de ajuste de hiperparâmetros por meio de busca em grade. Os parâmetros testados para os modelos de fatoração de matriz podem ser observados na Tabela 3, enquanto os dos modelos baseados em redes neurais, incluindo as propostas TAI2Vec, na Tabela 4.

Tabela 3 – Espaço de busca de hiperparâmetros para os métodos de fatoração de matrizes.

Hiperparâmetro	**ALS**	**BPR**
Fatores Latentes (k)	$\{50, 100, 200\}$	$\{50, 100, 200\}$
Regularização (λ)	$\{10^{-2}, 10^{-4}, 10^{-6}\}$	$\{10^{-2}, 10^{-4}, 10^{-6}\}$
Número de Épocas	$\{10, 20, 50, 100\}$	$\{10, 20, 50, 100\}$
Taxa de Aprendizado (α)	—	$\{0.0025, 0.025, 0.25\}$

Tabela 4 – Espaço de busca de hiperparâmetros para os modelos baseados em redes neurais.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	$\{5, 10\}$	10	10	10
Taxa de Aprendizado (α)	$\{0.025, 0.25\}$	$\{0.025, 0.25\}$	$\{0.025, 0.25\}$	$\{0.025, 0.25\}$
Número de Épocas	$\{20, 50, 100\}$	$\{20, 50, 100\}$	$\{20, 50, 100\}$	$\{20, 50, 100\}$
Limiar de Sub Sampling (t)	$\{10^{-3}, 10^{-4}, 10^{-5}\}$	$\{10^{-3}, 10^{-4}, 10^{-5}\}$	$\{10^{-3}, 10^{-4}, 10^{-5}\}$	$\{10^{-3}, 10^{-4}, 10^{-5}\}$
Exp. de Neg. Sampling (η)	$\{-1, -0.5, 0.5, 1\}$	$\{-1, -0.5, 0.5, 1\}$	$\{-1, -0.5, 0.5, 1\}$	$\{-1, -0.5, 0.5, 1\}$
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	$\{182, 365, 730\}$	—	—
Filtro de Tempo Mínimo	—	—	300s	300s
Expoente de Tempo	—	—	$\{1, 1.5, 2\}$	—
Taxa de Decaimento	—	—	—	$\{3, 5\}$
Peso mínimo	—	—	—	$\{0.3, 0.5\}$

Os parâmetros escolhidos para os métodos de redes neurais foram selecionados com base nos apontamentos de (CASELLES-DUPRéS; LESAINT; ROYO-LETELIER, 2018) e em uma extensa avaliação empírica preliminar. Além das variáveis exploradas na busca em grade, certas configurações foram mantidas fixas por não apresentarem variações significativas no desempenho final durante os testes iniciais. Assim, definiu-se para todos os modelos a utilização de 7 amostras negativas por amostra positiva, embeddings com dimensão $d = 50$ e tamanho de batch de 2^{14}. Adicionalmente, aplicou-se um fator de regularização de 10^{-6} para garantir a estabilidade do processo de convergência. Os parâmetros temporais do SeqI2V, em específico,

foram escolhidos a partir dos valores definidos no seu trabalho original (PIRES; PASCON; ALMEIDA, 2021).

Métodos de *embeddings* de contexto, como o Item2Vec, tradicionalmente empregam uma janela contextual de tamanho fixo. Entretanto, ao incluir atributos temporais para realizar a ponderação do contexto, tais como as propostas TAI2Vec-Disc e TAI2Vec-Cont, tal procedimento se torna desnecessário. Entretanto, devido a limitações computacionais geradas por históricos de consumo excessivamente longos, foi aplicada uma janela contextual deslizante também nos métodos temporais. Ainda assim, foi utilizada uma janela de tamanho "alto", de forma a não atrapalhar a captura de padrões temporais por ambos os métodos. O Item2Vec, por outro lado, requer a hiperparametrização deste valor para averiguar se um contexto estático sozinho é capaz de impulsionar o desempenho, permitindo uma comparação justa entre os métodos.

5.1.4 Métricas de avaliação e testes estatísticos

Para garantir uma análise rigorosa do desempenho dos modelos desenvolvidos, foram implementadas métricas de assertividade, ordenamento e diversidade, compartilhadas entre os diferentes cenários de avaliação.

Em relação à assertividade, foi utilizada a taxa de acerto, ou ***Hit Rate* (HR)**. O *Hit Rate* é um quantificador binário para cada usuário, retornando o valor 1 caso ao menos um item recomendado esteja presente na lista de itens consumidos, e 0 caso contrário. Dessa forma, a métrica mensura a proporção de usuários que receberam ao menos uma recomendação correta, sendo definida pela razão entre o total de acertos (*hits*) e o número total de usuários:

$$\mathrm{HR} = \frac{\#hits}{\#users} \tag{18}$$

Complementarmente, também foi empregado o Ganho Cumulativo Descontado Normalizado, ou *Normalized Discounted Cumulative Gain (NDCG)*, no qual, além de acerto, também é avaliado o ordenamento dos itens recomendados. Para seu cálculo, é necessário três etapas distintas: inicialmente, é calculado o Ganho Cumulativo Descontado (DCG) e, em seguida, o Ganho Cumulativo Descontado Ideal (IDCG); o DCG quantifica a utilidade da lista de recomendações gerada pelo modelo, atribuindo pesos maiores aos itens que aparecem no topo do ranking, enquanto o IDCG representa a pontuação máxima teórica, simulando um cenário onde o sistema ordena perfeitamente todos os itens de interesse do usuário nas primeiras posições possíveis. Ambas as métricas operam no pressuposto de que a utilidade de um item deve sofrer um decaimento conforme sua posição no ranking aumenta, sendo expressas pelas equações:

$$\mathrm{DCG} = \frac{1}{|U|}\sum_{u\in U}\sum_{n=1}^{N}\frac{g(n,u)}{\log_2(n+1)} \qquad \text{e} \qquad \mathrm{IDCG} = \frac{1}{|U|}\sum_{u\in U}\sum_{n=1}^{Z}\frac{1}{\log_2(n+1)} \tag{19}$$

onde a função $g(n, u)$ atua como um verificador binário para determinar a relevância do item na posição n da recomendação $s(u, N)$ para o usuário u. Ela assume o valor 1 se o item recomendado pertence ao conjunto de itens efetivamente consumidos (I_u), e 0 caso contrário:

$$g(n, u) = \begin{cases} 1, & \text{se } s(u, N)_n \in I_u \\ 0, & \text{caso contrário} \end{cases} \tag{20}$$

No cálculo do ganho ideal, a variável Z limita a soma ao valor mínimo entre o tamanho da lista recomendada N e a quantidade total de itens consumidos pelo usuário ($|I_u|$), ou seja, $Z = \min(N, |I_u|)$. Após a obtenção desses valores, o NDCG é derivado através da normalização do ganho real pelo ganho máximo possível:

$$\text{NDCG@} \ = \frac{\text{DCG}}{\text{IDCG@}} \tag{21}$$

Enquanto o Hit Rate se preocupa exclusivamente em validar se a recomendação está correta ou não, o NDCG foca na forma como essas recomendações são apresentadas, recompensando algoritmos que são eficazes em posicionar os itens relevantes nas primeiras colocações da lista sugerida. Ambas variam no intervalo $[0, 1]$, onde valores maiores indicam um desempenho superior.

Para os experimentos *online*, foram também observadas métricas de diversidade. A primeira é a **Diversidade Intra-Lista (ILD)**, que mede a variedade de itens dentro de uma mesma lista de recomendação, buscando evitar redundância entre as opções apresentadas ao usuário. Considerando um conjunto de k itens distintos $\{i_1, i_2, \ldots, i_k\}$ com vetores de representação normalizados v_j, a ILD é definida como:

$$\text{ILD} = 1 - \frac{2}{k(k-1)} \sum_{j=1}^{k} \sum_{l=j+1}^{k} \cos(v_j, v_l), \tag{22}$$

em que $\cos(v_j, v_l)$ representa a similaridade de cosseno entre os vetores dos itens i_j e i_l.

Por último, calculou-se também a **Cobertura**, que mede a proporção do catálogo de itens que foi efetivamente exposta aos usuários:

$$\text{Cobertura} = \frac{|I_{\text{expostos}}|}{|I_{\text{total}}|}, \tag{23}$$

em que $|I_{\text{expostos}}|$ corresponde ao número de itens distintos recomendados pelo menos uma vez e $|I_{\text{total}}|$ ao número total de itens do catálogo.

Ambas as métricas desempenham um papel importante na avaliação do sistema de recomendação para além da precisão imediata. Enquanto a ILD assegura que o usuário não seja desmotivado por recomendações muito similares, a métrica de Cobertura monitora a capacidade do algoritmo de explorar todo o potencial do catálogo disponível.

Para garantir a validade estatística das comparações entre os modelos, foi também aplicado o **Teste de Friedman**, um método não-paramétrico utilizado para detectar diferenças estatisticamente significativas entre os resultados obtidos ao longo dos N conjuntos de dados testados. A estatística de Friedman baseia-se no ranqueamento dos modelos em cada base de dados, sendo definida por:

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4} \right], \tag{24}$$

onde k é o número de modelos, N o número de conjuntos de dados e R_j a soma dos postos (*ranks*) do j-ésimo modelo. Caso o valor-p resultante seja inferior ao nível de significância α, rejeita-se a hipótese nula de que todos os modelos possuem desempenho equivalente.

Complementarmente, para realizar comparações par a par entre o modelo proposto e os *benchmark*, utilizou-se o **Teste de Bonferroni-Dunn**, que quantifica se a diferença no *ranking* médio dos algoritmos avaliados se distancia acima de um dado intervalo de confiança (CD):

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{25}$$

em que q_α corresponde a um valor crítico de avaliação para garantir significância estatística.

5.2 Avaliação estática

Ainda que o objetivo central deste trabalho resida no cenário incremental, desenvolver e avaliar *embeddings* diretamente através de algoritmos de recomendação incremental pode ser computacionalmente custoso. Além disso, a falta de um protocolo experimental bem definido dificulta a comparação dos resultados com outros algoritmos presentes na literatura.

Desta maneira, foram primeiramente implementados algoritmos de recomendação estáticos, que geram as recomendações em lote a partir de partições de treino e teste geradas sobre bases de dados de recomendação. Como vantagem, o ambiente estático permite o isolamento de variáveis, eliminando o ruído das estratégias de exploração para focar exclusivamente na capacidade das *embeddings* de capturar relações entre usuários e itens.

O objetivo do experimento é avaliar a efetividade das *embeddings* geradas pelas propostas temporais aplicadas em um cenário de recomendação estática. Para gerar as recomendações a serem avaliadas, as *embeddings* foram empregadas em algoritmos baseados em similaridade. Para os modelos de fatoração de matriz ALS e BPR, capazes de gerar representações de itens e usuários, foi utilizada similaridade usuário-item. Já para os modelos de *embeddings* neurais (Item2Vec, SeqI2V e TAI2Vec), foi adotada recomendação item-item.

Na recomendação usuário-item, calcula-se a similaridade entre cada usuário e todos os itens não interagidos para gerar recomendações. Isso é realizado utilizando métricas de similaridade padrão, como similaridade cosseno ou produto escalar (ALJUNID; HUCHAIAH, 2021). Para um dado usuário u, sua representação p_u é comparada com as representações de todos os itens não consumidos q_i. A pontuação de similaridade $R_{u,i}$ entre o usuário u e o item i é calculada conforme mostrado na Equação 26:

$$R_{u,i} = p_u \cdot q_i \tag{26}$$

No caso da recomendação de similaridades Item-Item, para cada usuário u, o conjunto de itens que ele consumiu anteriormente, I_u, é recuperado (ALJUNID; HUCHAIAH, 2021). A similaridade $S_{u,i}$ entre um item não consumido i e os itens consumidos $j \in I_u$ é então calculada como a média das similaridades entre i e os itens em I_u, conforme dado na Equação 27:

$$S_{u,i} = \frac{\sum_j^{I_u} q_i \cdot q_j^T}{|I_u|} \tag{27}$$

5.2.1 Protocolo experimental

O treinamento e avaliação dos modelos foi realizado por meio de uma estratégia *holdout*, na qual os conjuntos de dados, já pré-processados, são ordenados temporalmente e divididos em três subconjuntos (treinamento, validação e teste), em uma proporção respectiva de 8:1:1, garantindo que todos os dados de treinamento sejam de um período anterior ao dos dados de teste e validação, e todos os dados de validação de um período anterior ao de teste, respeitando

a sequência temporal dos eventos. Ainda que comumente seja empregada a estratégia *leave-one-out*, na qual, para o histórico de cada usuário, é separado o último item pra teste e o penúltimo para validação, Gusak et al. (2025) concluíram que a separação baseada em um ponto temporal fixo é na verdade a única estratégia onde não há nenhum tipo de vazamento de dados na avaliação.

Após a separação em treino e teste, foram removidos das bases de treinamento todos os usuários que possuíam apenas uma interação. O procedimento foi realizado para adequar a base aos métodos baseados no Item2Vec, que dependem de que usuários consumam ao menos dois itens diferentes para poder gerar coocorrências e alimentar suas redes neurais. Em seguida, evitou-se o problema de *cold-start* nas bases de validação e teste, removendo todos os itens e usuários que não se encontravam na base de treinamento. Desta forma, ao prever a nota ou gerar a recomendação, os métodos teriam à disposição todos os usuários e itens requisitados.

Para o ajuste de parâmetros, foram testadas diferentes combinações de valores de forma exaustiva na partição de validação, por meio de busca em grade, usando os parâmetros mencionados na Seção 5.1.3. No experimento final foram então executados todos os protocolos de manipulação dos dados já mencionados, com a única diferença de que a base de treinamento foi combinada com a de validação, evitando criar uma lacuna temporal entre as iterações treinadas e avaliadas, antes de ser aplicado sobre a base de teste.

Os métodos foram avaliados por meio das métricas HR@N e do NDCG@N, explicadas em maiores detalhes na Seção 5.1.4

5.2.2 Resultados

Para avaliar o desempenho das variantes propostas do TAI2Vec em relação aos algoritmos de *benchmark* ALS, BPR, Item2Vec e SeqI2V, foram conduzidos experimentos focados na recomendação Top-N. A Figura 14 apresenta o desempenho da métrica NDCG variando N de 1 a 20. Adicionalmente, as Tabelas 5 e **??** sumarizam os resultados para NDCG@10 e Hit Rate@10, respectivamente. Para facilitar a leitura e análise dos resultados, as células das tabelas foram formatadas em escala de cinza, onde tons mais escuros representam os melhores desempenhos em cada conjunto de dados.

Os resultados demonstram que as variantes do TAI2Vec apresentam desempenho competitivo, superando frequentemente os *benchmarks* estáticos e sequenciais. A variação discreta, especificamente, obteve os melhores resultados globais, superando o Item2Vec e o SeqI2V em 75% das bases avaliadas.

As principais exceções ocorreram nas bases ML-100k e ML-1M, onde os métodos TAI2Vec apresentaram desempenho inferior ao SeqI2V e, no caso da ML-1M, também ao Item2Vec. Este comportamento pode ser explicado por este conjunto de dados se diferenciar dos demais ao apresentar uma alta densidade de interações em um intervalo de tempo curto (característica intensificada na ML-1M), criando um cenário em que as preferências dos usuários tendem a ser

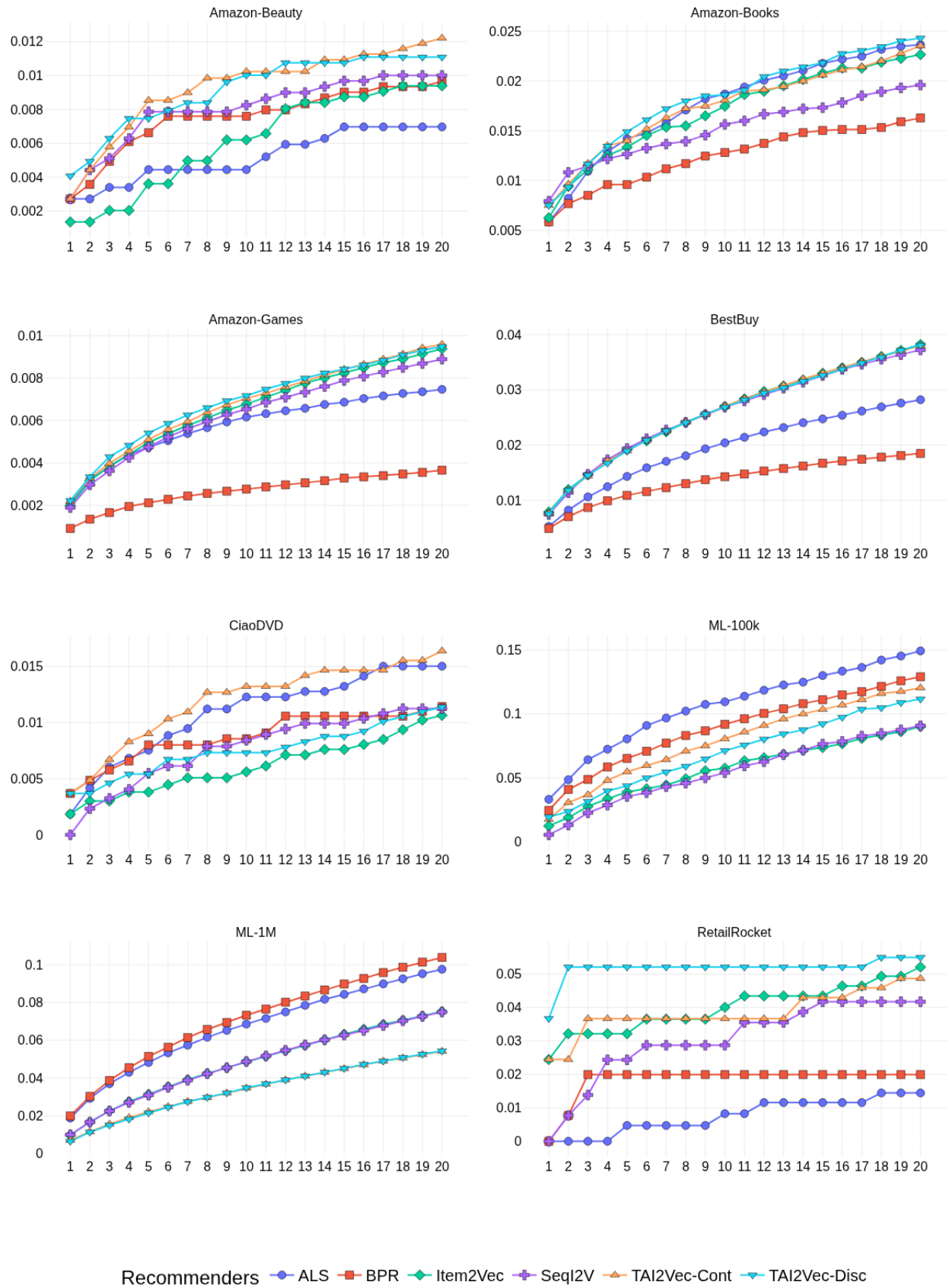


Figura 14 – NDCG por recomendação Top-N

Tabela 5 – Comparação de recomendadores para NDCG@10

Conjunto de dados	ALS	BPR	Item2Vec	Seq2Vec	TAI2Vec-Cont	TAI2Vec-Disc
Amazon-Beauty	0.0044	0.0076	0.0062	0.0083	0.0102	0.01
Amazon-Books	0.0187	0.0128	0.0175	0.0156	0.0181	0.0186
Amazon-Games	0.0062	0.0028	0.0068	0.0065	0.007	0.0072
BestBuy	0.0204	0.0143	0.027	0.0269	0.0271	0.0269
CiaoDVD	0.0123	0.0086	0.0056	0.0084	0.0132	0.0073
ML-100k	0.1095	0.0918	0.0575	0.0539	0.0805	0.0712
ML-1M	0.0685	0.0732	0.0487	0.0487	0.0347	0.0348
RetailRocket	0.0083	0.0199	0.0401	0.0287	0.0367	0.0521

Tabela 6 – Comparação de recomendadores para Hit Rate@10

Conjunto de dados	ALS	BPR	Item2Vec	Seq2Vec	TAI2Vec-Cont	TAI2Vec-Disc
Amazon-Beauty	0.0071	0.0142	0.0156	0.0156	0.0212	0.0198
Amazon-Books	0.0422	0.0274	0.0384	0.0323	0.04	0.0395
Amazon-Games	0.0138	0.0063	0.0159	0.0154	0.0164	0.0164
BestBuy	0.068	0.0437	0.0878	0.0871	0.0871	0.0874
CiaoDVD	0.0649	0.0455	0.039	0.0649	0.0909	0.0455
ML-100k	0.4941	0.3647	0.3765	0.4	0.4235	0.4588
ML-1M	0.6439	0.6567	0.5397	0.5414	0.4697	0.4791
RetailRocket	0.0645	0.0968	0.1613	0.1613	0.129	0.1613

mais estáveis e capturadas de forma eficaz por padrões de coocorrência de longo prazo. Adicionalmente, os *timestamps* das interações derivam do momento da avaliação do filme, e não de quando o usuário efetivamente o assistiu; os modelos TAI2Vec são afetados por essa imprecisão por dependerem do tempo exato para o cálculo de métricas, enquanto o SeqI2V utiliza cortes fixos no tempo e demonstra maior resistência a esse ruído.

Em contrapartida, os ganhos mais expressivos do TAI2Vec são observados em cenários do mundo real caracterizados por intervalos de interação esparsos e irregulares, onde os sinais temporais fornecem o contexto para distinguir mudanças na intenção do usuário. Isso sugere que o TAI2Vec é mais adequado para cenários reais com heterogeneidade temporal, em oposição a conjuntos de dados de avaliações curados e densos, sendo particularmente robusto em ambientes onde os métodos de *embeddings* convencionais têm dificuldade em modelar a evolução do comportamento do usuário devido à escassez de dados.

Em relação aos métodos de fatoração de matriz, os resultados foram variados de acordo com a base de dados analisadas. Como visto na Tabela 5, os modelos, especialmente sua variação Discreta, apresentaram resultados competitivos com o ALS e o BPR, superando ao menos um dos algoritmos de fatorização em 75% dos casos, e falhando apenas nos cenários já observados das bases MovieLens. Ainda que o ALS e o BPR tenham apresentados resultados superiores certas bases, os valores alcançados são mais instáveis quando comparados ao TAI2Vec, decaindo expressivamente em cenários específicos (como o ALS na Amazon Beauty ou o BPR na BestBuy). Entretanto, por não se tratarem de modelos de *embeddings* neurais de

itens, como o TAI2Vec, as análises comparativas foram conduzidas especialmente em relação ao Item2Vec e ao SeqI2V.

Tabela 7 – Ganho percentual sob Item2Vec para as métricas NDCG@10 e Hit Rate@10 em recomendação não incremental

	NDCG@10			**Hit Rate@10**		
Conjunto de dados	Seq2Vec	TAI2Vec-Cont	TAI2Vec-Disc	Seq2Vec	TAI2Vec-Cont	TAI2Vec-Disc
Amazon-Beauty	+33.07%	+65.16%	+61.58%	+0.00%	+36.36%	+27.27%
Amazon-Books	-10.40%	+3.47%	+6.62%	-15.71%	+4.29%	+2.86%
Amazon-Games	-3.49%	+3.99%	+5.93%	-3.01%	+2.88%	+3.27%
BestBuy	-0.42%	+0.30%	-0.32%	-0.82%	-0.87%	-0.43%
CiaoDVD	+49.92%	+135.52%	+30.21%	+66.67%	+133.33%	+16.67%
ML-100k	-6.20%	+40.17%	+23.86%	+6.25%	+12.50%	+21.87%
ML-1M	+0.01%	-28.81%	-28.46%	+0.32%	-12.97%	-11.23%
RetailRocket	-28.32%	-8.43%	+30.09%	+0.00%	-20.00%	+0.00%
Mean	4.27%	26.42%	16.19%	6.71%	19.44%	7.54%
Median	-1.96%	3.73%	15.24%	0.00%	3.58%	3.06%

A Tabela 7 apresenta o ganho percentual dos algoritmos temporais em comparação ao Item2Vec. Observa-se que a melhoria na taxa de acerto é, em geral, inferior à obtida no NDCG. Esse comportamento evidencia a superioridade das variantes do TAI2Vec especificamente na tarefa de ordenação: embora a taxa de acertos seja competitiva, os algoritmos temporais demonstram uma capacidade maior de posicionar os itens de interesse no topo do ranking.

Para validar os resultados apresentados, foi realizado um teste de Friedman nos valores de NDCG@10 ($p = 0,382$). Embora o p-valor não tenha atingido o limiar rigoroso de $\alpha = 0,05$, o resultado é de certa forma esperado dado o tamanho moderado da amostra ($N = 8$ conjuntos de dados) e a alta variância entre domínios.

Com base nos dados apresentados, é possível responder parcialmente duas das perguntas de pesquisa que motivaram este estudo:

- **P1: Em que medida a integração explícita de intervalos temporais entre interações durante o treinamento de modelos de embeddings supera o desempenho de modelos puramente estáticos em métricas de recomendação Top-N?**

  A integração temporal superou consistentemente os modelos estáticos. O TAI2Vec-Cont apresentou ganhos expressivos sobre o Item2Vec, com crescimentos médios de 26,42% em NDCG@10 e 16,19% em Hit Rate@10. Mesmo sob uma análise conservadora pela mediana, tratando de *outliers* como CiaoDVD, o modelo manteve ganhos positivos de 3,73% (NDCG) e 15,24% (Hit Rate). Esses índices indicam os benefícios da integração temporal em uma avaliação estática.

- **P2: Como a modelagem dinâmica de contextos temporais personalizados ao comportamento individual dos usuários impacta a qualidade das recomendações em comparação a métodos gerais que utilizam janelas de tempo globais e estáticas?**

A modelagem dinâmica do TAI2Vec oferece resultados consistentemente superiores ao SeqI2V na maioria das bases. Enquanto o SeqI2V apresenta ganhos irrelevantes ou até mesmo pioras no resultado (-1,96%), o TAI2Vec-Disc mantém um crescimento mediano sólido (+15,24%). O TAI2Vec mostra-se superior especialmente onde o SeqI2V falha, como na RetailRocket, onde o SeqI2V perde para o benchmark estático (-28,32%) enquanto o TAI2Vec-Disc atinge +30,09% de ganho. Isso confirma que adaptar-se ao ritmo individual do usuário, ao menos em um cenário offline estático, é mais eficaz do que aplicar janelas globais rígidas.

Além da qualidade da recomendação em cenários estáticos, a viabilidade desses modelos na validação da hipótese depende da sua capacidade de adaptação contínua a novos dados. Para investigar essa característica, a próxima seção explora o desempenho das variantes proposta em cenários de aprendizado incremental, avaliando como o modelo evolui conforme novas interações são processadas em tempo real.

5.3 Avaliação incremental

Multi-Armed Bandits (MAB) fornecem uma estrutura para a tomada de decisão sequencial em que um agente repetidamente seleciona ações dentro de um conjunto de opções e gera aprendizados novos por meio de tentativa-e-erro (AFSAR; CRUMP; FAR, 2021). No contexto de sistemas de recomendação, isso corresponde à recomendação de um item e à observação das respectivas recompensas ou do feedback do usuário. Algoritmos de MAB clássicos observam apenas o feedback do braço escolhido, o que limita sua capacidade de generalização. Os *Contextual Bandits* (CMAB) estendem essa abordagem ao incorporar informações auxiliares, ou seja, vetores de contexto que descrevem o estado atual do ambiente, tornando-os capazes de generalizar o aprendizado (SILVA et al., 2022). Nesse contexto, a forma como os itens e usuários são representados afeta diretamente o desempenho do modelo. Uma representação latente que capture com precisão as características semânticas dos itens permite que o agente identifique padrões de recompensa de maneira mais eficiente, acelerando a convergência.

Para avaliar a qualidade das representações propostas em um cenário contínuo, esta seção apresenta um protocolo de avaliação incremental sobre bases de dados offline. Nesse formato, os dados são apresentados ao agente cronologicamente, simulando um fluxo contínuo de interações em tempo real. Cada interação segue um ciclo de três etapas: o modelo recebe o contexto, realiza uma predição e, após a verificação do *ground truth*, utiliza o feedback para atualizar seus pesos de forma imediata.

Por se tratar de uma avaliação offline, o protocolo está sujeito a vieses inerentes por se restringir a *logs* históricos provenientes de uma política anterior desconhecida, problema melhor detalhado na Seção 2.2.3. No entanto, este fator não compromete a validação da hipótese central, uma vez que o objetivo do experimento é a comparação entre diferentes métodos de representação de itens e usuários, e não a descoberta de novas informações através de estratégias de exploração. Dado que todas as estratégias exploratórias e os algoritmos de aprendizado serão mantidos constantes, qualquer variação no desempenho é resultado exclusivamente da qualidade das *embeddings* testadas.

5.3.1 Modelagem do estado do usuário e recomendação

Como já visto anteriormente na Seção 3.2.4, as abordagens de aprendizado incremental que incorporam *embeddings* podem ser generalizadas em um *framework* unificado composto por três módulos: de *embeddings*, de representação de estados e de aprendizado, conforme observado na Figura 15.

Os métodos TAI2Vec-Disc e TAI2Vec-Cont, assim como os de benchmark, atuam diretamente na implementação do módulo de *embeddings* ao fornecerem as representações latentes dos itens. Essas representações servem de base para que o sistema de recomendação opere como um esquema de CMABs, no qual o módulo de representação de estados consolida o histórico de interações do usuário para gerar o contexto necessário à tomada de decisão. Dessa forma, a

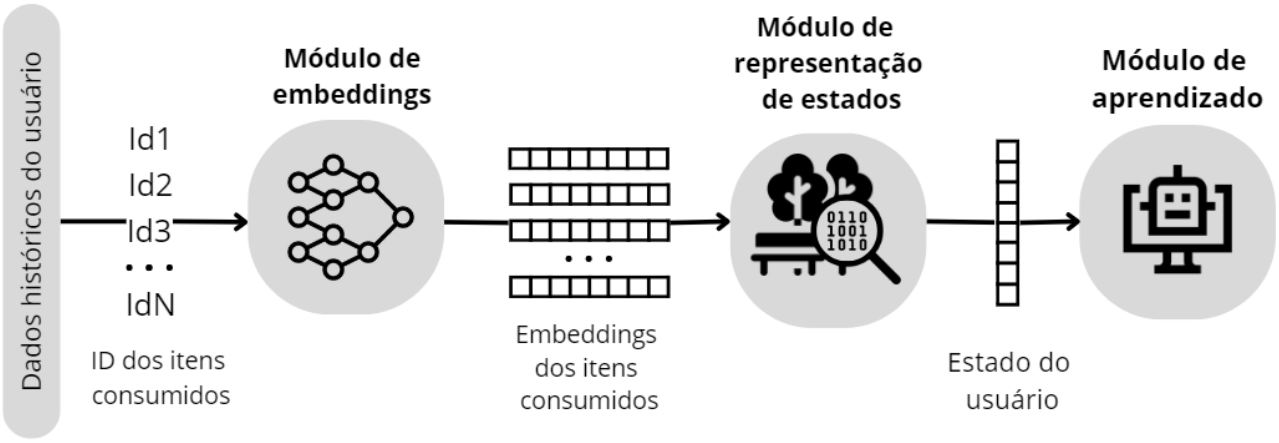


Figura 15 – Representação em alto nível do *framework* unificado proposto por Liu et al. (2024)

representação latente do usuário $\mathbf{u}$ assume o papel do contexto $\mathbf{x}_t$ disponível para o agente no momento da iteração.

Neste esquema, o contexto é derivado dinamicamente durante a inferência através da agregação do histórico de interações H_u. Seguindo os protocolos padrão, o vetor do usuário é definido como o centroide (média aritmética) das *embeddings* dos itens que ele consumiu:

$$\mathbf{u} = \frac{1}{|H_u|} \sum_{j \in H_u} \mathbf{v}_j \tag{28}$$

Para um item candidato i, que representa um braço $a \in \mathscr{A}$, a pontuação de preferência $S_{u,i}$ é calculada por meio da similaridade de cosseno entre o vetor de contexto do usuário e a *embedding* do item candidato. Matematicamente, essa operação equivale à média das similaridades entre o item candidato e todos os itens presentes no histórico do usuário:

$$S_{u,i} = \mathbf{u}^\top \mathbf{v} i = \frac{1}{|H_u|} \sum j \in H_u \mathbf{v}_j^\top \mathbf{v}_i \tag{29}$$

A Figura 16 ilustra a intuição por trás desta representação implícita, onde o usuário é projetado no espaço latente pela agregação dos itens consumidos anteriormente. Nela, os itens candidatos, destacados com uma borda, são pontuados por sua proximidade média em relação a esse histórico.

Na abordagem proposta baseada em *bandits*, a política de seleção de braços consiste em ordenar os itens não consumidos em ordem decrescente de $S_{u,i}$, tratando a pontuação como a recompensa esperada para aquele contexto e selecionando os K melhores candidatos para recomendação. Esta abordagem garante que a qualidade da recomendação seja impulsionada pelas semânticas temporalmente fundamentadas aprendidas durante a etapa de treinamento.

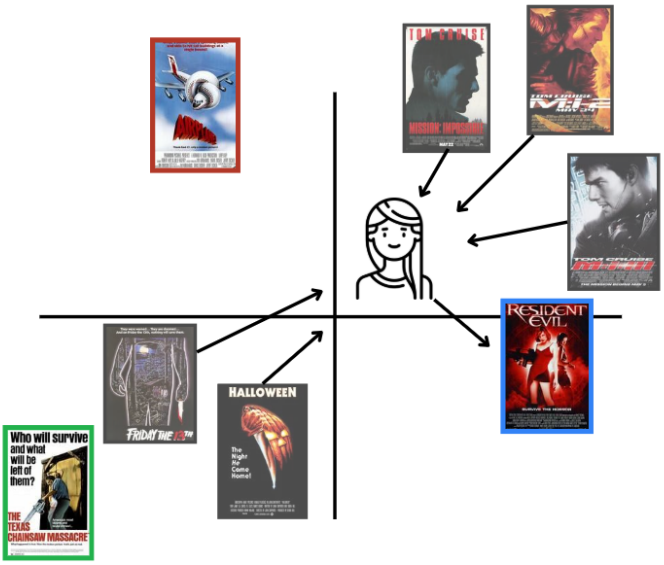


Figura 16 – Ilustração da modelagem do usuário via *embeddings* de item.

5.3.2 Protocolo experimental

Para avaliar os modelos, foi adotado o protocolo experimental ilustrado na Figura 17. Primeiramente, os conjuntos de dados foram ordenados cronologicamente para simular um fluxo contínuo de interações. Cada base foi dividida em duas metades principais: a primeira destinada ao ajuste inicial e a segunda à avaliação propriamente dita **(Passo 1)**. Da porção inicial, os primeiros 40% dos dados foram reservados exclusivamente para o treinamento dos modelos de *embeddings* TAI2Vec-Disc, TAI2Vec-Cont, e *benchmarks* **(Passo 2)**, enquanto os 10% subsequentes serviram para a busca em grade e seleção de hiperparâmetros, cujos valores detalhados podem ser consultados na Seção 5.1.3. Os melhores parâmetros encontrados para o modelo de *embeddings* permaneceram os mesmos durante o restante da avaliação **(Passo 5)**. Os estados dos usuários foram modelados dinamicamente como o centroide dos vetores dos itens consumidos em seu histórico, conforme a lógica descrita na Seção 5.3.1 **(Passo 3)**. Dessa forma, cada iteração é representada por um par de vetores que compõem o contexto necessário para o funcionamento dos *Contextual Bandits*.

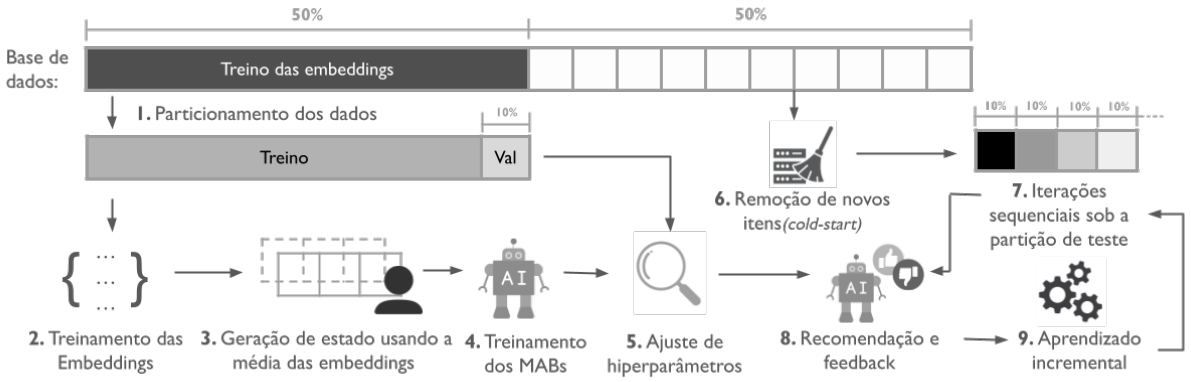


Figura 17 – Diagrama do protocolo experimental para recomendação incremental.

Para este tipo de avaliação incremental, é importante que os algoritmos não comecem com seus pesos aleatórios, pois os dados disponíveis para teste são limitados e a ausência de treinamento prévio poderia impedir que o modelo atingisse a convergência a tempo de gerar resultados relevantes. Por isso, é realizada uma etapa de aquecimento **(Passo 4)**, onde o modelo é exposto sequencialmente às interações da partição inicial para realizar o ajuste prévio de seus parâmetros de decisão. O aquecimento garante que, ao chegar à etapa de avaliação, o agente apresente uma estimativa mínima de quais braços atingem melhores resultados para determinados perfis de usuários. Como consequência, essa inicialização atua na redução do impacto do problema de *cold-start* e assegura que as métricas de desempenho não sejam um reflexo da sua aleatoriedade inicial.

A segunda metade do conjunto de dados foi integralmente dedicada à avaliação incremental, operando sob o regime *test-then-train*. Este processo é inerentemente cíclico, repetindo-se a cada novo lote de dados até que toda a base de teste seja processada:

- **Processamento do Lote e Filtragem (Passos 6 e 7):** A partição de teste é segmentada em fatias temporais de 10% cada. No início de cada iteração, os novos dados são integrados ao fluxo, realizando o pré-processamento para a remoção de itens em cenário de *cold-start*. Isso garante que o agente atue apenas sobre itens previamente vistos e devidamente mapeados no espaço de *embeddings*.

- **Avaliação (Passo 8):** Seguindo a premissa *test-then-train*, o modelo utiliza o conhecimento acumulado até a iteração anterior para gerar recomendações para o lote atual. O sistema observa a recompensa atual contida nos dados, servindo como a métrica oficial de desempenho do agente para aquela fatia temporal.

- **Aprendizado Incremental (Passo 9):** Após a fase de avaliação, as interações recém-observadas no Passo 8 são utilizadas para atualizar incrementalmente os modelos lineares dos agentes.

- **Repetição do Ciclo:** O processo retorna ao **Passo 6**, onde os dados que acabaram de ser utilizados para teste e treino passam a compor o histórico fixo para a próxima fatia de 10%. Esse *loop* contínuo garante que o modelo evolua dinamicamente, incorporando o aprendizado de cada lote antes de processar o passo temporal subsequente.

Para o aprendizado e geração das recomendações, foram utilizados os algoritmos LinGreedy e LinUCB. Ambos foram integrados ao ciclo incremental de teste e treino **(Passos 8 e 9)**. O LinGreedy atua com uma estratégia de exploração ϵ-greedy, enquanto o LinUCB direciona a seleção de braços baseada em limites superiores de confiança sobre os contextos lineares fornecidos pelas *embeddings* (*upper confidence bounds*). Para quantificar o desempenho dos algoritmos incrementais, foram calculados a HR@N e o NDCG@N,previamente explicados em maiores detalhes na Seção 5.1.4.

5.3.3 Resultados

Os resultados obtidos encontram-se apresentados na Figura 18, que ilustra o NDCG acumulado das *embeddings* ao longo das dez janelas temporais, sob a atuação do agente LinUCB. Adicionalmente, as Tabelas 8 e 9 sumarizam os resultados referentes à última janela, a qual já incorpora, inerentemente, as informações de todas as janelas anteriores. Para facilitar a leitura e a análise dos resultados, as células das tabelas foram formatadas em escala de cinza, em que tons mais escuros representam os melhores desempenhos em cada conjunto de dados.

Os agentes LinGreedy e LinUCB apresentaram comportamentos muito similares em relação aos resultados obtidos, comportamento esperado para este tipo de avaliação (PIRES et al., 2025). Desta forma, a análise dos resultados foca exclusivamente nos valores atingidos pelo LinUCB. Entretanto, os resultados relativos ao LinGreedy podem ser consultados no Apêndice C.2.

Adicionalmente, a análise se restringe apenas a métodos baseados em *embeddings* neurais. Métodos de fatoração de matrizes, como o ALS e o BPR, foram desconsiderados na comparação, pois são modelados para gerar representações estáticas de usuários e itens para predição de forma conjunta. Uma vez que os métodos incrementais geram representações de usuários dinamicamente em tempo de inferência, comparar as diferentes abordagens criaria uma falsa equivalência de desempenho.

Os resultados obtidos mostram uma superioridade dos métodos temporais, especialmente no TAI2Vec-Disc, que superou o Item2Vec e o SeqI2V em todos os cenários testados. Isso inclui a base de dados ML-1M, que havia apresentado resultados inferiores em experimentos anteriores. Essa melhora pode ser atribuída à alimentação contínua de dados de forma parcial, o que mitiga o impacto da alta densidade de interações característica da base. Ao simular o fluxo de dados de forma mais próxima a um cenário real, o algoritmo TAI2Vec demonstra maior robustez, de forma similar ao que foi visto nos experimentos estáticos.

Tabela 8 – Resultados do LinUCB sob a métrica Hit Rate

Conjunto de dados	Item2Vec	SeqI2V	TAI2Vec-Cont	TAI2Vec-Disc
Amazon-Beauty	0,0166	0,0184	0,0189	0,0197
Amazon-Books	0,0211	0,0192	0,0196	0,0218
Amazon-Games	0,0182	0,0199	0,0186	0,0265
BestBuy	0,0235	0,0236	0,0261	0,0243
CiaoDVD	0,0297	0,0307	0,0316	0,0364
ML-100k	0.0823	0.0738	0,0874	0,0903
ML-1M	0,0286	0,0299	0,0385	0,0405
RetailRocket	0,0395	0,0455	0,0535	0,0435

Como evidenciado pela Tabela 10, a análise do ganho percentual revela a clara superioridade das abordagens que incorporam a dimensão temporal de forma explícita. O modelo TAI2Vec-Disc consolidou-se como o melhor desempenho geral, apresentando um ganho médio de 19,74% em NDCG e 19,43% em taxa de acerto em relação ao *benchmark* Item2Vec. Mesmo o

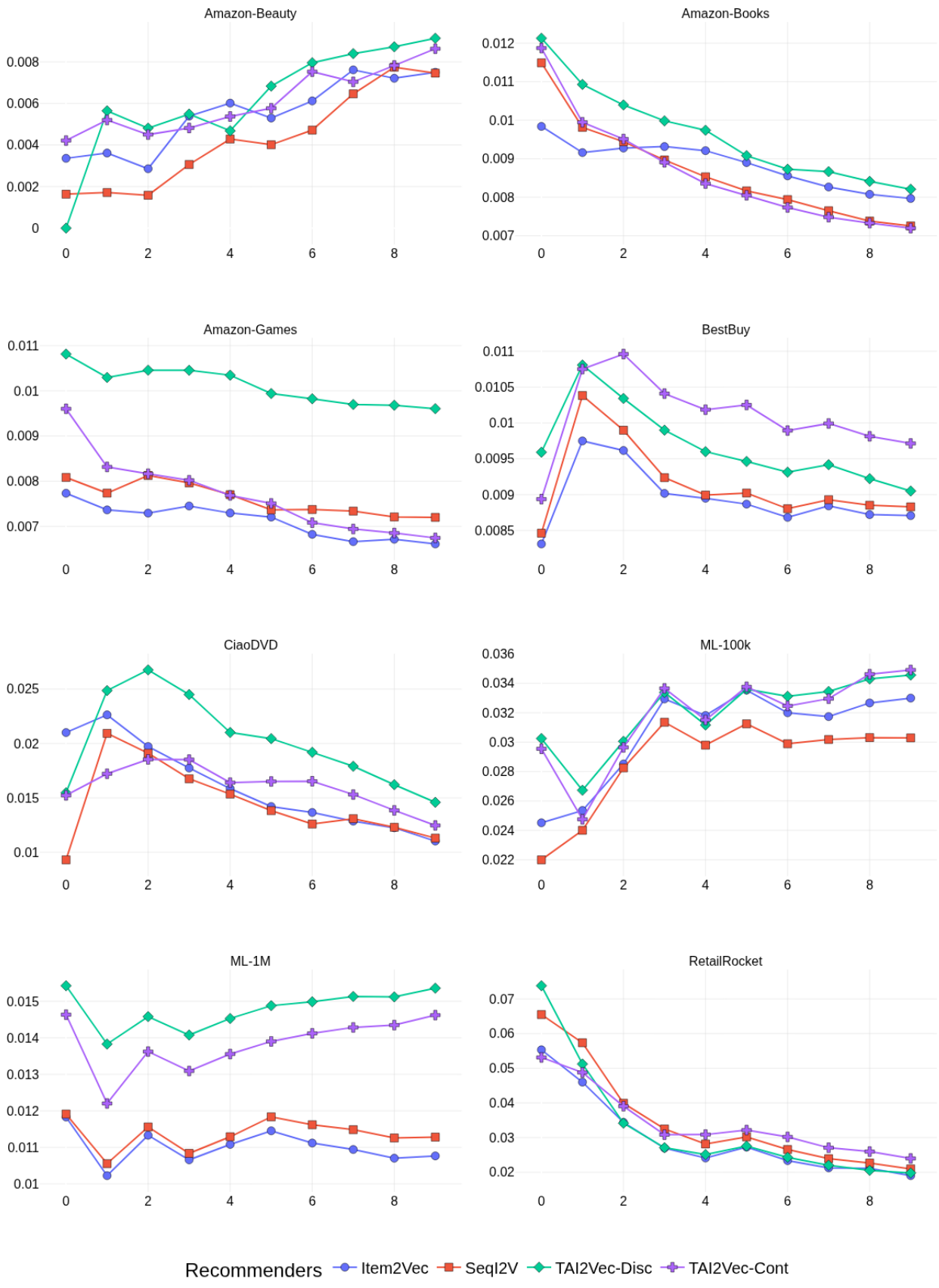


Figura 18 – NDCG por conjunto de dados pelo LinUCB

Tabela 9 – Resultados do LinUCB sob a métrica NDCG

Conjunto de dados	Item2Vec	SeqI2V	TAI2Vec-Cont	TAI2Vec-Disc
Amazon-Beauty	0,0075	0,0075	0,0086	0,0091
Amazon-Books	0,008	0,0073	0,0072	0,0082
Amazon-Games	0,0066	0,0072	0,0067	0,0096
BestBuy	0,0087	0,0088	0,0097	0.009
CiaoDVD	0,011	0,0113	0,0125	0,0146
ML-100k	0,033	0,0303	0,0349	0.0346
ML-1M	0,0108	0,0113	0,0146	0,0154
RetailRocket	0,019	0,021	0,024	0,0198

Tabela 10 – Ganho percentual sob Item2Vec para as métricas NDCG@10 e Hit Rate@10 em recomendação não incremental pelo agente LinUCB

	NDCG			**Hit Rate**		
Conjunto de dados	TAI2Vec-Cont	TAI2Vec-Disc	SeqI2V	TAI2Vec-Cont	TAI2Vec-Disc	SeqI2V
Amazon-Beauty	+15,17%	+21,90%	-0,47%	+14,33%	+18,88%	+11,11%
Amazon-Books	-9,66%	+3,02%	-8,93%	-7,05%	+3,25%	-9,06%
Amazon-Games	+2,02%	+45,26%	+8,90%	+2,38%	+45,69%	+9,48%
BestBuy	+11,57%	+3,94%	+1,40%	+10,92%	+3,42%	+0,34%
CiaoDVD	+13,07%	+32,33%	+2,49%	+6,60%	+22,64%	+3,30%
ML-100k	+5,81%	+4,74%	-8,20%	+6,24%	+9,76%	-10,38%
ML-1M	+35,79%	+42,64%	+4,77%	+34,55%	+41,70%	+4,38%
RetailRocket	+26,07%	+4,10%	+10,30%	+35,32%	+10,12%	+15,08%
Média	12,48%	19,74%	1,28%	12,91%	19,43%	3,03%
Mediana	12,32%	13,32%	1,95%	8,76%	14,50%	3,84%

TAI2Vec-Cont, embora ligeiramente inferior à sua versão discretizada, manteve uma vantagem robusta com médias de ganho superiores a 12%, superando consistentemente o SeqI2V, que obteve ganhos modestos (média de 1,28% no NDCG).

Para validar os resultados apresentados, foi realizado um teste de Friedman nos valores de NDCG@10 em cima da última janela temporal ($p = 0.001279$), indicando a existência de diferenças entre os algoritmos avaliados. A partir desse resultado, aplicou-se o teste pós-hoc de Bonferroni-Dunn para identificar quais comparações entre os métodos apresentam evidências de distinção estatística.

Considerando uma diferença crítica de $CD = 3,29$, observou-se que a variante TAI2Vec-Disc apresentou valores superiores aos obtidos por SeqI2V e Item2Vec, com diferenças médias de 3,50 e 3,88, respectivamente, ambas acima da diferença crítica. Por outro lado, a variante TAI2Vec-Cont não apresentou evidências de distinção em relação a SeqI2V e Item2Vec, com diferenças de 2,13 e 2,50, valores inferiores à diferença crítica estabelecida. Assim, pode-se afirmar que a variação discreta do TAI2Vec é estatisticamente superior aos métodos *baselines*, mas não há evidências estatísticas de que a variação contínua mantém o mesmo comportamento.

Com base nos dados apresentados, é possível responder parcialmente duas das perguntas de pesquisa que motivaram este estudo:

- **P1: Em que medida a integração explícita de intervalos temporais entre interações durante o treinamento de modelos de embeddings supera o desempenho de modelos puramente estáticos em métricas de recomendação Top-N?**

  A integração explícita de intervalos temporais supera os modelos estáticos ao permitir que a arquitetura capture a evolução dinâmica do comportamento do usuário, com ganhos médios superiores a 19%. Ao analisar a mediana do ganho percentual em comparação aos resultados do experimento offline (não incremental), evidencia-se que algoritmos incrementais, que já possuem informações temporais implícitas em sua composição, beneficiam-se de forma mais significativa do que algoritmos em lote ao processarem dados temporais provenientes das *embeddings*.

- **P2: Como a modelagem dinâmica de contextos temporais personalizados ao comportamento individual dos usuários impacta a qualidade das recomendações em comparação a métodos gerais que utilizam janelas de tempo globais e estáticas?**

  A modelagem baseada no comportamento individual do usuário superou significativamente os métodos de janelas globais e estáticas. O desempenho superior do TAI2Vec-Disc, com ganho médio de 19,74% no NDCG, demonstra que capturar o ritmo específico de cada usuário é mais eficaz para a qualidade das recomendações do que o uso de limiares globais.

- **P3: Protocolos experimentais offline tradicionalmente utilizados na literatura para avaliação de recomendadores incrementais são capazes de mensurar com assertividade a evolução do aprendizado sob um fluxo contínuo de dados?**

  O protocolo experimental proposto é válido para mensurar a evolução do aprendizado sob um fluxo contínuo de dados, demonstrando que a atualização das *embeddings* a cada partição sequencial funciona de maneira eficaz ao refinar as representações dos itens dinamicamente. Além de proporcionar uma boa base de comparação entre diversos métodos de representação do usuário.

Em síntese, observa-se que os resultados do cenário incremental mantêm um padrão de desempenho similar aos obtidos nos experimentos não incrementais, evidenciando a capacidade do TAI2Vec de gerar representações semanticamente significativas e manter consistência nos resultados.

Entretanto, é importante ressaltar que as avaliações *offline*, mesmo quando segmentadas em janelas temporais, tendem a favorecer estratégias estritamente gulosas. Como o ambiente de teste é baseado em *logs* históricos estáticos, o protocolo de avaliação não consegue capturar o valor real da exploração, que seria a descoberta de novos itens ou a adaptação a mudanças súbitas de tendência em tempo real. Dessa forma, seguindo o mesmo comportamento da avaliação não incremental, as métricas acabam refletindo a capacidade do modelo de replicar o passado contido nos *logs* de avaliação, como evidenciado por Pires et al. (2025). Em cenários

de avaliação mais complexos, como a descoberta de novos gostos do usuário, é recomendado o uso de estratégias de avaliação *off-policy* ou de ambientes simulados.

5.4 Simuladores

A avaliação online por Testes A/B mede a capacidade real do sistema de gerar recomendações relevantes atavés de interação direta com usuários em produção (LI et al., 2010b). No entanto, sua implementação é complexa, custosa e oferece riscos à experiência do usuário, caso modelos inferiores sejam implantados. Simuladores buscam mitigar esses riscos ao fornecer um ambiente controlado para experimentação prévia, permitindo que novas estratégias sejam validadas sem expor a base de usuários real a possíveis falhas.

Entre os simuladores propostos na literatura destacam-se o **Virtual-Taobao**, baseado em GANs e imitação multiagente (SHI et al., 2018); o **RecSim** e sua extensão **RecSim NG**, que oferecem ambientes configuráveis e suporte a aprendizado de políticas e cenários multiagente (IE et al., 2019a; MLADENOV et al., 2021); o **RecoGym**, estruturado no padrão OpenAI Gym e focado em recomendações via *bandits* (ROHDE et al., 2018); o ***Concept-Drift Stream Generator***, voltado à geração sintética de fluxos com desvio de conceito abrupto ou gradual (CAROPRESE et al., 2025); e o **KuaiSim**, um simulador orientado a dados para recomendação em sessões e listas, com modelagem explícita do retorno do usuário (ZHAO et al., 2023). Ainda que sejam destinados à avaliação contínua e online, a maioria desses simuladores negligencia informações temporais fundamentais, como o momento das interações e os intervalos entre consumos, sendo o KuaiSim uma das poucas exceções com maior foco nesse aspecto.

Desta maneira, o KuaiSim (ZHAO et al., 2023) foi adotado como plataforma de avaliação. Tendo sido construído sobre o conjunto de dados KuaiRand, uma base de vídeos curtos da plataforma Kuaishou que se destaca por conter interações não enviesadas, coletadas via exposição aleatória, é possível gerar uma modelagem mais fiel das preferências dos usuários.

Essa abordagem orientada a dados oferece dois benefícios principais para a validação da hipótese: *(i)* permite que as *embeddings* sejam treinadas sobre a mesma base de dados utilizada para modelar as regras do simulador; e *(ii)* possibilita, de forma fácil, a integração dos modelos temporais de *embeddings* através da substituição da representação vetorial de itens nativas do simulador.

5.4.1 Funcionamento do KuaiSim

A arquitetura do KuaiSim é composta por três módulos interdependentes que, juntos, reproduzem a jornada de interação do usuário: resposta imediata, saída da sessão e retenção.

- **Módulo de Resposta Imediata:** Atua como o núcleo do simulador ao estimar a reação do usuário para cada recomendação. O módulo é treinado por via de um modelo supervisi-

onado que processa atributos dos itens, dos usuários, e o histórico de interações através de um *Transformer*. Esse processo gera um estado do usuário que, combinado à representação do item candidato em uma rede neural, calcula a probabilidade de múltiplos *feedbacks* simultâneos como cliques, curtidas, comentários e visualizações prolongadas. Cada tipo de ação possui um valor de recompensa associado (peso w_f), baseado na relevância e frequência do sinal no conjunto de dados real. A recompensa do item é a soma ponderada das ocorrências desses *feedbacks*, permitindo que o simulador diferencie uma interação de menor relevância (como apenas clicar) de uma interação de significativa (como curtir e seguir o autor).

- **Módulo de Saída da Sessão:** Determina a duração de cada sessão ao traduzir o acúmulo de estímulos em uma decisão de abandono. O simulador utiliza uma variável de paciência (*temper*) que é atualizada dinamicamente a cada passo t seguindo a regra:

$$\text{temper}_{t+1} = \text{temper}_t + \text{clip}(\bar{r}_t - 2, [-2, 0])$$

onde $\bar{r}_t$ representa a recompensa média obtida pelo *slate* no passo atual.

Essa formulação implica que, se a recompensa for baixa, o valor da paciência decresce; se for alta ($\bar{r}_t \geq 2$), suaviza o decréscimo; no entanto, a variável nunca aumenta. Esse mecanismo busca simular a perda de interesse do usuário, encerrando a sessão assim que a paciência atinge um limiar pré-determinado.

- **Módulo de Retenção:** Fecha o ciclo de simulação ao estimar o intervalo temporal (δ_t) entre sessões consecutivas. Este componente avalia o impacto das recomendações a longo prazo através de um classificador sequencial. Um *Transformer* analisa a sequência de sessões passadas e os intervalos anteriores para capturar tendências de engajamento. Uma rede neural profunda gera uma distribuição de probabilidade sobre os possíveis dias de retorno, da qual o valor de δ_t é amostrado.

O fluxo do simulador funciona como um ciclo contínuo, onde, a partir de recomendação gerada pelo agente de MABs, o Módulo de Resposta Imediata gera o feedback instantâneo do usuário, que é então convertido em uma recompensa para o Módulo de Saída, controlando diretamente o tempo de permanência na sessão atual através do parâmetro de paciência. Uma vez encerrada a sessão, o acúmulo dessas interações é processado pelo Módulo de Retenção, que amostra o intervalo δ_t para o retorno do usuário. O tempo de retorno funciona como o passo final do ciclo, podendo ser usado para mensurar como as recomendações em uma sessão impactam o engajamento a longo prazo. Logo em seguida o ciclo é reiniciado até atingir um valor pré-determinado de iterações.

Durante o ciclo, a atualização dos algoritmos de *bandits* ocorre de forma online e contínua, de forma semelhante ao protocolo de aprendizado incremental. Assim que o ambiente processa a resposta do usuário e devolve a recompensa calculada para o conjunto de itens recomendados, o agente utiliza esses sinais para atualizar seu modelo interno e refinar sua política

de seleção. Esse processo de ajuste ocorre a cada passo da interação, garantindo que o algoritmo incorpore o feedback imediato da sessão para adaptar as escolhas futuras.

5.4.2 Adaptações no KuaiSim e protocolo experimental

A validação da hipótese central deste trabalho exigiu uma reestruturação do ecossistema de simulação KuaiSim, de modo que o impacto das diferentes arquiteturas de *embeddings* pudesse ser isolado e mensurado de forma justa.

A adaptação se deu no conjunto de dados KuaiRand, que serve de base para a construção das regras do simulador. O conjunto foi submetido a um ordenamento cronológico e segmentado em uma partição de 4:1:5. A primeira parcela dessa divisão foi destinada exclusivamente ao aprendizado das *embeddings* dos itens, sendo submetida a um novo particionamento de 80% para treinamento e 20% para validação, sobre os quais foi conduzida uma busca em grade para ajuste dos hiperparâmetros detalhados na seção 5.1.3. Uma vez encontrada a melhor combinação, as partições de treino e validação foram concatenadas para um retreino final, gerando as *embeddings* definitivas que passam a representar os itens dentro do simulador.

As *embeddings* originais do KuaiSim são compostas por características qualitativas e quantitativas (tais como categoria do vídeo, duração e *tags*), transformadas em vetores contínuos durante o pré-processamento e alimentadas para o treinamento do Módulo de Resposta Imediata. Na adaptação proposta, foi removida a dependência desses metadados brutos. Agora, qualquer informação processada pelo recomendador provém estritamente dos algoritmos de embedding avaliados. É importante notar que, nesta arquitetura, as representações dos usuários permanecem inalteradas.

A segunda metade do banco de dados foi reservada ao treinamento do simulador. No entanto, como o simulador também deriva suas regras de recompensa e comportamento a partir das características latentes dos itens e usuários, qualquer alteração na forma como os itens são representados modifica a própria lógica operacional do ambiente. Assim, para cada algoritmo de geração de *embeddings* testado, foi necessário instanciar e treinar a sua própria iteração do simulador. Essa abordagem foi adotada para assegurar que o agente e o ambiente funcionem sob o mesmo espaço semântico, garantindo as seguintes características:

- **Alinhamento do Espaço Vetorial**: Ao treinar o simulador com as mesmas *embeddings* do agente, evita-se que o ambiente gere recompensas baseadas em relações de proximidade que o modelo não reconhece.

- **Isolamento de Variáveis**: O uso de um mapeamento vetorial único para o simulador e o recomendador assegura que variações nas recompensas reflitam apenas a qualidade das *embeddings*, eliminando o ruído de uma possível falha de comunicação entre agente e ambiente.

Além da adaptação na representação dos itens, todos os demais componentes do simulador seguem o funcionando da mesma forma como foi estabelecido na Seção 5.4.1. O fluxo completo da avaliação pode ser observada na Figura 19.

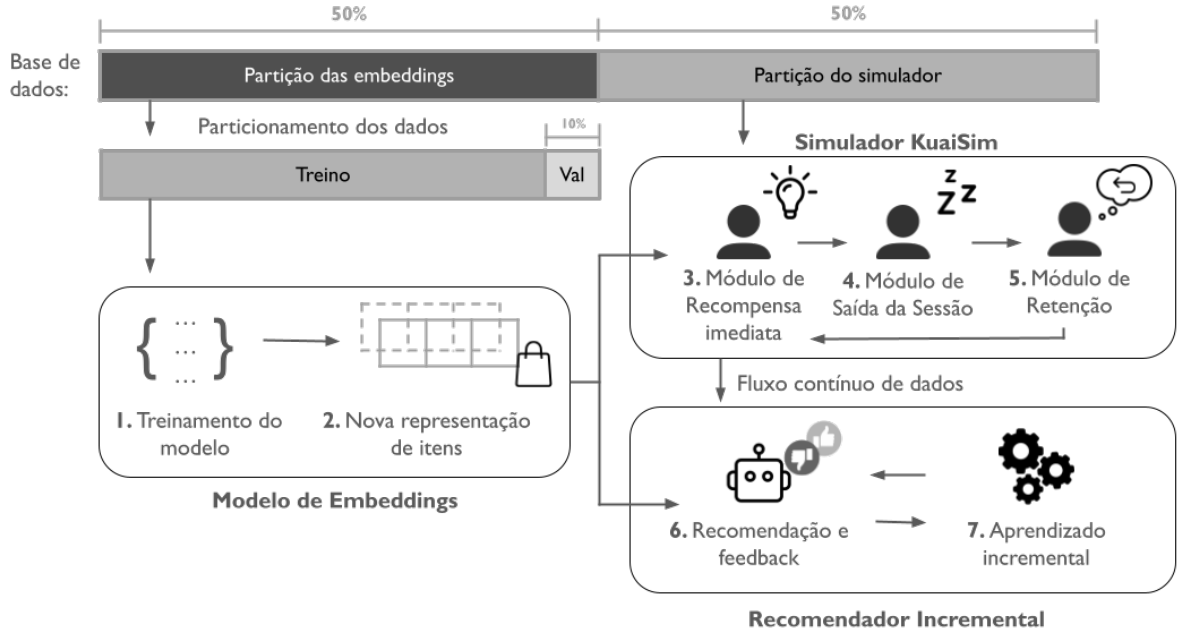


Figura 19 – Diagrama do protocolo experimental online usando simuladores.

Uma vez que os algoritmos de MABs utilizam as representações dos itens como contexto para suas decisões, os agentes foram modificados para operar de forma integrada a esse ecossistema. Assim, cada algoritmo é instanciado especificamente para processar a representação vetorial da rodada de testes em questão, interagindo simultaneamente com o simulador e com o modelo de *embeddings* correspondente. Ao contrário do protocolo experimental apresentado na Seção 5.3.2, os Mabs não precisam passar por uma fase de aquecimento, pois o fluxo de dados gerado pelo simulador é dinâmico e roda quantos passos forem necessários até os modelos atingirem uma convergência inicial.

5.4.3 Métricas do simulador

Para a avaliação online do desempenho dos algoritmos incrementais via simulador, foram usadas métricas de assertividade, retenção e diversidade. Em relação à assertividade, foi empregada a recompensa média do feedback do Módulo de Resposta Imediata, onde, para cada recomendação, é retornada uma série de possíveis ações tomadas por um usuário (como cliques, curtidas, visualizações). Cada interação gera uma recompensa r_i calculada como a soma ponderada das ações pelos pesos w_f:

$$\text{Recompensa Média} = \frac{\sum_{i=1}^{N} r_i}{N} \tag{30}$$

em que N é o número total de interações observadas. Esta métrica serve como o sinal primário para o ajuste das políticas dos agentes, permitindo observar a eficácia da política em converter recomendações em engajamento explícito.

Para avaliar além da resposta imediata, são empregadas métricas de engajamento que avaliam o impacto das recomendações na retenção e na profundidade do consumo. O Tamanho Médio da Sessão mede o número médio de interações por sessão, sendo diretamente influenciado pelo Módulo de Saída da Sessão, uma vez que a permanência do usuário é ditada pelo consumo da paciência (*temper*) em resposta à qualidade dos itens:

$$\text{Tamanho Médio da Sessão} = \frac{1}{S}\sum_{s=1}^{S} n_s \tag{31}$$

em que S é o número de sessões e n_s o número de interações na sessão s.

Complementarmente, o Tempo Médio de Retorno avalia a frequência com que os usuários retornam ao sistema através da distribuição de intervalos temporais (δ_t) após o fim de uma sessão.

$$\text{Tempo Médio de Retorno} = \frac{1}{N}\sum_{i=1}^{N} \delta_t^{(i)} \tag{32}$$

em que N é o número total de retornos observados e δ_t é o intervalo amostrado pelo simulador entre sessões consecutivas.

Como é possível observar, todas as métricas apresentadas estão diretamente conectadas ao processo iterativo do simulador. Embora a assertividade possa ser medida em cenários *offline* e seu viés mitigado de certa forma, simuladores são os únicos que conseguem reproduzir a dinâmica causal e o impacto do histórico de recomendações no estado futuro do usuário, algo que previamente só seria possível com o feedback imediato de usuários em um sistema online rodando em produção.

Além das métricas do próprio simulador, serão observadas as métricas de diversidade Diversidade Intra-Lista (ILD) e Cobertura, estas duas que são explicadas em maiores detalhes na Seção 5.1.4.

5.4.4 Resultados

Os resultados a seguir foram segmentados para diferentes valores de $N \in \{1, 5, 10, 20\}$ no contexto de recomendação top-N. O desempenho de cada algoritmo de *embedding*, em relação às métricas de assertividade detalhadas na Seção 5.4.3, encontra-se ilustrado nas Figuras 20 e 27, que apresentam os resultados para os agentes LinGreedy e LinUCB, respectivamente. Em relação à Recompensa Média e ao Tamanho Médio da Sessão, valores mais elevados são preferíveis, indicando maior engajamento ou precisão. Em contrapartida, para o Tempo Médio

de Retorno, resultados menores são mais desejados, pois indicam maior poder de retenção do usuário.

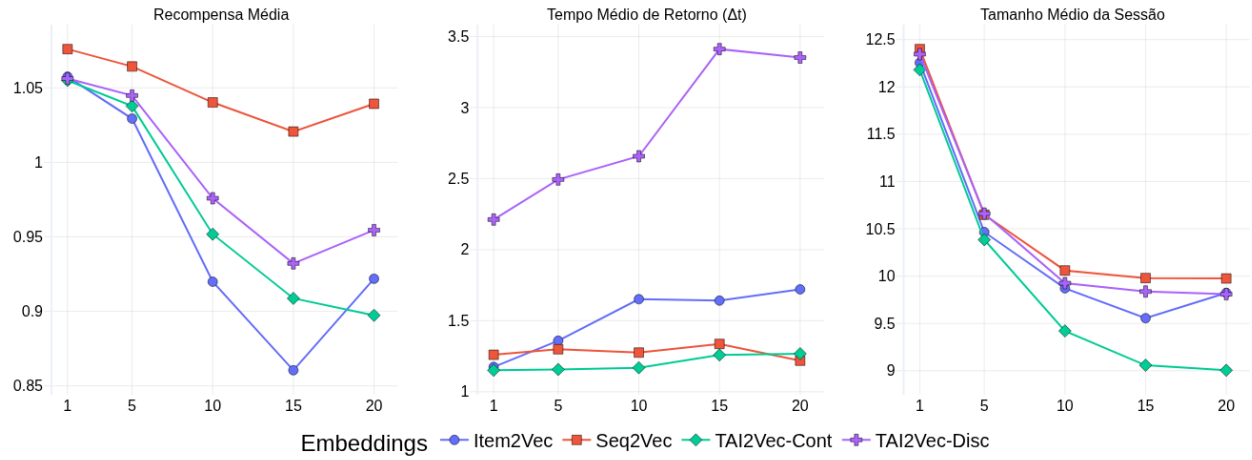


Figura 20 – Métricas de assertividade no LinGreedy

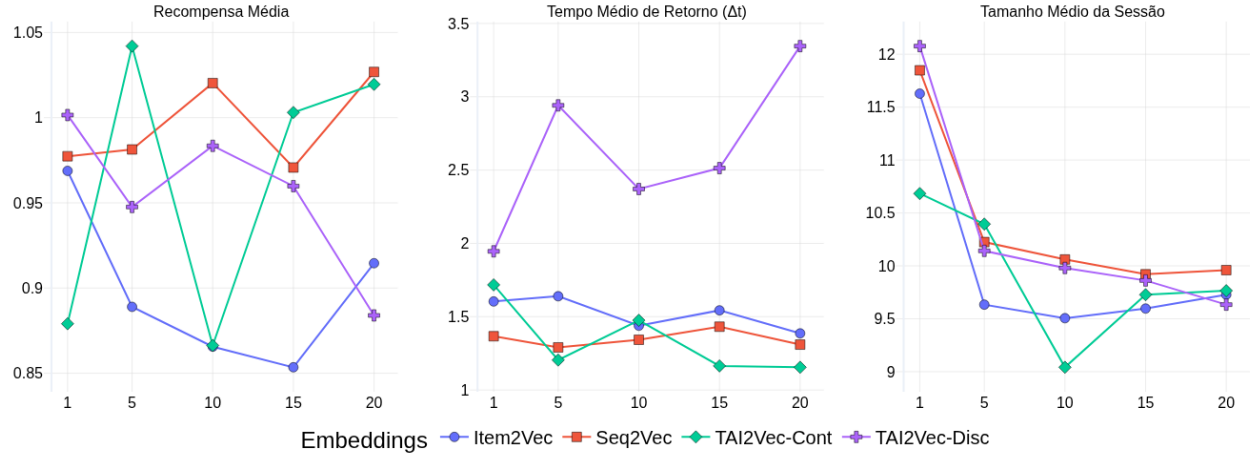


Figura 21 – Métricas de assertividade no LinUCB

De forma geral, foi vista uma superioridade dos algoritmos temporais em relação ao Item2Vec, mostrando resultados melhores especialmente na Recompensa média. No entanto, o método SeqI2V, que trata atributos temporais como limites estáticos, superou os métodos dinâmicos tanto em recompensa média como no tamanho médio da sessão do usuário.

Uma explicação plausível para esse fenômeno reside na construção do simulador sobre a base de dados KuaiRand. Por utilizar exposição aleatória na geração dos *logs* históricos, esse ambiente mitiga significativamente os vieses da avaliação offline, o que beneficia a simulação. No entanto, essa metodologia de coleta prejudica especificamente os modelos TAI2Vec, que pressupõem uma causalidade direta entre escolhas imediatas que pode não ocorrer no KuaiRand, dado que a exibição dos itens não foi pautada na relevância para perfis específicos. Estas características distanciam a base de uma jornada de consumo orgânica, cenário em que o TAI2Vec apresenta melhores resultados.

A curto prazo, essa falta de correlação orgânica gera ruídos nas coocorrências do TAI2Vec, que calcula a importância das relações baseando-se estritamente na frequência de consumo in-

dividual, resultando em janelas efetivas de contexto reduzidas. Já no SeqI2V esse efeito é mitigado, cujo limiar de corte estático pré-definido de meses ou anos gera grupos temporais muito maiores, diluindo o impacto desses ruídos.

É importante notar, entretanto, que mesmo com essa maior suscetibilidade ao ruído do simulador, todos os modelos temporais continuam superando consistentemente o Item2Vec. Isso demonstra que a longo prazo, apesar da sensibilidade ao ruído, a integração de fatores temporais gera representações de contexto superiores à abordagem puramente estática.

Além disso, ao observar essas métricas, é possível observar como a dinâmica de análise multifacetada dos simuladores é capaz de revelar nuances que métricas offline ignorariam. Por exemplo, ao comparar TAI2Vec-Disc e TAI2Vec-Cont:

- O TAI2Vec-Disc obteve resultados de feedback imediato superiores. Contudo, ele exibe o maior Tempo de Retorno entre os modelos, sugerindo que, embora a recomendação seja precisa no curto prazo, ela pode não ser eficaz em sustentar o interesse recorrente do usuário e motivar seu regresso ao sistema.

- Em contrapartida, o TAI2Vec-Cont registrou o menor tempo de retorno, embora com sessões ligeiramente mais curtas, indicando uma dinâmica de visitas frequentes, mas de menor duração.

O comportamento dos algoritmos em relação à variedade das recomendações é apresentado nas Figuras 22 e 23, que detalham a Diversidade Intra-Lista (ILD) e a Cobertura para os agentes LinGreedy e LinUCB, respectivamente.

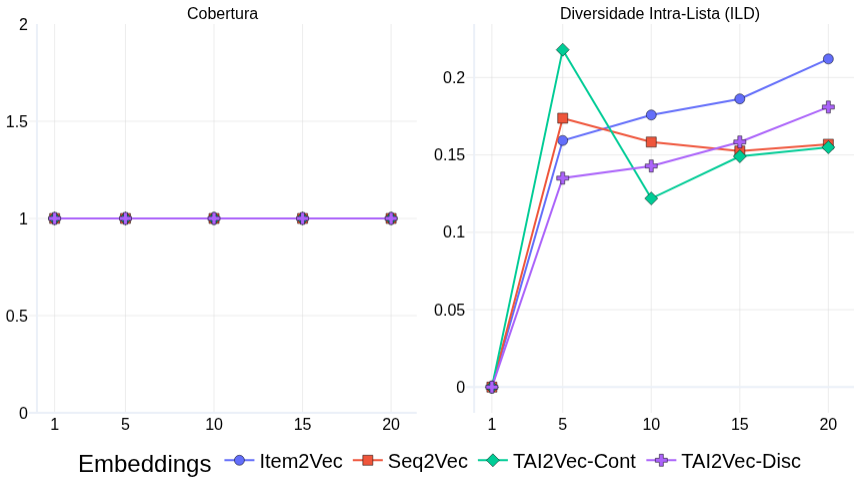


Figura 22 – Métricas de diversidade no LinGreedy

O LinGreedy atingiu 100% de cobertura em todos os modelos. Esse comportamento é esperado, pois sua probabilidade ϵ de exploração garante a visita completa do catálogo a longo prazo. No entanto, o Item2Vec apresentou a maior Diversidade Intra-Lista, o que se justifica através do comportamento dos métodos temporais, que diminuem o escopo contextual

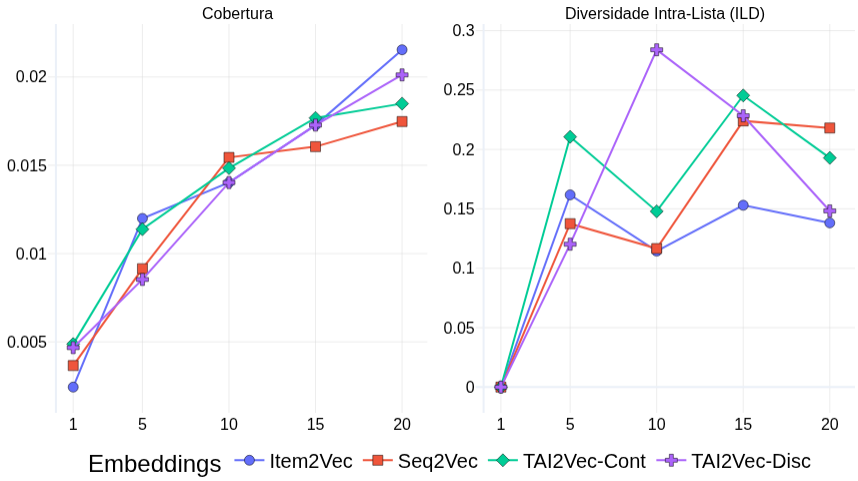


Figura 23 – Métricas de diversidade no LinUCB

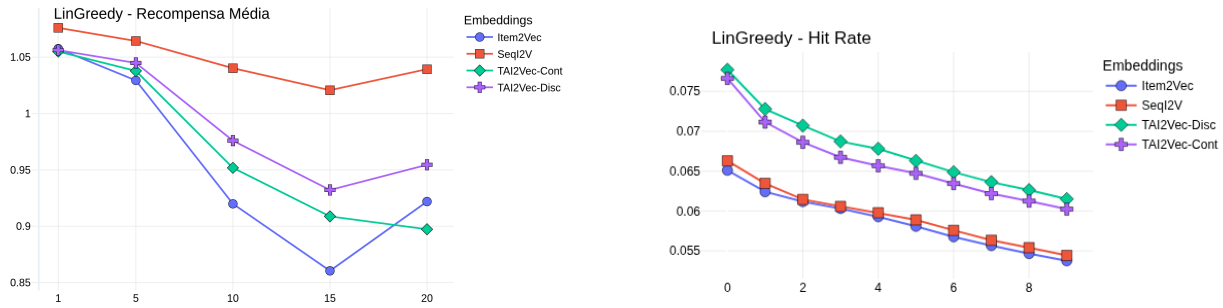
para priorizar apenas relações mais relevantes, enquanto o modelo estático mantém conexões globais e, consequentemente, mais dispersas. Para o LinUCB, os resultados de diversidade mostraram-se inconclusivos, indicando que, para este agente, a representação dos itens afeta a assertividade, mas não altera a dinâmica de exploração do catálogo.

Investigando além das métricas dos simuladores, um dos objetivos centrais deste experimento é averiguar em que medida a utilização do KuaiSim é capaz de mitigar os vieses de logs encontrados na avaliação offline. No entanto, surgem desafios metodológicos para realizar essa comparação de forma direta.

Conforme detalhado na Seção 5.4.2, a base de dados KuaiRand foi dividida em duas metades para a validação da hipótese: a primeira dedicada exclusivamente ao treinamento das embeddings e a segunda à definição das regras de comportamento do simulador. Somado a isso, como discutido na Seção 5.4.3, o KuaiSim utiliza métricas de recompensa que diferem das métricas de assertividade aplicadas aos métodos offline (incrementais e não incrementais). Essa discrepância dificulta a análise sobre se o simulador cumpre efetivamente o papel de mitigar vieses.

Na tentativa de estabelecer um paralelo entre os métodos online e offline, realizou-se uma avaliação offline sobre a partição originalmente destinada ao simulador. O protocolo seguido foi o mesmo descrito na Seção 5.3, alterando-se apenas a amostra dos dados: utilizaram-se as mesmas embeddings treinadas na primeira metade, mas validadas na partição subsequente. Embora as métricas offline não sejam idênticas à recompensa média do simulador, ambas convergem no objetivo de medir a assertividade, justificando a viabilidade desta comparação experimental.

A Figura 24 apresenta a comparação entre as métricas de assertividade utilizando as mesmas bases de dados e a mesma estratégia de exploração e aprofundamento. No cenário de avaliação offline, o eixo X representa as janelas temporais detalhadas na Seção 5.3, enquanto na avaliação online, o eixo X indica o valor de N na recomendação Top-N. Notavelmente, os algoritmos TAI2Vec apresentam um desempenho superior na avaliação 24 (offline) em comparação



(a) Online (b) Offline

Figura 24 – Comparação entre métricas de assertividade em cenários online e offline em recomendação incremental pelo LinGreedy.

à online, ampliando sua vantagem sobre o Item2Vec e superando o SeqI2V.

O contraste entre a proximidade de desempenho entre o Item2Vec e os modelos temporais no simulador com a vantagem destes no cenário offline sugere que a dinâmica de exploração influencia os resultados. No ambiente simulado, a diversidade intra-lista superior do Item2Vec favorece a convergência do agente, permitindo uma varredura mais dispersa do catálogo. Esse comportamento parece mascarar a menor precisão temporal do modelo estático frente aos modelos temporais na avaliação offline. Quanto ao SeqI2V, seu comportamento no KuaiSim reflete limitações já observadas em bases como o MovieLens. Em ambientes onde interações densas são fragmentadas, como é o caso da avaliação incremental offline, o uso de limiares de corte estáticos mostra-se insuficiente para capturar contextos significativos.

A diferença de comportamento para o SeqI2V entre a avaliação online e offline conduzida sobre uma mesma base de dados levanta um questionamento sobre a capacidade do simulador KuaiSim de avaliar de forma justa os métodos comparados. Ao calcular uma probabilidade de feedback e, consequentemente, uma recompensa média com base em todo o histórico (real e simulado) do usuário, a ferramenta não é capaz de capturar mudanças temporais e desvios de conceito. Tal particularidade justifica a melhora expressiva do SeqI2V, que incorpora o tempo na representação gerada—técnica comprovadamente superior à abordagem agnóstica temporal—mas sem realizar quebras capazes de filtrar padrões e comportamentos recentes, como é feito pelo TAI2Vec. O KuaiSim se mantém como uma das maneiras mais seguras e flexíveis de se avaliar algoritmos de *bandits* de forma online, mas tais comportamentos evidenciam um promissor caminho de pesquisa no estudo e desenvolvimento de simuladores de recomendação.

A partir dos resultados atingidos, podemos responder a 3 das perguntas de pesquisa previamente estabelecidas na introdução da dissertação:

- **P1: Em que medida a integração explícita de intervalos temporais entre interações durante o treinamento de modelos de embeddings supera o desempenho de modelos puramente estáticos em métricas de recomendação Top-N?**

  A integração temporal superou o modelo estático Item2Vec em todas as métricas de engajamento. Contudo, o modelo estático manteve a maior Diversidade Intra-Lista, mostrando que, embora a integração temporal aumente a assertividade, ela o faz ao custo de restringir o escopo contextual das recomendações.

- **P2: Como a modelagem dinâmica de contextos temporais personalizados ao comportamento individual dos usuários impacta a qualidade das recomendações em comparação a métodos gerais que utilizam janelas de tempo globais e estáticas?**

  A análise dos resultados indica que a modelagem dinâmica enfrenta desafios específicos quando aplicada a ambientes simulados baseados em logs de exposição aleatória, como o KuaiRand. Em contrapartida, o método de janelas globais e estáticas demonstrou um desempenho superior em quase todas as métricas. Para consolidar essas conclusões, trabalhos futuros devem focar na reprodutibilidade dos experimentos em diferentes bases de dados, testando se a eficácia da segmentação temporal estática se mantém em simuladores com regras de interação distintas.

- **P4: Como a adaptação de simuladores de larga escala permite a integração de novas representações vetoriais para uma avaliação versátil e livre de vieses de seleção?**

  Ao treinar o simulador e o agente sob o mesmo espaço semântico, este ambiente simulado revela nuances comportamentais, como a diferença entre feedback imediato e retenção de longo prazo, que seriam invisíveis em avaliações estáticas tradicionais. Trata-se de um grande benefício quando comparado à avaliação offline.

Capítulo 6

Conclusão

Esta dissertação propõe, desenvolve e avalia o TAI2Vec, um grupo de modelos de *embeddings* baseados em redes neurais concebido para integrar fatores temporais ao treinamento de representações vetoriais. Inspirado em uma arquitetura análoga ao Item2Vec, o modelo busca gerar representações latentes de itens que otimizam a tarefa de recomendação Top-N. O grande diferencial do TAI2Vec é como ele se ajusta a diferentes domínios e ritmos de consumo.

O objetivo principal deste trabalho é avaliar o desempenho dessas *embeddings* em cenários de recomendação incremental. Nesse contexto, as informações temporais são introduzidas implicitamente a algoritmos de Multi-Armed Bandits (MAB), que são capazes de se adaptar em tempo real a mudanças de comportamento, mas que, nativamente, não processam dados temporais além da ordem das iterações.

Embora a validação central da hipótese resida na recomendação incremental, os experimentos realizados em cenários online e offline constituíram uma contribuição valiosa para o trabalho. Foram detalhadas com profundidade metodologias de avaliação para cada cenário, com destaque para a avaliação incremental offline e por simuladores, os quais ainda pouco exploradas na literatura atual e ainda não possuem protocolos bem estabelecidos.

Os resultados demonstram a superioridade do TAI2Vec na maioria dos algoritmos testados, evidenciando que o modelo atinge seu desempenho pleno em cenários que mimetizam fielmente o comportamento orgânico de consumo da vida real. Em contrapartida, observou-se que a eficácia das representações latentes é sensível à integridade dos dados temporais, apresentando resultados menos expressivos em bases de dados onde essa dimensão é distorcida ou artificial. É o caso do conjunto de dados MovieLens, no qual os *timestamps* das interações muitas vezes não refletem o momento real do consumo, e do KuaiRand, que, apesar de possuir uma metodologia de coleta rigorosa para o treinamento de recomendadores, não replica com fidelidade os ritmos do cotidiano.

Por outro lado, em cenários que preservam a coerência temporal e, especialmente, em conjuntos de dados esparsos, a informação contextual fornecida pela integração do tempo às *embeddings* mostrou-se altamente eficaz. Nessas condições, o TAI2Vec compensou a escassez de interações ao utilizar o fator temporal como um referencial sólido para refinar as recomendações.

Os experimentos também destacam a natureza *model-agnostic* das representações geradas. A capacidade do TAI2Vec de elevar o desempenho de algoritmos distintos, desde o tradicional K-NN em avaliações estáticas até o LinUCB em contextos incrementais, comprova o alto poder de generalização da arquitetura.

Além disso, espera-se que as metodologias de avaliação detalhadas nesta dissertação também sirvam de inspiração para trabalhos futuros que busquem rigor estatístico e realismo em sistemas de recomendação dinâmicos. Nesse sentido, esta pesquisa reforça que o uso de simuladores é um caminho essencial para o desenvolvimento de métricas mais robustas, servindo como base para novas investigações sobre como avaliar o aprendizado interativo de forma segura e escalável.

6.1 Perspectivas futuras

Existem alguns pontos que ainda podem ser melhor explorados em relação à introdução de fatores temporais em recomendadores incrementais. Em especial, é possível expandir o escopo do trabalho para além do módulo de geração de *embeddings* e atuar diretamente na representação do estado do usuário. Especificamente, essa inserção pode ser feita de duas maneiras:

- **Integração explicita do tempo:** Embora as *embeddings* possam carregar dados temporais, essa informação costuma perder força quando misturada ao histórico. Para evitar essa diluição, propõe-se injetar atributos temporais (como dia e turno) de forma explícita no agente. Assim, o modelo aprende uma função-valor $Q(s, a, T)$, onde o tempo T atua como um influenciador direto na decisão, além do estado s e da ação a. Na prática, após o módulo de *embeddings* processar os itens consumidos, o estado do usuário é consolidado. Esse vetor de estado é então combinado aos atributos temporais antes de ser entregue ao agente, garantindo que a recomendação seja sensível ao momento exato da solicitação.

- **Ponderação Temporal do Histórico:** Atualmente, o estado do usuário é consolidado por meio de uma média aritmética dos itens consumidos, o que assume, implicitamente, que todos os eventos do histórico possuem o mesmo peso preditivo. Pretende-se substituir a média aritmética simples do estado do usuário por uma média ponderada temporal. Através de funções de decaimento, o modelo passará a priorizar interações recentes na hora de gerar sua representação.

Sobre possíveis melhorias do TAI2Vec, dois possíveis caminhos podem ser seguidos:

- ❏ **Explicabilidade do modelo:** Diferente dos modelos sequenciais tradicionais de "caixa-preta", as curvas de decaimento do TAI2Vec-Cont oferecem uma métrica interpretável do interesse atual do usuário. É planejado investigar como esses pesos temporais podem ser apresentados a usuários ou analistas para justificar recomendações específicas.

- ❏ **Arquiteturas Híbridas:** Será explorado também uma arquitetura unificada que combine a detecção de limites do TAI2Vec-Disc com a modelagem de decaimento do TAI2Vec-Cont. O objetivo é utilizar mecanismos de atenção para que o sistema aprenda, de forma automática, o equilíbrio ideal entre essas duas abordagens para cada usuário.

- ❏ **Comparação com novas arquiteturas:** Durante as avaliações sob o protocolo incremental, a análise restringiu-se a algoritmos que compartilhavam a mesma arquitetura de redes neurais. Essa escolha foi feita com o objetivo de isolar o impacto da introdução de fatores temporais na qualidade das recomendações. No entanto, é possível expandir esse escopo para incluir arquiteturas distintas, tais como modelos baseados em *Self-Attention* (e.g., SASRec (KANG; McAuley, 2018)) e redes neurais recorrentes (e.g., GRU4Rec (VASSøY et al., 2018)).

- ❏ **Integração do modelo de embeddings com metadados:** O tratamento de alguns problemas clássicos de recomendação, como o *cold-start* (WANG et al., 2021), acabou ficando de fora do escopo inicial deste projeto. No entanto, isso pode ser mitigado através da integração de metadados dos itens (categorias, descrições textuais ou *tags*) diretamente ao TAI2Vec. Espera-se que essa introdução ajude com a recomendação de itens novos que ainda não possuem iterações suficientes para terem uma representação fiel, além de elevar a qualidade e a precisão das recomendações de forma geral.

Quanto às etapas de avaliação, perspectivas futuras residem na ampliação da ferramenta KuaiSim e simuladores semelhantes, aprimorando suas funcionalidades para otimizar a análise de diferentes representações de usuários e itens. Complementarmente, sugere-se a realização de treinamentos em diversas bases de dados para testar a escalabilidade e a viabilidade prática do simulador.

Referências

ADOMAVICIUS, G.; TUZHILIN, A. Multidimensional recommender systems: A data warehousing approach. **Electronic Commerce**, Springer Science+Business Media, Berlin/Heidelberg, Germany, v. 2232, p. 180–192, 2001.

ADOMAVICIUS, G.; TUZHILIN, A. Context-Aware Recommender Systems. In: RICCI, F.; ROKACH, L.; SHAPIRA, B. (Ed.). **Recommender Systems Handbook**. Springer US, 2015. p. 191–226. ISBN 978-1-4899-7636-9 978-1-4899-7637-6. Disponível em: <https://link.springer.com/10.1007/978-1-4899-7637-6_6>.

AFSAR, M. M.; CRUMP, T.; FAR, B. Reinforcement learning based recommender systems: A survey. 1 2021. Disponível em: <http://arxiv.org/abs/2101.06286>.

AGGARWAL, C. C. **Recommender Systems**. Springer International Publishing, 2016. ISBN 978-3-319-29657-9 978-3-319-29659-3. Disponível em: <http://link.springer.com/10.1007/978-3-319-29659-3>.

AGRAWAL, S.; GOYAL, N. **Thompson Sampling for Contextual Bandits with Linear Payoffs**. arXiv. Disponível em: <http://arxiv.org/abs/1209.3352>.

AHN, Y.; RHEE, E.; LEE, J. Dual embedding with input embedding and output embedding for better word representation. v. 27, n. 2, p. 1091, 2022. ISSN 2502-4760, 2502-4752. Disponível em: <https://ijeecs.iaescore.com/index.php/IJEECS/article/view/29076>.

ALJUNID, M. F.; HUCHAIAH, M. D. An efficient hybrid recommendation model based on collaborative filtering recommender systems. **CAAI Transactions on Intelligence Technology**, IET, Hertfordshire, UK, v. 6, n. 4, p. 480–492, 2021.

AUER, P. Using confidence bounds for exploitation-exploration trade-offs. **J. Mach. Learn. Res.**, JMLR.org, v. 3, p. 397–422, mar 2003. ISSN 1532-4435.

AUER, P.; CESA-BIANCHI, N. Finite-time Analysis of the Multiarmed Bandit Problem. 2002.

BAGHI, V. et al. Improving ranking function and diversification in interactive recommendation systems based on deep reinforcement learning. In: **2021 26th International Computer Conference, Computer Society of Iran (CSICC)**. Tehran, Iran: IEEE, 2021. p. 1–7.

BALTRUNAS, L.; AMATRIAIN, X. Towards time-dependant recommendation based on implicit feedback. In: **Proceedings of the RecSys 2009 Workshop on Context-Aware**

Recommender Systems. New York, NY, USA: Association for Computing Machinery, 2009. (RecSys '09), p. 1–5.

BARKAN, O.; KOENIGSTEIN, N. Item2Vec: Neural item embedding for collaborative filtering. In: **IEEE 26th International Workshop on Machine Learning for Signal Processing**. Vietri sul Mare, Italy: IEEE, 2016. (MLSP 2016), p. 1–6.

BENGIO, Y. et al. A Neural Probabilistic Language Model. 2003.

BOBADILLA, J. et al. Recommender systems survey. **Knowledge-Based Systems**, v. 46, p. 109–132, 7 2013. ISSN 09507051.

BOUNEFFOUF, D.; RISH, I. **A Survey on Practical Applications of Multi-Armed and Contextual Bandits**. 2019. Disponível em: <http://arxiv.org/abs/1904.10040>.

BURKE, R. Hybrid recommender systems: Survey and experiments. **User modeling and user-adapted interaction**, Springer, v. 12, p. 331–370, 2002.

CAMPOS, P. G.; DÍEZ, F.; CANTADOR, I. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. **User Modeling and User-Adapted Interaction**, Springer, v. 24, p. 67–119, 2014.

CAROPRESE, L. et al. Modelling concept drift in dynamic data streams for recommender systems. **ACM Transactions on Recommender Systems**, Association for Computing Machinery, New York, NY, USA, v. 3, n. 2, p. 1–28, 2025.

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

CHANG, J. W. et al. Music recommender using deep embedding-based features and behavior-based reinforcement learning. Multimedia Tools and Applications, v. 80, n. 26-27, p. 34037–34064, 2019. ISSN 15737721.

CHEN, M. et al. Top-k off-policy correction for a REINFORCE recommender system. In: **Proceedings of the 12th ACM International Conference on Web Search and Data Mining**. Melbourne, Australia: Association for Computing Machinery, 2019. (WSDM '19), p. 456–464.

CHEN, X. et al. Generative adversarial user model for reinforcement learning based recommendation system. In: **Proceedings of the 36th International Conference on Machine Learning**. Long Beach, CA, USA: PMLR, 2019. (ICML 2019), p. 1818–1832.

CHENG, H.-T. et al. Wide & deep learning for recommender systems. In: **Proceedings of the 1st Workshop on Deep Learning for Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2016. (DLRS 2016), p. 7–10. ISBN 9781450347952. Disponível em: <https://doi.org/10.1145/2988450.2988454>.

DING, Y.; LI, X. **Time Weight Collaborative Filtering**. 2005.

DUDIK, M.; LANGFORD, J.; LI, L. Doubly robust policy evaluation and learning. In: **Proceedings of the 28th International Conference on International Conference on Machine Learning**. Madison, WI, USA: Omnipress, 2011. (ICML'11), p. 1097–1104.

DULAC-ARNOLD, G. et al. Deep reinforcement learning in large discrete action spaces. **arXiv**, v. 1512.07679, p. 1–11, 2015.

EKSTRAND, M. D. Collaborative Filtering Recommender Systems. **Foundations and Trends® in Human–Computer Interaction**, v. 4, n. 2, p. 81–173, 2011. ISSN 1551-3955, 1551-3963.

GAMA, J.; SEBASTIÃO, R.; RODRIGUES, P. P. On evaluating stream learning algorithms. **Machine Learning**, Springer Science+Business Media, Berlin/Heidelberg, Germany, v. 90, n. 3, p. 317–346, 2012.

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

GRBOVIC, M. et al. E-commerce in your inbox: Product recommendations at scale. In: **Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2015. (KDD '15), p. 1809–1818.

GUSAK, D. et al. Time to split: Exploring data splitting strategies for offline evaluation of sequential recommenders. In: **Proceedings of the Nineteenth ACM Conference on Recommender Systems**. [s.n.], 2025. p. 874–883. Disponível em: <http://arxiv.org/abs/2507.16289>.

HERLOCKER, J.; KONSTAN, J. A.; RIEDL, J. **An Empirical Analysis of Design Choices in Neighborhood-Based Collaborative Filtering Algorithms**. 2002. 287-310 p.

IE, E. et al. RecSim: A configurable simulation platform for recommender systems. **arXiv preprint**, p. 1–23, 2019.

IE, E. et al. Reinforcement learning for slate-based recommender systems: A tractable decomposition and practical methodology. 5 2019. Disponível em: <http://arxiv.org/abs/1905.12767>.

JATNIKA, D.; BIJAKSANA, M. A.; SURYANI, A. A. Word2Vec Model Analysis for Semantic Similarities in English Words. v. 157, p. 160–167, 2023. ISSN 18770509. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1877050919310713>.

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

KANG, W.-C.; McAuley, J. **Self-Attentive Sequential Recommendation**. arXiv, 2018. Disponível em: <http://arxiv.org/abs/1808.09781>.

KHUSRO, S.; ALI, Z.; ULLAH, I. Recommender Systems: Issues, Challenges, and Research Opportunities. In: KIM, K. J.; JOUKOV, N. (Ed.). **Information Science and Applications (ICISA) 2016**. Springer Singapore, 2016. v. 376, p. 1179–1189. ISBN 978-981-10-0556-5 978-981-10-0557-2. Disponível em: <http://link.springer.com/10.1007/978-981-10-0557-2_112>.

KO, H. et al. A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields. **Electronics**, v. 11, n. 1, p. 141, jan. 2022. ISSN 2079-9292.

KOREN, Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: **Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2008. (KDD '08), p. 426–434.

KOREN, Y. Collaborative filtering with temporal dynamics. v. 53, n. 4, p. 89–97, 2010. ISSN 0001-0782, 1557-7317. Disponível em: <https://dl.acm.org/doi/10.1145/1721654.1721677>.

KOREN, Y.; BELL, R.; VOLINSKY, C. Matrix factorization techniques for recommender systems. **Computer**, IEEE, v. 42, n. 8, p. 30–37, 2009.

KULKARNI, S.; RODD, S. F. Context aware recommendation systems: A review of the state of the art techniques. **Computer Science Review**, Elsevier, v. 37, p. 100255, 2020.

KVERNADZE, G. et al. **Two Is Better Than One: Dual Embeddings for Complementary Product Recommendations**. arXiv, 2022. Disponível em: <http://arxiv.org/abs/2211.14982>.

LATHIA, N.; HAILES, S.; CAPRA, L. Temporal collaborative filtering with adaptive neighbourhoods. In: **Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval**. Boston, MA, USA: Association for Computing Machinery, 2009. (SIGIR '09), p. 796–797.

LI, L. et al. A contextual-bandit approach to personalized news article recommendation. In: **Proceedings of the 19th international conference on World wide web**. ACM, 2010. (WWW '10). Disponível em: <http://dx.doi.org/10.1145/1772690.1772758>.

LI, L. et al. A contextual-bandit approach to personalized news article recommendation. In: **Proceedings of the 19th International Conference on World Wide Web**. New York, NY, USA: Association for Computing Machinery, 2010. (WWW'09), p. 661–670.

LI, L. et al. An unbiased offline evaluation of contextual bandit algorithms with generalized linear models. In: **Proceedings of the 2011 International Conference on On-line Trading of Exploration and Exploitation 2**. New York, NY, USA: JMLR.org, 2011. (OTEAE'11), p. 19–36.

LILLICRAP, T. P. et al. Continuous control with deep reinforcement learning. 9 2015. Disponível em: <http://arxiv.org/abs/1509.02971>.

LINDEN, G.; SMITH, B.; YORK, J. Amazon.com recommendations: Item-to-item collaborative filtering. v. 7, n. 1, p. 76–80, 2003. ISSN 1089-7801. Disponível em: <http://ieeexplore.ieee.org/document/1167344/>.

LINDEN, G.; SMTITH, B.; YORK, J. Amazon.com Recommendations: Item-to-item collaborative filtering. **IEEE Internet Computing**, IEEE, New York, NY, USA, v. 7, n. 1, p. 76–80, 2003.

LIU, F. et al. End-to-End Deep Reinforcement Learning based Recommendation with Supervised Embedding. In: **Proceedings of the 13th International Conference on Web Search and Data Mining**. ACM, 2024. p. 384–392. ISBN 978-1-4503-6822-3. Disponível em: <https://dl.acm.org/doi/10.1145/3336191.3371858>.

LIU, F. et al. Top-aware reinforcement learning based recommendation. v. 417, p. 255–269, 2020. ISSN 09252312. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0925231220311656>.

LIU, F. et al. State representation modeling for deep reinforcement learning based recommendation. v. 205, p. 106170, 2020. ISSN 09507051. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S095070512030407X>.

LIU, H. et al. Automated Embedding Size Search in Deep Recommender Systems. In: **Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval**. ACM, 2020. p. 2307–2316. ISBN 978-1-4503-8016-4. Disponível em: <https://dl.acm.org/doi/10.1145/3397271.3401436>.

LIU, X.; ABERER, K. Soco: a social network aided context-aware recommender system. In: **Proceedings of the 22nd International Conference on World Wide Web**. New York, NY, USA: Association for Computing Machinery, 2013. (WWW '13), p. 781–802. ISBN 9781450320351. Disponível em: <https://doi.org/10.1145/2488388.2488457>.

LO, Y.-Y. et al. Temporal Matrix Factorization for Tracking Concept Drift in Individual User Preferences. v. 5, n. 1, p. 156–168, 2018. ISSN 2329-924X. Disponível em: <http://ieeexplore.ieee.org/document/8125765/>.

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

MA, H. et al. Improving recommender systems by incorporating social contextual information. **ACM Transactions on Information Systems (TOIS)**, ACM New York, NY, USA, v. 29, n. 2, p. 1–23, 2011.

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

MIKOLOV, T. et al. **Efficient Estimation of Word Representations in Vector Space**. 2013. Disponível em: <http://arxiv.org/abs/1301.3781>.

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

MLADENOV, M. et al. RecSim NG: Toward principled uncertainty modeling for recommender ecosystems. **arXiv preprint**, p. 1–23, 2021.

MNIH, V. et al. Playing atari with deep reinforcement learning. **arXiv preprint arXiv:1312.5602**, 2013.

OKU, K. et al. Context-aware svm for context-dependent information recommendation. In: **Proceedings of the 7th International Conference on Mobile Data Management**. New York, NY, USA: IEEE, 2006. (MDM '06), p. 1–4.

PANNIELLO, U.; TUZHILIN, A.; GORGOGLIONE, M. Comparing context-aware recommender systems in terms of accuracy and diversity. **User Modeling and User-Adapted Interaction**, v. 24, p. 35–65, 2 2014. ISSN 09241868.

PARK, S. et al. Temporal linear item-item model for sequential recommendation. In: **Proceedings of the 18th ACM International Conference on Web Search and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2025. (WSDM '25), p. 354–362.

PAVLOVSKI, M. et al. Time-aware user embeddings as a service. In: **Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining**. Virtual Event, CA, USA: Association for Computing Machinery, 2020. p. 3194–3202.

PIRES, P. R.; ALMEIDA, T. A. Interact2vec – an efficient neural network-based model for simultaneously learning users and items embeddings in recommender systems. v. 181, p. 113408, 2025. ISSN 15684946. Disponível em: <http://arxiv.org/abs/2506.22648>.

PIRES, P. R. et al. Exploitation over exploration: Unmasking the bias in linear bandit recommender offline evaluation. In: **Proceedings of the 19th ACM Conference on Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2025. (RecSys'24), p. 1–10.

PIRES, P. R.; PASCON, A. C.; ALMEIDA, T. A. Time-dependent item embeddings for collaborative filtering. In: **Proceedings of the 10th Brazilian Conference on Intelligent Systems**. Virtual Event, Brazil: Springer Nature, 2021. (BRACIS '21).

PUTERMAN, M. L. Chapter 8: Markov decision processes. In: _____. **Handbooks in Operations Research and Management Science**. Amsterdam, Netherlands: Elsevier Science Publishers B. V., 2007. v. 2, p. 331–434. ISBN 0927-0507.

RENDLE, S. Factorization machines with libfm. **ACM Transactions on Intelligent Systems and Technology**, v. 3, 5 2012. ISSN 21576904.

RENDLE, S. et al. BPR: Bayesian Personalized Ranking from Implicit Feedback. 2009.

ROHDE, D. et al. RecoGym: A reinforcement learning environment for the problem of product recommendation in online advertising. **arXiv preprint**, p. 1–5, 2018.

SAITO, Y. et al. Open Bandit Dataset and Pipeline: Towards realistic and reproducible off-policy evaluation. In: **Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks**. Red Hook, NY, USA: Curran Associates Inc., 2021. (NeurIPS'21), p. 1–14.

SEZERER, E.; TEKIR, S. **A Survey On Neural Word Embeddings**. 2021. Disponível em: <http://arxiv.org/abs/2110.01804>.

SHANI, G. et al. An mdp-based recommender system. **Journal of Machine Learning Research**, v. 6, n. 9, 2005.

SHANIGU, G. S.; HECKERMAN, D.; BRAFMAN, R. I. An mdp-based recommender system *. **Journal of Machine Learning Research**, v. 6, p. 1265–1295, 2005. Disponível em: <www.mitos.co.il>.

SHI, J.-C. et al. Virtual-Taobao: Virtualizing real-world online retail environment for reinforcement learning. **arXiv preprint**, p. 1–15, 2018.

SILVA, N. et al. Multi-Armed Bandits in Recommendation Systems: A survey of the state-of-the-art and future directions. v. 197, p. 116669, 2022. ISSN 09574174. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0957417422001543>.

SLIVKINS, A. Introduction to Multi-Armed Bandits. v. 12, n. 1-2, p. 1–286, 2019. ISSN 1935-8237, 1935-8245. Disponível em: <http://www.nowpublishers.com/article/Details/MAL-068>.

SU, X.; KHOSHGOFTAAR, T. M. A Survey of Collaborative Filtering Techniques. v. 2009, p. 1–19, 2009. ISSN 1687-7470, 1687-7489. Disponível em: <https://www.hindawi.com/journals/aai/2009/421425/>.

SUNEHAG, P. et al. Deep reinforcement learning with attention for slate markov decision processes with high-dimensional states and actions. 12 2015. Disponível em: <http://arxiv.org/abs/1512.01124>.

SUTTON, R. S.; BARTO, A. G. **Reinforcement Learning: An Introduction Second edition, in progress**. 1998.

TAKÁCS, G. et al. Matrix factorization and neighbor based algorithms for the Netflix Prize problem. In: **Proceedings of the 2008 ACM Conference on Recommender Systems**. New York, NY, USA: Association for Computing Machinery, 2008. (RecSys '08), p. 267–274.

TAKÁCS, G.; TIKK, D. Alternating least squares for personalized ranking. In: **Proceedings of the Sixth ACM Conference on Recommender Systems**. ACM, 2012. p. 83–90. ISBN 978-1-4503-1270-7. Disponível em: <https://dl.acm.org/doi/10.1145/2365952.2365972>.

TRABELSI, F.; KHTIRA, A.; ASRI, B. E. Hybrid Recommendation Systems: A State of Art:. In: **Proceedings of the 16th International Conference on Evaluation of Novel Approaches to Software Engineering**. Online Streaming: SCITEPRESS - Science and Technology Publications, 2021. p. 281–288. ISBN 978-989-758-508-1.

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

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

VASSøY, B. et al. **Time is of the Essence: a Joint Hierarchical RNN and Point Process Model for Time and Item Predictions**. arXiv, 2018. Disponível em: <http://arxiv.org/abs/1812.01276>.

VINAGRE, J.; JORGE, A. M.; GAMA, J. An overview on the exploitation of time in collaborative filtering. **Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery**, Wiley-Blackwell, v. 5, p. 195–215, 9 2015. ISSN 19424795.

WANG, S. et al. A Survey on Session-based Recommender Systems. v. 54, n. 7, p. 154:1–154:38, 2021. ISSN 0360-0300. Disponível em: <https://dl.acm.org/doi/10.1145/3465401>.

WANG, Y.-X.; AGARWAL, A.; DUDIK, M. Optimal and adaptive off-policy evaluation in contextual bandits. **arXiv preprint**, p. 1–23, 2016.

WU, Q.; IYER, N.; WANG, H. Learning contextual bandits in a non-stationary environment. In: . [s.n.], 2018. p. 495 – 504. Cited by: 49; All Open Access, Bronze Open Access, Green Open Access. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85051466151&doi=10.1145%2f3209978.3210051&partnerID=40&md5=12f68c8025abe6b66b55dbf861af03ab>.

YEGANEGI, R.; HARATIZADEH, S.; EBRAHIMI, M. STAR: A session-based time-aware recommender system. **Neurocomputing**, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 573, n. C, p. 127104:1–127104:13, 2024.

ZARZOUR, H.; AL-SHARIF, Z. A.; JARARWEH, Y. RecDNNing: A recommender system using deep neural network with user and item embeddings. In: **2019 10th International Conference on Information and Communication Systems (ICICS)**. IEEE, 2019. p. 99–103. ISBN 978-1-72810-045-6. Disponível em: <https://ieeexplore.ieee.org/document/8809156/>.

ZENG, C. et al. Online context-aware recommendation with time varying multi-armed bandit. v. 13-17-Augu, p. 2025–2034, 2016.

ZHANG, S. et al. Deep learning based recommender system: A survey and new perspectives. **ACM Computing Surveys**, Association for Computing Machinery, v. 52, 2 2019. ISSN 15577341.

ZHANG, Y. et al. A time-aware self-attention based neural network model for sequential recommendation. **Applied Soft Computing**, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 133, n. 109894, p. 1–13, 2023.

ZHAO, K. et al. KuaiSim: a comprehensive simulator for recommender systems. In: **Proceedings of the 37th International Conference on Neural Information Processing Systems**. Red Hook, NY, USA: Curran Associates Inc., 2023. (NIPS'18), p. 44880–44897.

ZHAO, X. et al. Deep reinforcement learning for search, recommendation, and online advertising: A survey. v. 2019, p. 1–15, 2019. ISSN 1931-1745, 1931-1435. Disponível em: <http://arxiv.org/abs/1812.07127>.

ZHAO, X. et al. Recommendations with negative feedback via pairwise deep reinforcement learning. p. 1040–1048, 2018.

ZHAO, X. et al. Whole-chain recommendations. In: **Proceedings of the 29th ACM International Conference on Information & Knowledge Management**. New York, NY, USA: Association for Computing Machinery, 2020. (CIKM '20), p. 1883–1891.

ZHOU, L. **A Survey on Contextual Multi-armed Bandits**. 2016. Disponível em: <http://arxiv.org/abs/1508.03326>.

ZIMDARS, A.; CHICKERING, D. M.; MEEK, C. Using temporal data for making recommendations. In: **Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence**. Seattle, WA, USA: Morgan Kaufmann Publishers Inc., 2001. (UAI '01), p. 580–588.

ZOU, L. et al. Reinforcement learning to optimize long-term user engagement in recommender systems. In: . New York, NY, USA: Association for Computing Machinery, 2019. (KDD '19), p. 2810–2818. ISBN 9781450362016. Disponível em: <https://doi.org/10.1145/3292500.3330668>.

APÊNDICE A

Melhores parâmetros para recomendação não-incremental

Neste apêndice, são apresentados os melhores hiperparâmetros obtidos a partir do processo de otimização realizado segundo o protocolo experimental detalhado na Seção 5.2.1. O espaço de busca considerado para essa etapa encontra-se definido na Seção 5.1.3.

Tabela 11 – Melhores hiperparâmetros para Amazon-Beauty em recomendação não-incremental.

Hiperparâmetro	**Item2Vec**	**Seq2Vec**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	5	10	10	10
Taxa de Aprendizado (α)	0.025	0.25	0.25	0.25
Número de Épocas	100	20	20	100
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	1	1	0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	365	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	2	—
Taxa de Decaimento	—	—	—	5
Peso mínimo	—	—	—	0.5

Tabela 12 – Melhores hiperparâmetros para Amazon-Books em recomendação não-incremental.

Hiperparâmetro	**Item2Vec**	**Seq2Vec**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.025	0.025	0.025	0.025
Número de Épocas	100	100	100	100
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	730	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	5
Peso mínimo	—	—	—	0.5

Tabela 13 – Melhores hiperparâmetros para Amazon-Games em recomendação não-incremental.

Hiperparâmetro	**Item2Vec**	**Seq2Vec**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	5	10	10	10
Taxa de Aprendizado (α)	0.025	0.025	0.025	0.025
Número de Épocas	50	50	50	50
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	730	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	5
Peso mínimo	—	—	—	0.3

Tabela 14 – Melhores hiperparâmetros para BestBuy em recomendação não-incremental.

Hiperparâmetro	**Item2Vec**	**Seq2Vec**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	5	10	10	10
Taxa de Aprendizado (α)	0.025	0.025	0.025	0.025
Número de Épocas	100	100	50	50
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	365	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1.5	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.3

Tabela 15 – Melhores hiperparâmetros para CiaoDVD em recomendação não-incremental.

Hiperparâmetro	**Item2Vec**	**Seq2Vec**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.025	0.025	0.025	0.025
Número de Épocas	100	100	100	100
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	730	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.3

Tabela 16 – Melhores hiperparâmetros para ML-100k em recomendação não-incremental.

Hiperparâmetro	**Item2Vec**	**Seq2Vec**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	5	10	10	10
Taxa de Aprendizado (α)	0.025	0.025	0.025	0.025
Número de Épocas	20	20	20	20
Limiar de Sub Sampling (t)	10^{-3}	10^{-4}	10^{-4}	10^{-4}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	730	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.3

Tabela 17 – Melhores hiperparâmetros para ML-1M em recomendação não-incremental.

Hiperparâmetro	**Item2Vec**	**Seq2Vec**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.25	0.25	0.25	0.25
Número de Épocas	100	100	50	50
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	730	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	2	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.3

Tabela 18 – Melhores hiperparâmetros para RetailRocket em recomendação não-incremental.

Hiperparâmetro	**Item2Vec**	**Seq2Vec**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.25	0.025	0.025	0.025
Número de Épocas	20	20	20	20
Limiar de Sub Sampling (t)	10^{-4}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-1	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	365	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	5
Peso mínimo	—	—	—	0.3

APÊNDICE B

Melhores parâmetros para recomendação incremental

Neste apêndice, são apresentados os melhores hiperparâmetros obtidos a partir do processo de otimização realizado segundo o protocolo experimental detalhado na Seção 5.2.1. O espaço de busca considerado para essa etapa encontra-se definido na Seção 5.1.3.

Tabela 19 – Melhores hiperparâmetros para Amazon-Beauty.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	5	10	10	10
Taxa de Aprendizado (α)	0.25	0.25	0.25	0.25
Número de Épocas	20	20	20	20
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-1	-1	-1	-1
Dimensão dos Fatores (d)	50	50	50	50
Neg. Samples (k)	7	7	7	7
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	182	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.3

Tabela 20 – Melhores hiperparâmetros para Amazon-Books.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.025	0.25	0.025	0.025
Número de Épocas	50	50	50	50
Limiar de Sub Sampling (t)	10^{-4}	10^{-4}	10^{-3}	10^{-4}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Dimensão dos Fatores (d)	50	50	50	50
Neg. Samples (k)	7	7	7	7
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	730	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	2	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.2

Tabela 21 – Melhores hiperparâmetros para Amazon-Games.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.25	0.025	0.25	0.25
Número de Épocas	20	100	20	100
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	365	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	2	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.3

Tabela 22 – Melhores hiperparâmetros para BestBuy.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	5	10	10	5
Taxa de Aprendizado (α)	0.025	0.025	0.025	0.25
Número de Épocas	20	20	20	100
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	182	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.2

Tabela 23 – Melhores hiperparâmetros para CiaoDVD.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	5	10	10	10
Taxa de Aprendizado (α)	0.25	0.025	0.025	0.25
Número de Épocas	20	20	50	20
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	0.5	1	0.5	0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	182	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.2

Tabela 24 – Melhores hiperparâmetros para kuaisim.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.25	0.25	0.25	0.025
Número de Épocas	100	20	20	20
Limiar de Sub Sampling (t)	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	182	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	2	—
Taxa de Decaimento	—	—	—	5
Peso mínimo	—	—	—	0.5

Tabela 25 – Melhores hiperparâmetros para ML-100k.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.025	0.025	0.25	0.025
Número de Épocas	20	50	20	20
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	182	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	1	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.5

Tabela 26 – Melhores hiperparâmetros para ML-1M.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	10	10	10	10
Taxa de Aprendizado (α)	0.25	0.025	0.025	0.25
Número de Épocas	20	20	100	20
Limiar de Sub Sampling (t)	10^{-3}	10^{-3}	10^{-4}	10^{-3}
Exp. de Neg. Sampling (η)	-0.5	-0.5	-0.5	-0.5
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	182	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	2	—
Taxa de Decaimento	—	—	—	3
Peso mínimo	—	—	—	0.3

Tabela 27 – Melhores hiperparâmetros para RetailRocket.

Hiperparâmetro	**Item2Vec**	**SeqI2V**	**TAI2Vec-Disc**	**TAI2Vec-Cont**
Tamanho da Janela (w)	5	10	10	10
Taxa de Aprendizado (α)	0.25	0.25	0.25	0.25
Número de Épocas	20	20	20	20
Limiar de Sub Sampling (t)	10^{-4}	10^{-3}	10^{-3}	10^{-3}
Exp. de Neg. Sampling (η)	-1	-1	-1	-1
Parâmetros Temporais Específicos				
Limiar de corte (Dias)	—	182	—	—
Filtro de Tempo Mínimo	—	—	300	300
Expoente de Tempo	—	—	2	—
Taxa de Decaimento	—	—	—	5
Peso mínimo	—	—	—	0.5

APÊNDICE C

Resultados adicionais

Este apêndice apresenta resultados complementares dos experimentos realizados com os algoritmos LinUCB e LinGreedy no cenário de recomendação incremental offline. Os gráficos aqui expostos detalham o desempenho desses modelos sob as métricas de Hit Rate e NDCG, oferecendo uma visão da evolução da assertividade ao longo das iterações experimentais, complementando as análises discutidas no Capítulo Seção 5.3.

C.1 Resultados adicionais do LinUCB em recomendação incremental offline

=

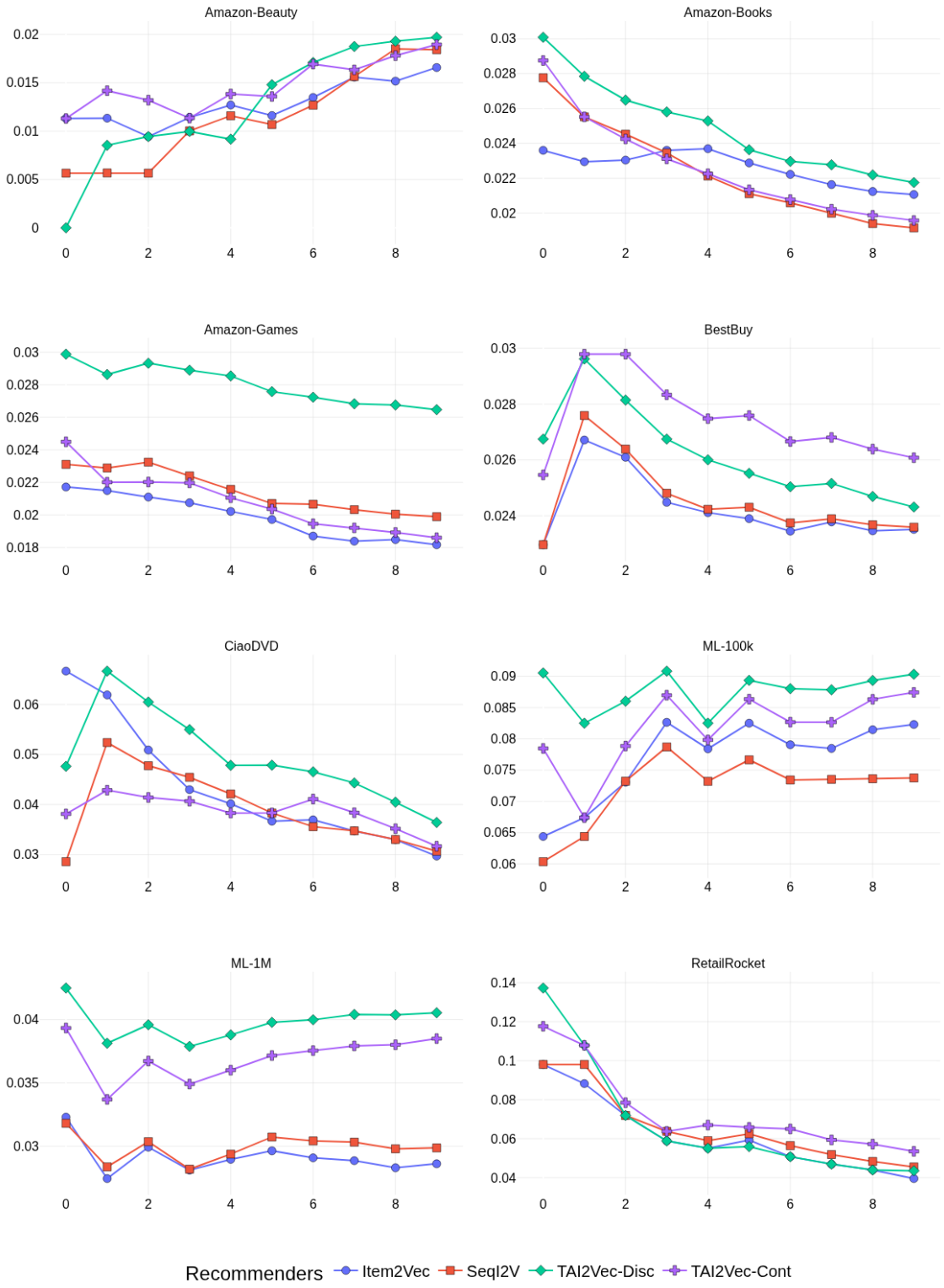


Figura 25 – Resultados do LinUCB sob a métrica Hit Rate

C.2 Resultados adicionais do LinGreedy em recomendação incremental offline

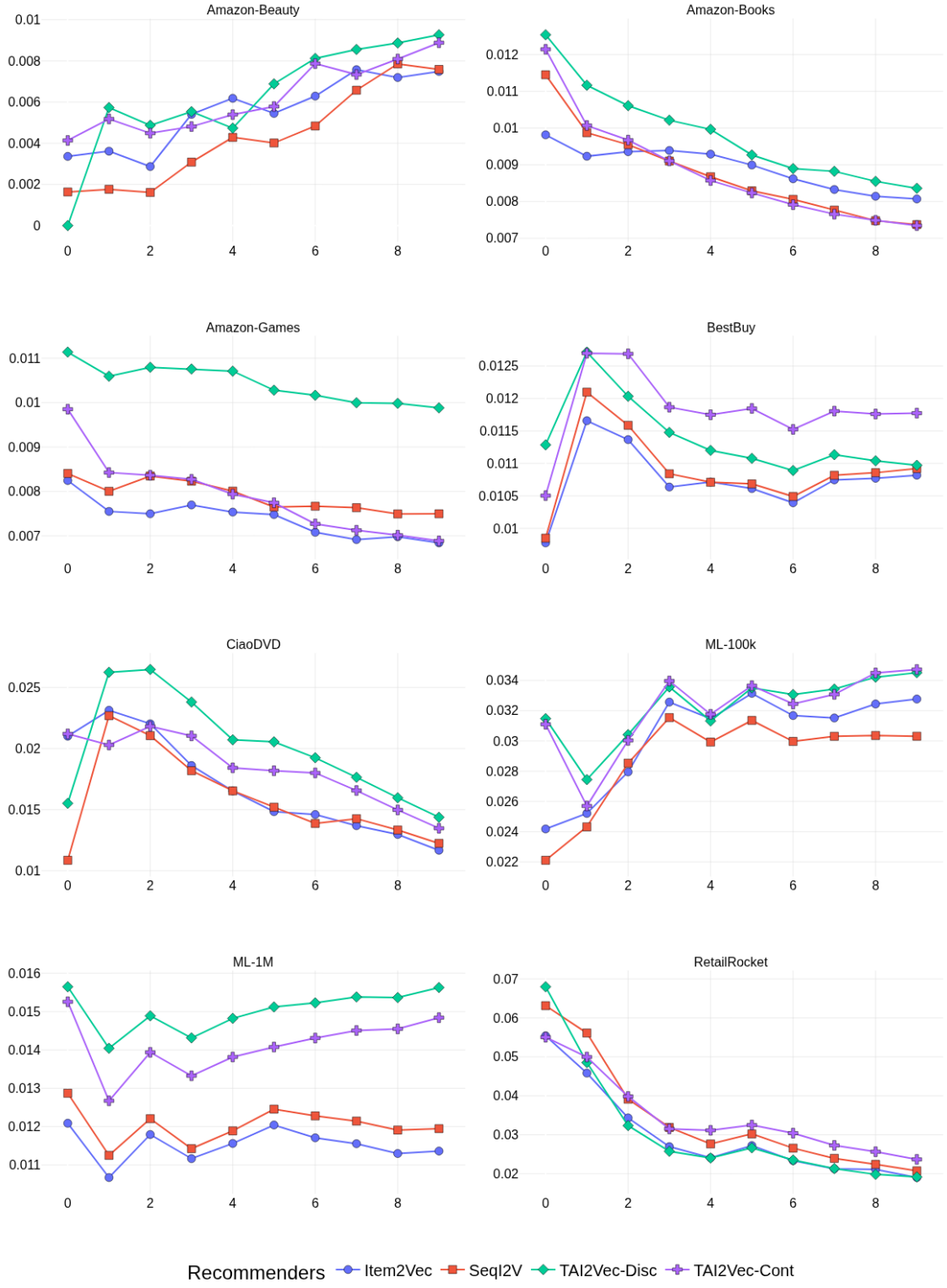


Figura 26 – Resultados do LinGreedy sob a métrica NDCG

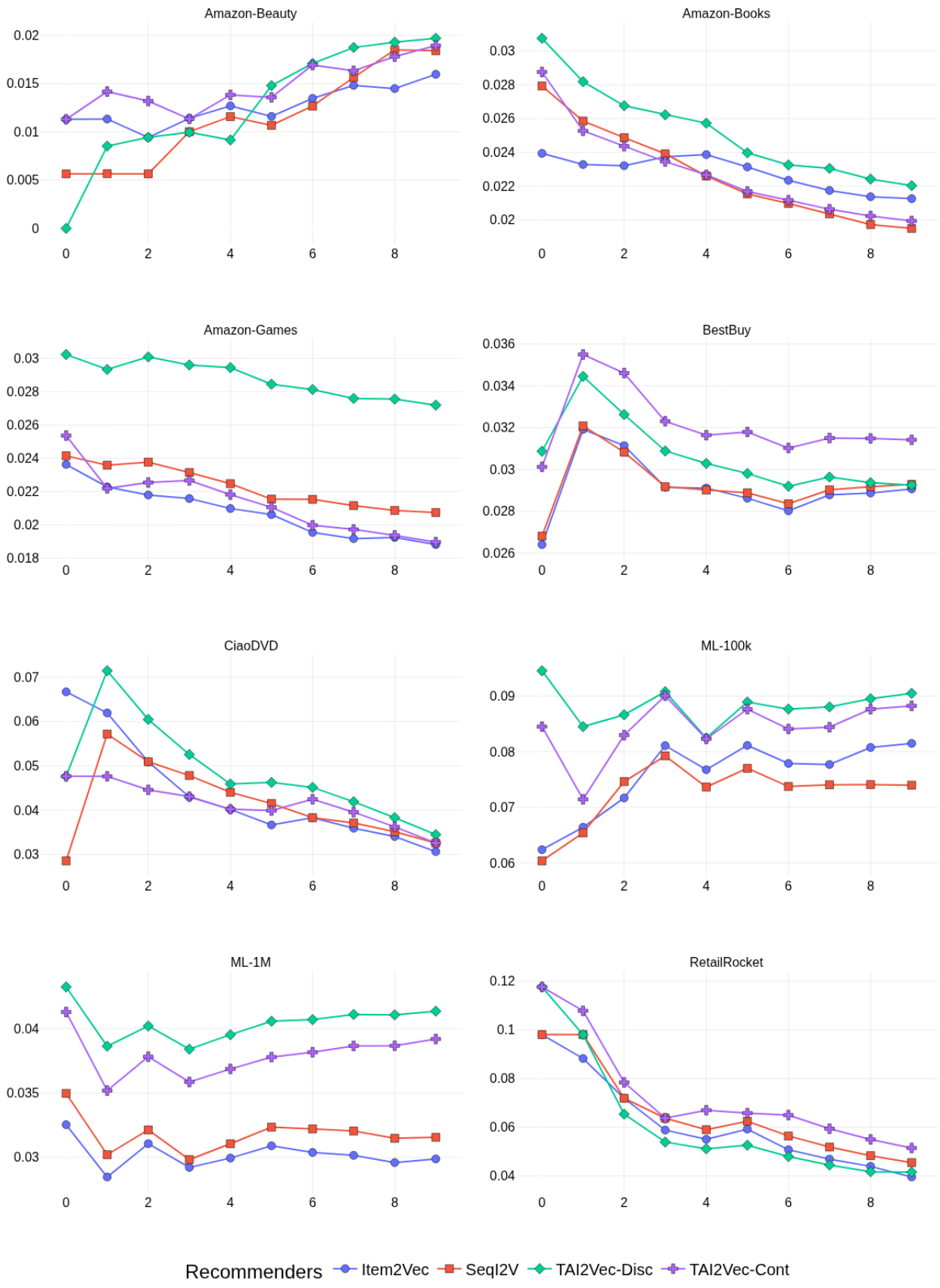


Figura 27 – Resultados do LinGreedy sob a métrica Hit Rate