UNIVERSIDADE
FEDERAL RURAL
DE PERNAMBUCO

Ulisses Chaves Silva

Recomendação Sensível ao Contexto para Comunicação Aumentativa e Alternativa Baseada em Aprendizagem de Máquina

Recife

2024

Ulisses Chaves Silva

Recomendação Sensível ao Contexto para Comunicação Aumentativa e Alternativa Baseada em Aprendizagem de Máquina

Monografia apresentada ao Curso de Bacharelado em Ciências da Computação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Ciências da Computação.

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Computação

Curso de Bacharelado em Ciências da Computação

Orientador: André Câmara Alves do Nascimento

Recife

2024

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

S586r Silva, Ulisses Chaves
Recomendação Sensível ao Contexto para Comunicação Aumentativa e Alternativa Baseada em Aprendizagem de Máquina / Ulisses Chaves Silva. - 2024.
47 f. : il.

Orientador: Andre Camara Alves do Nascimento.
Inclui referências e apêndice(s).

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Bacharelado em Ciência da Computação, Recife, 2024.

1. Recomendação sensível ao contexto. 2. Comunicação aumentativa e alternativa. 3. Aprendizagem de máquina. I. Nascimento, Andre Camara Alves do, orient. II. Título

CDD 004

MINISTÉRIO DA EDUCAÇÃO E DO DESPORTO
UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO (UFRPE)
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

http://www.bcc.ufrpe.br

Ficha de Aprovação do Trabalho de Conclusão de Curso

Trabalho defendido por Ulisses Chaves Silva às 17 horas do dia 23 de fevereiro de 2024, como requisito para conclusão do curso de Bacharelado em Ciência da Computação da Universidade Federal Rural de Pernambuco, intitulado **Recomendação Sensível ao Contexto para Comunicação Aumentativa e Alternativa Baseada em Aprendizagem de Máquina**, orientado por André Câmara Alves do Nascimento e aprovado pela seguinte banca examinadora:

André Câmara Alves do Nascimento
DC/UFRPE

Péricles Barbosa Cunha de Miranda
DC/UFRPE

Expresso minha gratidão especial à minha família pelo apoio oferecido ao longo desta jornada. Não menos importante, dedico este trabalho aos meus professores, cuja orientação e ensinamentos foram fundamentais para alcançar este ponto do conhecimento. Não posso deixar de mencionar meus amigos e colegas, cujas contribuições foram valiosas para o desenvolvimento da minha carreira profissional e acadêmica.

Agradecimentos

Agradeço a Deus por ter me concedido sabedoria e ciência durante a graduação. Agradeço a minha família pela assistência prestada e por estar sempre ao meu lado. Também quero agradecer especialmente aos meus professores pelo apoio dado. Além disso, agradece a todos os amigos e colegas e contribuiram de alguma forma para minha formação como profissional e acadêmico.

“Receba com simplicidade tudo que acontece em sua vida.”
(Rashi)

Resumo

Comumente, observa-se a adoção de novas técnicas baseadas em inteligência artificial e aprendizagem de máquina (AM) em diversos contextos. Com o avanço das redes neurais artificiais, que possibilitam a representação de diversos tipos de dados e a compreensão das complexas relações entre eles, essa tendência foi ainda mais impulsionada. No entanto, a literatura atual mostra-se escassa ao tentar encontrar estudos atualizados que relacionem essas tecnologias a metodologias pedagógicas para resolver os diversos problemas sociais e promover a inclusão. Este trabalho propõe abordagens atuais utilizadas em AM para a recomendação de pictogramas em um sistema de Comunicação Aumentativa e Alternativa (AAC). Diante da complexidade das necessidades de usuários de AAC, neste trabalho dois modelos neurais sensíveis ao contexto são apresentados e comparados. Esses modelos utilizam técnicas de aprendizagem de máquina para considerar o contexto dinâmico do usuário para gerar recomendações, adaptando-se à localização e ao tempo específicos desse usuário que possui alguma deficiência na comunicação. Adicionalmente, são destacados outros trabalhos que foram usados como base para a criação dessa solução para o problema de recomendação de pictogramas existente na aplicação móvel Livox.

Palavras-chave: Recomendação sensível ao contexto, Comunicação aumentativa e alternativa, Aprendizagem de máquina.

Abstract

Commonly, the adoption of new techniques based on artificial intelligence and machine learning (ML) is observed in various contexts. With the advancement of artificial neural networks, which enable the representation of various types of data and the understanding of complex relationships among them, this trend has been further propelled. However, the current literature appears scarce when attempting to find updated studies that relate these technologies to pedagogical methodologies to address various social issues and promote inclusion. This work proposes current approaches used in ML for the recommendation of pictograms in an Augmentative and Alternative Communication (AAC) system. Given the complexity of the needs of AAC users, this paper presents and compares two context-aware neural models. These models use machine learning techniques to consider the dynamic context of the user to generate recommendations, adapting to the specific location and time of this communication-disabled user. Additionally, other works are highlighted that served as a foundation for creating this solution for the existing pictogram recommendation problem in the Livox mobile application.

Keywords: Context-aware recommendation, Augmentative and alternative communication, Machine learning.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

IA	Inteligência Artificial
SR	Sistema de Recomendação
CARS	Context-Aware Recommender System
TF	TensorFlow
TFR	TensorFlow Ranking
TFRS	TensorFlow Recommenders
AAC	Augmentative and Alternative Communication
FC	Filtragem Colaborativa
RNA	Rede Neural Artificial
TF	TensorFlow
ScaNN	Scalable Nearest Neighbors
AM	Aprendizagem de Máquina
LTR	Learn To Ranking

Sumário

1 Introdução

Com a ideia de um mundo globalizado mais acessível para todos, nasceu a necessidade da criação de novas soluções para deficiências visuais, auditivas, motoras e intelectuais (GARG; SHARMA, 2020). Porém, com a individualidade de cada ser, é notória a dificuldade para um usuário com deficiência encontrar soluções tecnológicas para sua necessidade (HEUMADER; MURILLO-MORALES; MIESENBERGER, 2022).

A Comunicação Aumentativa e Alternativa (*Augmentative and Alternative Communication* ou AAC) apresenta-se como uma solução intermediária para lidar com questões de deficiências relacionadas a comunicação, oferecendo o potencial de proporcionar às pessoas com necessidades complexas nessa área, a oportunidade de acessar a magia e o poder da expressão verbal (LIGHT; DRAGER, 2007; HIGGINBOTHAM et al., 2007).

Com o passar do tempo, o conceito de AAC evoluiu e tem buscado atender as necessidades de uma população mais ampla, expandindo seu foco da comunicação interpessoal através de serviços de *internet* e dispositivos móveis, por exemplo. Essa mudança, trazendo novos métodos e abordagens, apresenta novas soluções e desafios para para as pessoas com necessidades de comunicação cada vez mais complexas (SHANE et al., 2012).

Um sistema de recomendação (SR) tem como função fornecer recomendações personalizadas para cada usuário do sistema. Essas sugestões são feitas empregando abordagens como a análise de características dos itens recomendados e do usuário, ou na análise de comportamentos e opiniões de usuários similares para entender as preferências dos usuários (KULKARNI; RODD, 2020).

Juntamente com AAC, um SR teria como objetivo usar conceitos atuais baseados em Inteligência Artificial (IA) para auxiliar nesse tipo de engajamento e facilitar a comunicação de pessoas com deficiências, podendo fazer sugestões apropriadas alinhadas à pesquisa com as preferências do usuário que sanasse a sua necessidade. No entanto, há barreiras ao se adotar um SR para sanar esse problema da comunicação, uma vez que é preciso haver esforço e estudo aprofundado das melhores práticas para se ter, no final, um resultado satisfatório (KULKARNI; RODD, 2020; SASSI; MELLOULI; YAHIA, 2017).

Existem muitos métodos aplicáveis para a construção de SR e escolher o que mais se adeque a casos de uso específicos pode ser uma tarefa árdua. No entanto, é fato que, atualmente, com o uso de smartphones, é possível saber e lidar com o contexto em tempo real no qual o usuário está inserido, tendo como recursos, o lugar, hora

e dia da semana (KULKARNI; RODD, 2020; SASSI; MELLOULI; YAHIA, 2017; ADOMAVICIUS; TUZHILIN, 2010). Logo, tendo o contexto acessível, é possível a existência de um sistema de recomendação sensível ao contexto (*context-aware recommender system* ou CARS), que podem superar os métodos de recomendações tradicionais, uma vez que esses não levam em consideração o ambiente e a situação do usuário (COSTA; DOLOG, 2019).

As redes neurais profundas têm ganhado bastante destaque na resolução de problemas reais e não seria diferente para SRs, pois as grandes quantidades de dados e as expectativas dos usuários, culminando em uma experiência personalizada, são cada vez maiores (MU, 2018; ZHANG et al., 2019). Por isso, a empresa Google tem sido uma grande entusiasta em soluções com a plataforma TensorFlow (TF), contendo bibliotecas escaláveis e eficientes, específicas para recomendação, como o TensorFlow Recommenders (TFRS) (ANANDARAJ et al., 2021).

Alinhado nas soluções tecnológicas, o Livox é um aplicativo móvel AAC com aplicação de IA para diversas funcionalidades, no intuito de dar voz para as pessoas com deficiência (LIVOX, 2023). O Livox adapta-se às necessidades de pessoas com dificuldades motoras e de comunicação verbal com o usando pictogramas (NEAMTU et al., 2019), que são desenhos figurativos estilizados usados para transmitir informações analógicas ou figurativas de forma direta, seja para indicar um objeto ou expressar uma ideia (TIJUS et al., 2007).

O trabalho de Hernandez et al. (HERNáNDEZ et al., 2014) trata de alguns tópicos levantados nesta seção. O trabalho consiste na construção de um SR para tomadas decisões para um sistema AAC. Entretanto, vale destacar que o uso de RNAs não foi avaliado no contexto de AAC, sendo essa avaliação importante para o avanço do estado da arte.

Desse modo, como objetivo geral, este trabalho vai apresentar a construção de duas propostas de modelos neurais de recomendação sensível ao contexto para AAC. Como objetivos específicos, este trabalho tem como:

- Unir RNA e AAC para trazer duas soluções para o problema de recomendação;
- Avaliar os modelos usando métricas adequadas para recomendação e determinar qual dos modelos apresenta a melhor eficiência.

1.1 Organização do Trabalho

Além deste capítulo introdutório, este trabalho está dividido em 4 capítulos:

- O capítulo 2 descreve referencial teórico e trabalhos antecedentes da pesquisa;

- O capítulo 3 apresenta a metodologia de pesquisa;
- O capítulo 4 apresenta o experimentos e os resultados da pesquisa;
- O capítulo 5 apresenta as conclusões e discute os rumos futuros.

2 Fundamentação Teórica

2.1 AAC

O trabalho de Higginbotham et al. (HIGGINBOTHAM et al., 2007), traz um conceito sobre o que seria AAC: são tecnologias que têm o potencial de oferecer a pessoas com necessidades complexas de comunicação acesso à magia e ao poder da comunicação. Esse é um assunto que se prolonga desde a década de 80, tendo pesquisadores e profissionais explorado tecnologias e técnicas para melhorar o acesso a dispositivos de comunicação para pessoas com necessidades especiais, podendo incluir o uso de dispositivos eletrônicos, símbolos gráficos (pictogramas), ou outras formas de comunicação para apoiar pessoas com dificuldades na fala ou comunicação (LIGHT; DRAGER, 2007).

Como uma ferramenta online para confecção de pranchas de comunicação, o Portal ARASAAC (ARASAAC, 2024) é um bom exemplo de uma tecnologia AAC. Essas pranchas de comunicação são compostas por símbolos e o Portal ARASAAC tem como proposta o desenvolvimento e a disponibilização desses símbolos através de pranchas para apoiar o desenvolvimento e utilização da AAC por meio de uma banco de imagens disponibilizados de forma gratuita.

Outro exemplo relevante é o Livox (LIVOX, 2023), um aplicativo *mobile* que utiliza pictogramas para facilitar a comunicação de pessoas com deficiências na fala. Por meio do aplicativo, o usuário pode selecionar pictogramas que representam suas necessidades ou ações. Em seguida, o aplicativo emite o nome do pictograma ou o que ele representa no idioma do usuário. A interface com os pictogramas no Livox é ilustrada na Figura 1.

Porém, por mais que existam muitas soluções no contexto de AAC relacionadas a inclusão, o que Garg e Sharma (GARG; SHARMA, 2020) sugerem é que sempre haja o desenvolvimento de pedagogias e de novas tecnologias assistivas que promovam a criatividade e ajudem a criar um ambiente seguro para as pessoas que possuam alguma deficiência, onde possam aprender e se comunicar. Por esses motivos, esse tópico ainda se mostra relevante para ser estudado.

2.2 Sistemas de Recomendação Sensível ao Contexto

Um SR tem a funcionalidade de fornecer aos usuários recomendações personalizadas e relevantes, com base nas suas preferências e histórico de ações. Eles

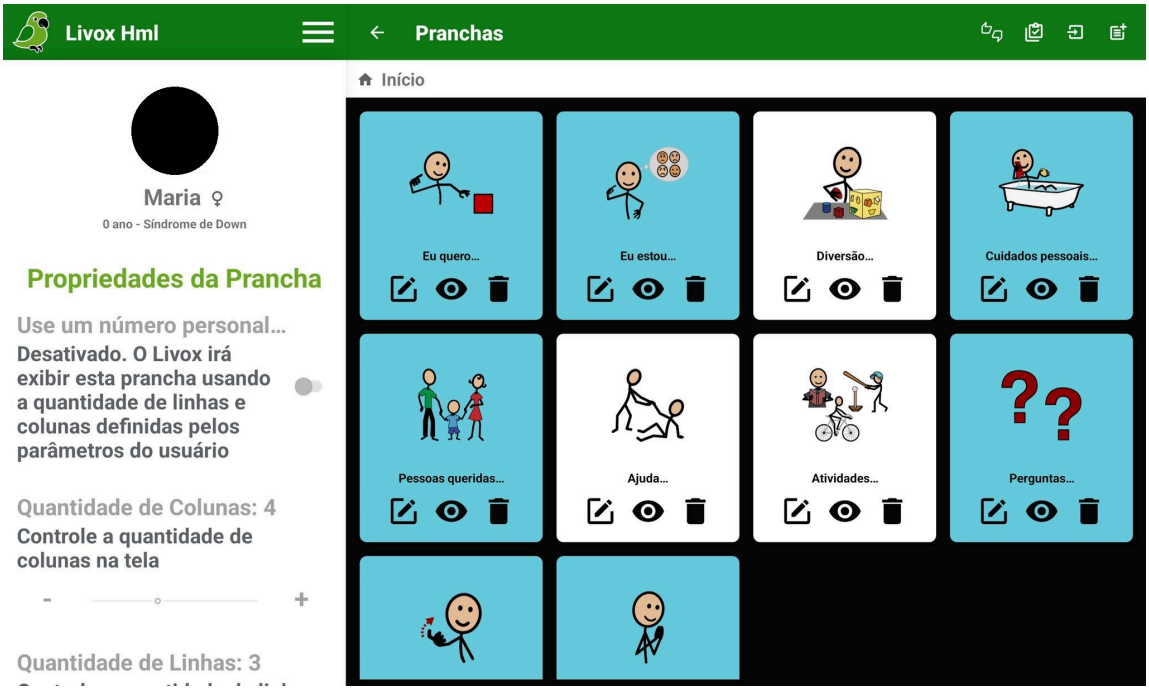

Figura 1 – O uso de pictogramas no aplicativo Livox

podem ajudar a melhorar a experiência e fidelidade do cliente, assim como aumentar as vendas. Além disso, os SRs podem ajudar a reduzir o tempo que os clientes gastam procurando produtos e serviços, tornando o processo de compra mais eficiente (SASSI; MELLOULI; YAHIA, 2017). Aplicações comuns que utilizam SR incluem *e-commerces*, que recomendam itens para estimular novas compras; aplicativos de turismo, que sugerem pontos próximos alinhados à localização e preferências do usuário; e plataformas de *streaming*, que apresentam filmes e séries não assistidos pelo usuário.

Existem abordagens mais tradicionais referentes a SR como a baseada em conteúdo, que recomenda com base nas características dos itens, também a de filtragem colaborativa (FC), que recomenda com base nas preferências de usuários com gostos semelhantes e por fim, a abordagem híbrida que combina a abordagem de conteúdo e FC (KULKARNI; RODD, 2020).

Com o passar das pesquisas e estudos acerca desses sistemas no percorrer dos anos, sempre com fins de melhorias, passou-se a levar em consideração a mudança nas preferências do usuário ao longo do tempo, devido a fatores contextuais (KULKARNI; RODD, 2020), principalmente devido a pesquisas de comportamento do consumidor (ADOMAVICIUS; TUZHILIN, 2010). Isso levou ao desenvolvimento dos sistemas com sensibilidade ao contexto para incorporar a influência do contexto nas preferências do usuário (KULKARNI; RODD, 2020).

As variáveis contextuais usadas como entradas para um modelo, podem variar dependendo da recomendação que precise ser feita. Dia da semana, hora do dia, coordenadas geográficas, dispositivo usado, idioma e emoções/humor do usuário são

algumas das possíveis variáveis usadas em CARS (KULKARNI; RODD, 2020).

Um problema comum em SR acontece quando um novo usuário se cadastra em uma aplicação, o sistema não possui nenhum histórico prévio desse usuário e, como resultado, o SR não é capaz de fornecer recomendações personalizadas para ele. Esse desafio comumente encontrado em SRs é conhecido como *cold start*. No entanto, a utilização do contexto pode atenuar esse problema. Ao fornecer seu contexto, qualquer usuário, independentemente de suas preferências ou tempo de uso, pode contribuir com informações relevantes para o sistema, permitindo assim a geração de recomendações mais adequadas e personalizadas (KULKARNI; RODD, 2020).

2.2.1 Sistemas de Recomedações Baseados em Redes Neurais

Com o objetivo de tentar emular o processamento de informações do cérebro humano, a rede neural artificial (RNA) tem a proposta de armazenar informações de memória por meio do processamento de informações tendo a forma de um modelo composto por neurônios interconectados. Cada neurônio representa uma função de saída específica, chamada função de ativação. As conexões entre os nós representam pesos para o sinal que passa por elas, que são equivalentes à memória da RNA (WU; FENG, 2018).

Com o avanço recentes das RNAs, os CARS também foram impactados de maneira positiva. A necessidade de aprimorar os resultados e representar de maneira mais eficaz as complexas relações de dados, ocasionou na exploração de novos tipos de entradas nos conjuntos de dados. Estes dados, frequentemente não lineares, apresentavam desafios para sistemas mais tradicionais na extração de informações contextuais. Esse cenário favoreceu o uso das incorporações (*embeddings*), que criam representações vetoriais de dados complexos, contribuindo, na maioria dos casos, para resultados superiores em comparação com SRs convencionais (MU, 2018; ZHANG et al., 2019).

Projetar a arquitetura do modelo é muito importante para definir a sua eficiência. A arquitetura de Duas Torres se mostra eficaz para um modelo neural de recuperação de informação (LI et al., 2022; TensorFlow Recommenders Authors, 2023a). Nela existem duas ”torres”, a primeira sendo a torre de consulta e a segunda de torre de candidato (respectivamente, como ”Query Feature”e ”Item Feature”na Figura 2). Enquanto a torre de consulta tem como entrada as características do modelo, a de candidato lida com os itens que serão recomendados. Cada torre é responsável por aprender representações específicas para diferentes tipos de entradas (LI et al., 2022).

Após as representações serem aprendidas nessas duas torres, normalmente é realizada uma operação de similaridade para medir quão bem a consulta se relaciona

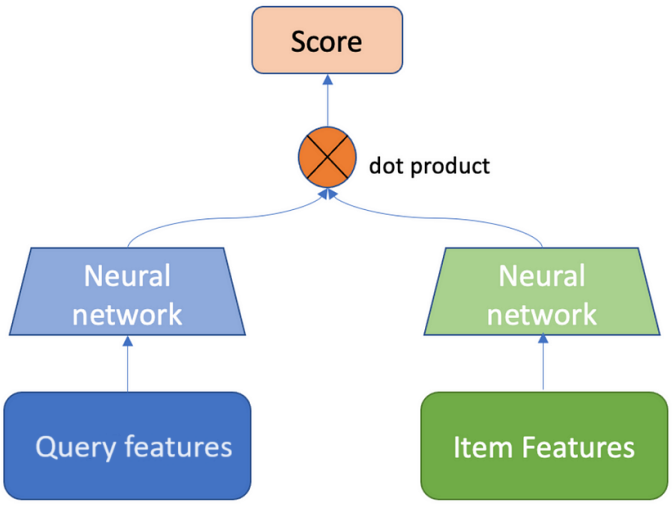

Figura 2 – Exemplo da arquitetura de Duas Torres

com os candidatos. Isso é frequentemente usado em SRs, nos quais o objetivo é encontrar itens relevantes para um usuário com base em sua consulta (LI et al., 2022). No contexto de recomendação a tarefa usada é a de recuperação de informações. Um exemplo dessa estrutura pode ser vista na Figura 2.

Outra estrutura de rede neural comumente usada é a de *feedforward*, onde a informação percorre em uma única direção, da entrada para a saída. Nessa arquitetura cada camada de neurônios está conectada à próxima, e os dados fluem diretamente da camada de entrada, que recebe os dados de entrada da rede, passando pelas camadas ocultas, que realizam transformações nos dados e, por último, chegando à camada de saída, produzindo os resultados da rede (D'AMICO et al., 2019).

Feedforward é utilizada em várias tarefas, como classificação, regressão e aproximação de funções. Um exemplo dessa arquitetura pode ser vista na Figura 3.

Para compor uma arquitetura, uma camada (*layer*) é uma unidade essencial que processa dados e que pode conter um conjunto de neurônios. Existem vários tipos de camadas, cada uma possui funções específicas na RNA. A camada densa (*dense layer*) por exemplo, tem um papel fundamental, onde cada neurônio conecta a todos os neurônios da camada anterior. Em termos simples, todos os dados gerados nas camadas anteriores são empregados como entrada para cada neurônio na camada densa. Essa conectividade completa permite que a camada densa aprenda padrões complexos (D'AMICO et al., 2019).

Como já citado, as incorporações são representações vetoriais de um dado

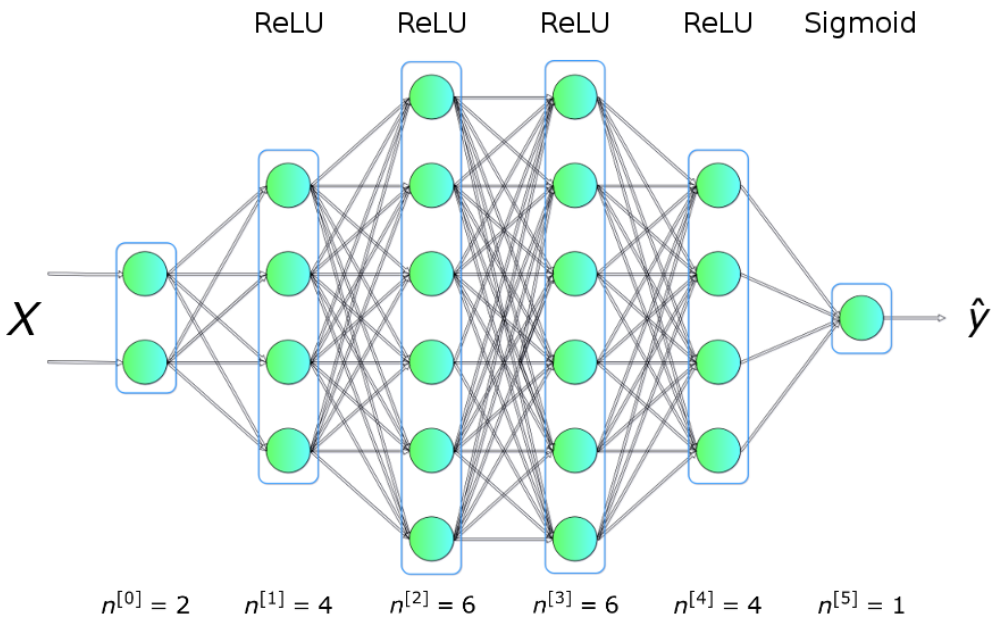

Figura 3 – Exemplo da arquitetura *feedforward*
(SKALSKI, 2024)

em um espaço contínuo. Logo, a camada de incorporação (*embedding layer*) tem a responsabilidade de mapear esses dados categóricos nesse espaço. Essa camada é crucial para inserir as informações nos modelos para que ele possa aprender padrões semânticos e relacionamentos entre essas categorias (ZHANG et al., 2019).

Função de ativação introduz não-linearidade na saída do neurônio, permitindo que a rede aprenda padrões mais complexos nos dados. Uma rede neural sem funções de ativação, seria equivalente a um modelo linear, independentemente do número de camadas, perdendo a capacidade de representar relações complexas nos dados (WU; FENG, 2018).

2.2.1.1 Recuperação de Informação

Algoritmos de recuperação de informações se concentram em organizar, armazenar e acessar grandes volumes de informações de forma eficiente. Segundo Roshi e Roohparvar (ROSHDI; ROOHPARVAR, 2015) , pode-se entender que a recuperação de informações é essencialmente composta por três processos principais:

- Indexação: Neste processo, os documentos são analisados e representados em uma forma resumida, geralmente por meio da extração de palavras-chave ou conceitos importantes. Isso permite uma rápida busca e recuperação posterior;
- Filtragem: Durante este processo, palavras irrelevantes, como palavras de parada (stop words) e termos comuns, são removidas para melhorar a precisão da

busca e reduzir o ruído nos resultados;

- Busca e Recuperação: Esta é a parte central do sistema, onde os usuários inserem consultas e o sistema busca documentos relevantes com base nessas consultas. Existem várias técnicas de busca disponíveis, que podem variar desde a simples correspondência de palavras-chave até algoritmos mais avançados de correspondência de texto.

Portanto, a recuperação de informações é essencial para organizar e acessar grandes volumes de informações de forma rápida e eficiente, facilitando a descoberta de conhecimento e a satisfação das necessidades de informação dos usuários. (ROSHDI; ROOHPARVAR, 2015).

2.2.1.2 Learning to Rank (LTR)

Um algoritmo LTR é um tipo de algoritmo de aprendizado de máquina que é projetado para classificar uma lista de itens em uma ordem específica com base em características relevantes. Em vez de classificar apenas um único item, como em problemas de classificação tradicionais, um algoritmo LTR é treinado para classificar vários itens em uma lista de acordo com sua relevância para uma determinada consulta ou contexto (ZEHLIKE; YANG; STOYANOVICH, 2022).

No contexto de sistemas de recomendação, por exemplo, um algoritmo LTR pode ser treinado para classificar uma lista de produtos ou itens de conteúdo de acordo com a probabilidade de um usuário interagir ou se envolver com cada item. Isso é especialmente útil em situações em que há uma grande quantidade de itens disponíveis e é necessário ordená-los de acordo com sua relevância para um usuário específico (ZEHLIKE; YANG; STOYANOVICH, 2022).

2.3 Trabalhos Relacionados

O trabalho apresentado por Hernandez (HERNáNDEZ et al., 2014) mostra como um algoritmo de AM, com finalidades pedagógicas, pode ser proveitoso para resolução de problemas de recomendação. Nele é descrito a criação de um modelo AAC para tomada de decisões, que, segundo os autores, funciona como um SR. No final, bons resultados são encontrados, tanto pelas métricas na eficiência das recomendações, como no desempenho observado em um estudo de caso, sendo validado por uma pessoa com afásia e por indivíduos considerados saudáveis.

O Livox, citado anteriormente, também adota soluções baseadas em AM. Essas soluções são explanadas no trabalho de Neamtu et al. (NEAMTU et al., 2019), onde

os autores enfatizam que Livox é uma aplicação móvel de AAC, baseada em pictogramas e que incorpora IA para se adaptar às necessidades de pessoas com desafios na comunicação. O artigo apresenta um CARS utilizando um algoritmo de classificação para pictogramas, além de uma funcionalidade de conversação que faz uso de processamento de linguagem natural. Essa funcionalidade visa permitir que usuários com deficiência respondam a perguntas de maneira mais rápida.

A sensibilidade de contexto já é usada em algoritmos de RNA como é mostrado através do ”Collective embedding for Neural Context-aware Recommender Systems” (CoNCARS), como é chamado pelos autores desse estudo, que usa técnicas convolucionais para aprender correlações não lineares entre diferentes dimensões de contexto (COSTA; DOLOG, 2019).

Mesmo para padrões lineares, uma vez que SRs tradicionais tendem a capturar os padrões sequenciais de curto prazo (comportamentos próximos no tempo), os modelos neurais recorrentes são mais eficazes em capturar padrões sequenciais mais longos, ou seja, a história global do comportamento, o que leva a recomendações mais eficazes, como proposto no projeto de Rakkappan et al. (RAKKAPPAN; RAJAN, 2019).

O trabalho de Anandaraj et al. (ANANDARAJ et al., 2021), elucida de forma clara a construção de um SR usando a biblioteca TFRS além de explanar sobre ela. Também no trabalho de Pasumarthi (PASUMARTHI et al., 2019) e no projeto de Damico et al. (D’AMICO et al., 2019) são mostradas algumas das muitas funcionalidades que a biblioteca TFR proporciona como as funções de perdas e métricas já implementadas.

As referências mencionadas neste capítulo serviram como base conceitual, auxiliando no desenvolvimento deste trabalho. Porém, o uso de RNA para a construção e avaliação de SRs em uma aplicação AAC não foi abordado na literatura, logo o objetivo deste trabalho é apresentar e avaliar dois modelos de CARS, utilizando abordagens de redes neurais para a recomendação de pictogramas (representados por frases) em uma aplicação AAC.

3 Metodologia

Seguindo a orientação de Gil (GIL, 2002), avançar os conhecimentos científicos sem uma preocupação direta com suas aplicações práticas é considerado pesquisa pura. Logo, o presente trabalho é uma pesquisa de finalidade básica pura, com objetivos descritivos e executada por meio de levantamento bibliográfico, testes e avaliações.

Inicialmente, buscou-se a base teórica acerca de CARS e AAC, tendo como referência trabalhos acadêmicos relevantes. Foram feitas algumas buscas para identificar um conjunto de artigos relevantes que falassem sobre o tema e que deveriam ser correspondidos pela busca final. Começamos examinando os artigos mais bem classificados por palavras-chave que seriam incluídas na *string* de pesquisa final. As buscas iniciais foram feitas usando palavras-chave gerais, incluindo, como, por exemplo:

- ”recomendação sensível ao contexto”;
- ”comunicação aumentativa e alternativa”.

Então foi usada a seguinte *string* de busca booleana para garantir a captura de uma grande variedade de artigos relacionada aos dois temas:

> *(((“recommender system” OR “recommendation”) AND (“context-sensitive” OR “context-aware”)) OR (“augmentative and alternative communication” OR aac))*.

Foi usada essa *string* para pesquisar os metadados relacionados a periódicos e anuais de conferências nas bases de dados bibliográficas:

- IEEEXplore;
- ACM Digital Library.

A busca resultou em 2.441 referências de artigos publicados entre 1999 e 2023, sendo 1.756 provenientes do IEEEXplore e 685 da ACM Digital Library. Após a aplicação dos critérios definidos abaixo, foram selecionados os artigos que abordam os temas deste trabalho.

Foram incluídos: (i) artigos completos, revisados por pares e publicados; (ii) o estudo está disponível nos serviços da biblioteca universitária e acessíveis aos autores durante o período da pesquisa; e (iii) artigos diretamente relacionados ao tema deste trabalho.

Foram excluídos: (i) artigos curtos (menor ou igual a 4 páginas); e (ii) textos não publicados em inglês.

Antes de aceitar um artigo no conjunto final para revisão, foi verificado para garantir que não houvesse duplicação. Por exemplo, no mesmo artigo listado em mais de uma base de dados, apenas um estudo seria incluído na revisão.

Com esses critérios, foram identificados 41 artigos duplicados e após excluir os resultados duplicados do conjunto de dados, foram identificados os artigos para a leitura dos títulos e resumos. Desses trabalhos, 27 foram encaminhados para a leitura da introdução e conclusão, em que 8 foram eliminados e 19 foram encaminhados para a fase de síntese e extração de dados.

3.1 Conjunto de Dados

Como mencionado por Géron (GéRON, 2019), a maior parte do tempo em trabalhos de AM é direcionada à preparação dos dados. A integridade e qualidade dos dados são cruciais para a obtenção de bons resultados em treinamento de modelos de AM. Assim, esta etapa da metodologia foi executada de forma minuciosa e precisa, uma vez que a base de dados se mostra como elemento fundamental de AM (MICHALSKI; CARBONELL; MITCHELL, 2013).

3.1.1 Preparação e Divisão dos Dados

Os dados foram extraídos da base de dados do Livox, uma vez que o aplicação possui um amplo conjunto de dados de usuários com deficiências na comunicação e transtornos de aprendizagem, que abrangem diversas características, incluindo informações contextuais. Esses dados contextuais são registrados quando o usuário interage com os pictogramas durante o uso do aplicativo e são enviados e armazenados em um banco de dados projetado para análise escalonável. É importante ressaltar que, para este estudo, os dados foram anonimizados para preservar a privacidade e confidencialidade das informações sensíveis dos usuários.

O conjunto de dados passou por uma série de condições que foram definidas como uma etapa de pré-processamento para uma seleção real dos dados, com o intuito de criar uma diversidade e variação no *dataset* e também para a apresentação de resultados melhores e reais nas avaliações (MICHALSKI; CARBONELL; MITCHELL, 2013).

Essas heurísticas levaram em conta alguns dados que não estiveram presentes no conjunto final de dados, mas foram indispensáveis para selecionar e processar quais as informações que podem ou não serem encaixadas em um contexto real. Logo,

no conjunto final de dados foram selecionados dados de localização (criados a partir de uma clusterização de valores referentes a longitude e latitude do usuário), dia da semana, hora do dia e frase (representação do pictograma). O seguimento de dados que não estão incluídos no conjunto final, mas que foram indispensáveis para a determinação das condições, foram o identificador do usuário no sistema, latitude completa e a longitude completa.

Essas condições foram definidas apartir de uma série de tentativas exaustivas e repetitivas para chegar em números que foram considerados, de forma empírica, satisfatórios para um modelo global de recomendação sensível ao contexto. Logo, em teoria, são formas de excluir dados excessivos e repetidos, ou que fogem do padrão da realidade de um contexto que o usuário esteja realmente inserido:

- Pelo menos dez cliques em locais diferentes para o usuário não ser retirado do *dataset*: Foram usados os dados de latitude e longitudes completos, de forma que foram retirados os usuários irrelevantes para um sistema de recomendação global. Em uma generalização e visando um caso real, se um usuário não estiver em pelo menos dez locais diferentes, provavelmente esse usuário não usou a aplicação por mais de uma vez ou não usou o suficiente para atingir a condição;
- Pelo menos vinte cliques em horários diferentes para o usuário não ser retirado do *dataset*: Foi usado os dados de hora do dia e dia da semana, de forma que sejam retirados os usuários irrelevantes para um sistema de recomendação global. Semelhante ao item anterior, presume-se que o usuário usou a aplicação poucas vezes, logo essa condição não foi atinginda;
- Pelo menos cinquenta cliques para o usuário não ser removido: Remover usuários considerados sem expressão para um sistema de recomendação global;
- Quinze ou mais linhas exatamente iguais para essas linhas serem removidas do *dataset*: Foram consideradas todas as informações do conjunto de dados iniciais, de forma que haja mais dados mais diversificados para o modelo;
- Pelo menos trinta ocorrências da frase no conjunto de dados, para essa frase não ser removida: Como a frase servirá como resultado final do modelo, essa condição serviu para remover os itens de recomendação irrelevantes;
- Foi usado *undersampling*, uma técnica de remoção de amostras para diminuir as classes majoritárias. Isso aconteceu pois durante a fase exploratória de dados, existiam classes que se repetiam massivamente e atrapalhariam o ajuste e treinamento do modelo (MOHAMMED; RAWASHDEH; ABDULLAH, 2020). A remoção consistiu em duas condições que deveriam ser atingidas mutualmente.

A primeira era que se a classe tivesse mais de cem repetições e a segunda se essas repetições contivessem as mesmas entradas do conjunto de dados finais. Embora não resolva o desequilíbrio das classes, essa etapa foi necessária para não haver tanta perda de informação.

Após o pré-processamento, o *dataset* final obtido teve 52818 linhas de dados e 2188 candidatos/classes, além de conter três colunas referentes ao contexto e uma para o item que será recomendado:

- Localização: Números inteiros representando localizações baseadas em suas longitudes e latitudes de cada um dos cliques. As latitudes e longitudes foram clusterizadas com OPTICS, um algoritmo que cria uma ordenação de um conjunto de dados com base em sua estrutura de cluster, utilizando informações de densidade (ANKERST et al., 1999);
- Dia da semana: Um número inteiro (de 0 a 6) representando o dia da semana que o clique foi realizado;
- Hora do dia: Um número inteiro (de 0 a 23) representando a hora do dia em que o clique foi realizado pelo usuário;
- Frase: Uma string contendo uma frase, servindo como intenção ou ação do usuário no contexto em que ele está inserido. No aplicativo Livox, ela representa um pictograma.

Esses dados inteiros foram normalizados de forma que os valores ficassem em um pequeno intervalo em torno de 0. Essa etapa de normalização foi feita através do processo de pré-processamento de características de entrada, presentes na biblioteca TFRS (TensorFlow Recommenders Authors, 2024b).

O *dataset* final considerou 321 usuários relevantes. A distribuição dos cliques ao longo do dia e dos cliques ao longo da semana pode ser visto nas Figuras 4 e 5, respectivamente. Na Figura 6 mostra o número de frases por usuário, onde no eixo 'x', cada barra representa um usuário e no eixo 'y', a quantidade de frases de cada um deles.

O uso moderado de *undersampling* para retirada de classes majoritárias (frases) foi necessário, pois, como visto na Figura 7, várias frases são compartilhadas e usadas por mais de um usuário, então a retirada dessas classes, que contribuem e são importantes para a criação de um contexto, poderia ocasionar na perda de informação. Então, no final, foi visto que cerca de 65,5% das frases estavam sendo compartilhadas por mais de um usuário.

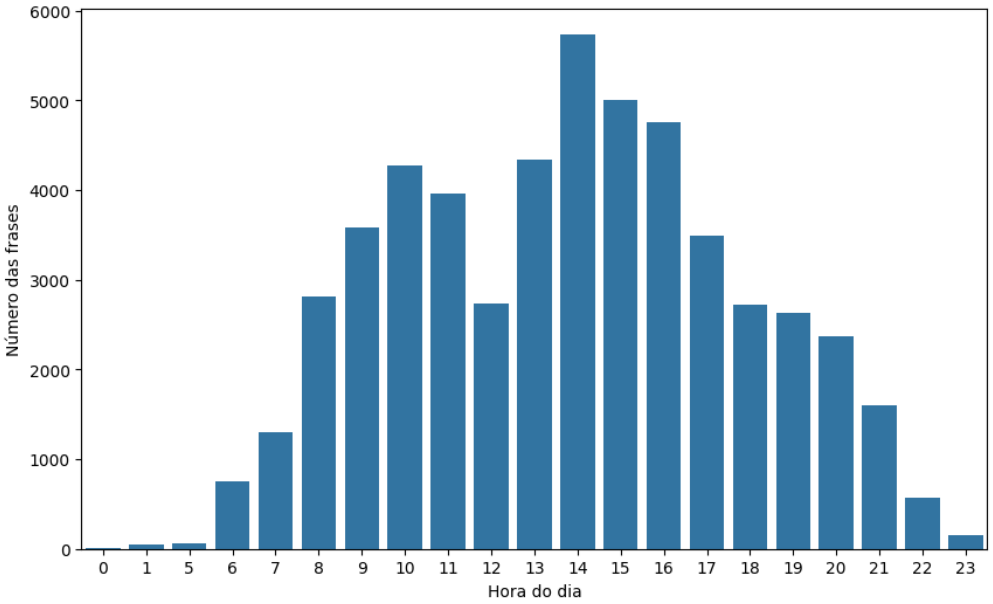

Figura 4 – Distribuição dos cliques ao longo do dia

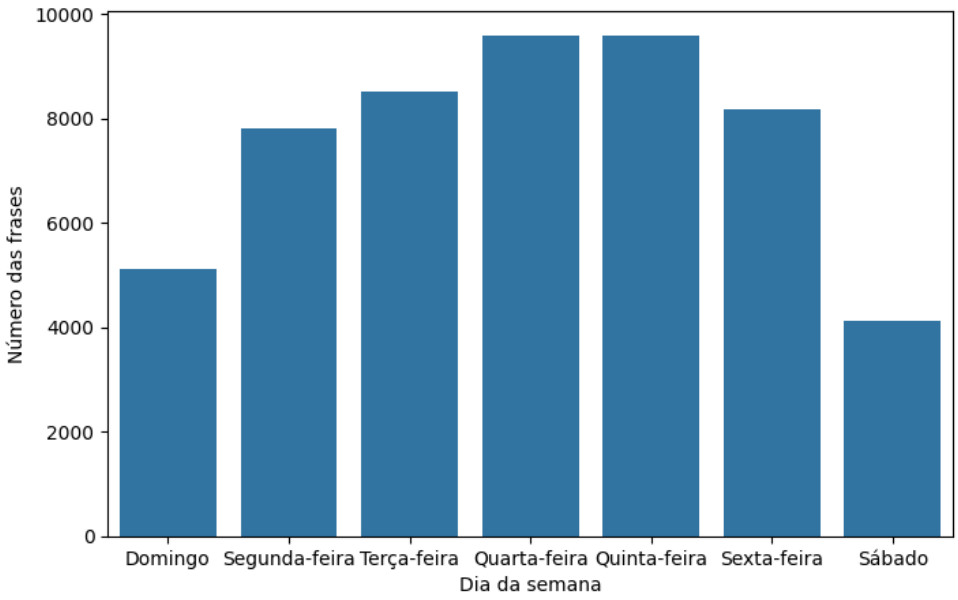

Figura 5 – Distribuição dos cliques ao longo da semana

3.2 Modelos

Os modelos foram construídos e avaliados utilizando a plataforma TF, especialmente com uso da biblioteca TFRS. O TFRS é dedicado a todo o fluxo de trabalho de um SR, desde a preparação dos dados até a criação e implantação do modelo. Tem como propósitos proporcionar uma curva de aprendizado mais fácil e a possibilidade de escalar para uma modelagem neural mais complexa (ANANDARAJ et al., 2021; TensorFlow Recommenders Authors, 2023c).

Além de capacitar a construção e avaliação de modelos de diversas abordagens de recomendação de forma flexível, o TFRS permite a incorporação sem restrições de

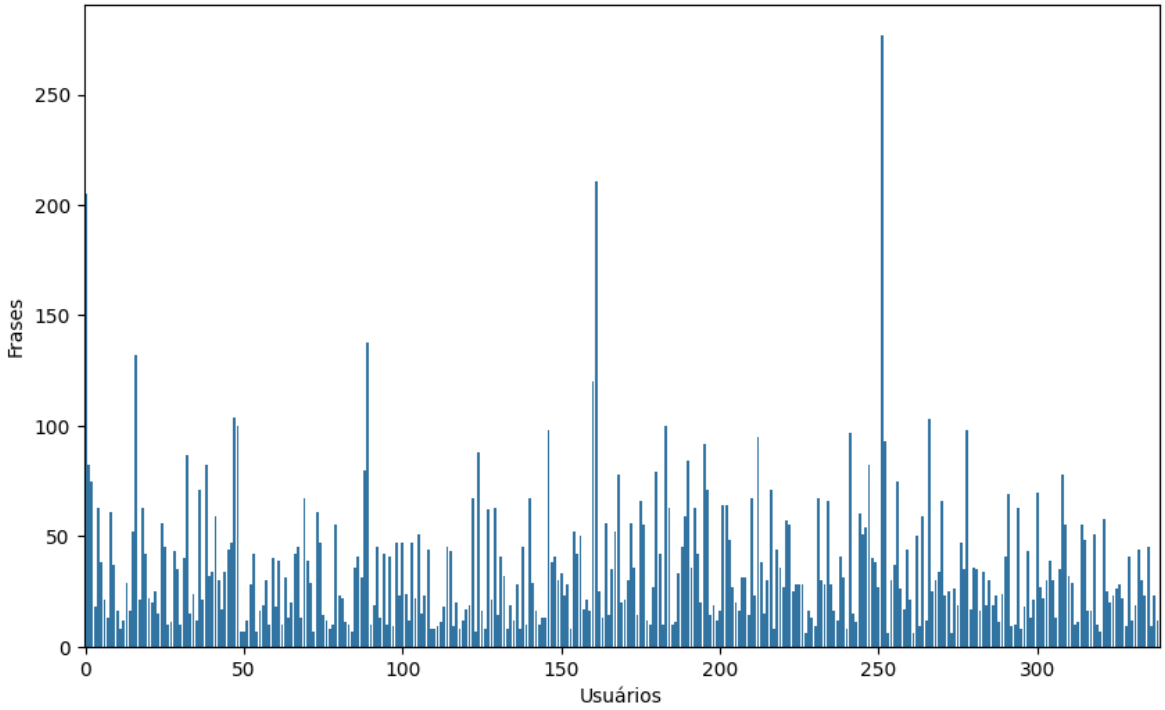

Figura 6 – Frases por usuário

itens, usuários e informações contextuais nos modelos de recomendação. Além disso, viabiliza o treinamento de modelos multitarefa que otimizam conjuntamente diversos objetivos de recomendação (TensorFlow Recommenders Authors, 2023c).

No primeiro modelo foi usada a abordagem presente na biblioteca TFRS, a tarefa de recuperação (*retrieval task*), que consiste em selecionar um conjunto relevante de candidatos dentre todos os possíveis. Esse conjunto relevante é, então, fornecido como recomendação. Dado que o modelo de recuperação pode lidar com milhões de candidatos, é necessário que ele seja computacionalmente eficiente (TensorFlow Recommenders Authors, 2023a).

Para o segundo modelo foi usada a tarefa de *ranking* (*ranking task*). Um modelo de *ranking* usa a abordagem Learn To Rank (LTR) que permite que determinados itens sejam classificados com base na sua relevância conforme uma determinada consulta. Em outras palavras, o algoritmo a partir de uma lista de itens, gera uma segunda lista com os itens mais relevantes no topo e os itens menos relevantes na parte inferior dessa segunda lista (TensorFlow Recommenders Authors, 2023b).

O Keras foi amplamente utilizado neste trabalho, pois ele facilitou e viabilizou a implementação das abordagens não lineares de RNA. Sendo uma biblioteca do TF, o Keras disponibiliza diversas camadas neurais como classes e isso foi essencial para a construção dos dois modelos neurais (Keras, 2023).

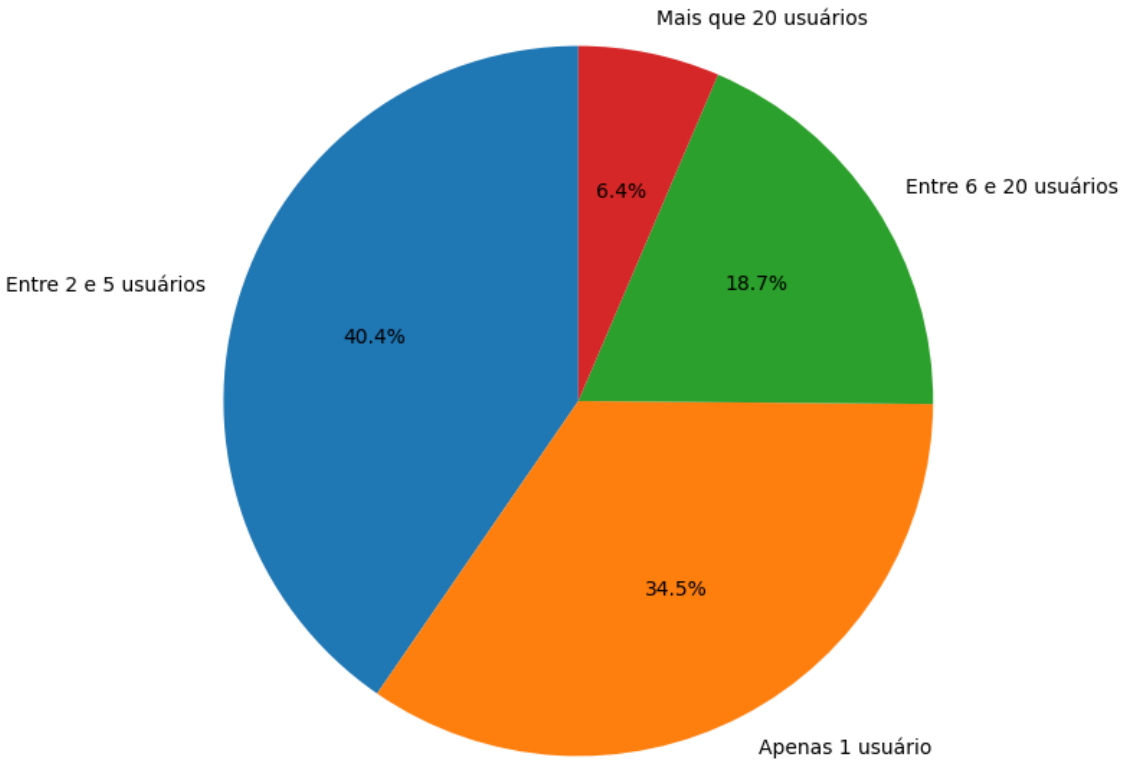

Figura 7 – Porcentagem das frases que são compartilhadas por usuários

3.2.1 Modelo de Recuperação

O modelo de recuperação foi construído a partir da arquitetura de Duas Torres, logo, a torre de consulta e a torre de candidato foram construídas separadamente.

3.2.1.1 Torre de consulta

A torre de consulta consistiu em ser um modelo composto em duas partes. A primeira parte é basicamente uma camada de incorporação das características. Essa camada é responsável por transformar as representações das informações de entrada do modelo de recuperação em vetores densos de números reais:

- *click_hour_embedding*: Recebe os dados em *string* representados por inteiros que vão de 0 a 23, sendo todos os valores possíveis e únicos para as horas do dia. Primeiramente, uma camada ”StringLookup” converte os valores categóricos em índices inteiros, e a camada ”Embedding” gera uma representação distribuída desses índices, ambas as camadas sendo classes da biblioteca Keras (Keras, 2023). O número categórico de saída para representar essa característica foi 4;
- *click_weekday_embedding*: Recebe os dados em *string* representados por inteiros que vão de 0 a 6, representando assim todos os valores possíveis e únicos

para os dias da semana. A construção é semelhante ao anterior, mas para receber os dados de dia da semana. O número categórico de saída para representar essa característica foi 16;

- *location_embedding*: Recebe os dados em *string* representados por inteiros, os números aqui podem variar mediante a clusterização que foi feita a partir da latitude e da longitude de cada usuário para representação das diversas localidades. Semelhante aos anteriores, mas para receber os dados de localização. O número categórico de saída para representar essa característica foi 64;

Logo após, ainda na primeira parte, as incoporações são concatenadas para representar o modelo de incorporações e ele é enviado para a segunda parte da torre de consulta.

A segunda parte da torre de consulta, recebe as características de entrada, passa-as pela camada de incorporações (primeira parte) e, em seguida, alimenta a representação resultante através das camadas densas. A camadas densas são empilhadas progressivamente mediante a escolha feita na criação do modelo, tendo a possibilidade de ter um modelo mais raso até um mais profundo. Essas camadas são separadas por uma função de ativação ”ReLU”, para permitir que as camadas ocultas aprendam as representações dos dados. (TensorFlow Recommenders Authors, 2023a; LI et al., 2022).

Na camada final não foi usada nenhuma função de ativação, pois usar uma função de ativação limitaria o espaço de saída dos *embeddings* finais e poderia impactar negativamente o desempenho do modelo (TensorFlow Recommenders Authors, 2023a). A saída da segunda parte representa a resposta da torre de consulta para as características de entrada.

3.2.1.2 Torre de candidato

Assim como na torre de consulta, a torre de candidato também foi dividida em duas partes, a primeira envolve uma camada de *embedding* e a segunda, sendo as camadas densas, para aprender a combinação não linear das características e gerar uma saída. Os candidatos escolhidos para o modelo foram as frases, pois representam os pictogramas para a recomendação. Dada a natureza dos dados das frases serem textos e não números (como os dados da torre de consulta), algumas mudanças foram feitas na camada de incorporações:

- *phrase_embedding*: Recebe os dados em *string* representados por uma cadeia de caracteres, onde representam os textos falados por cada usuário. De início, a camada ”StringLookup” converte os valores categóricos em índices inteiros, e a

camada ”Embedding” gera uma representação distribuída desses índices (Keras, 2023), semelhante ao que acontece nas incorporações da torre de consulta;

- *phrase_vectorizer*: Usa uma camada ”TextVectorization” do Keras para vetorizar o texto das frases. Essa camada converte o texto em uma sequência de índices inteiros. A camada ”TextVectorization” processa e normaliza o texto;
- *phrase_text_embedding*: Realiza uma incorporação do texto que foi vetorizado usando a camada ”Embedding”, seguida por uma camada ”GlobalAveragePooling1D”, ambas do Keras. A camada de *pooling* global realiza uma média ao longo das dimensões da sequência, produzindo um vetor de características fixo para representar o texto.

Da mesma maneira da segunda parte da torre de consulta, aqui também são definidas e usadas as camadas densas para realizar operações lineares seguidas de ativações ”ReLU”. Essas camadas densas formam uma torre de neurônios que processa as características categóricas representadas pela camada de incorporação. Isso gera uma representação final dessa torre, que reflete na resposta do modelo candidato. Essa abordagem permite que o modelo aprenda representações úteis para as frases, facilitando a tarefa de recuperação e geração de recomendações.

3.2.1.3 Implementação do Modelo de Recuperação

Para concluir, são instanciadas as duas torres já descritas nesta seção. Logo após, é difinida a tarefa de recuperação que compara os *embeddings* gerados pelo modelo de consulta e pelo de candidato. O objetivo é treinar o modelo para que itens relevantes estejam mais próximos em espaço de incorporação. Na Figura 8, pode-se ver como ficou a estrutura desse modelo.

Na tarefa de recuperação desse modelo também foi usado o ScaNN para ajudar o modelo a ser escalável e eficiente, permitindo melhores resultados e uma agilidade na recuperações de itens semelhantes. ScaNN utiliza uma estrutura de índice para representar dados de maneira eficiente e realizar buscas de vizinhos mais próximos de maneira escalável. Ele emprega técnicas como hashing e quantização para reduzir o espaço de busca e acelerar o processo de busca (GUO et al., 2020).

Sendo uma técnica desenvolvida pelo Google Research, ScaNN já é integrado à biblioteca do TFRS, tornando sua implementação simples e prática e o aproveitamento de suas capacidades de maneira direta para o modelo de recuperação (TensorFlow Recommenders Authors, 2024a).

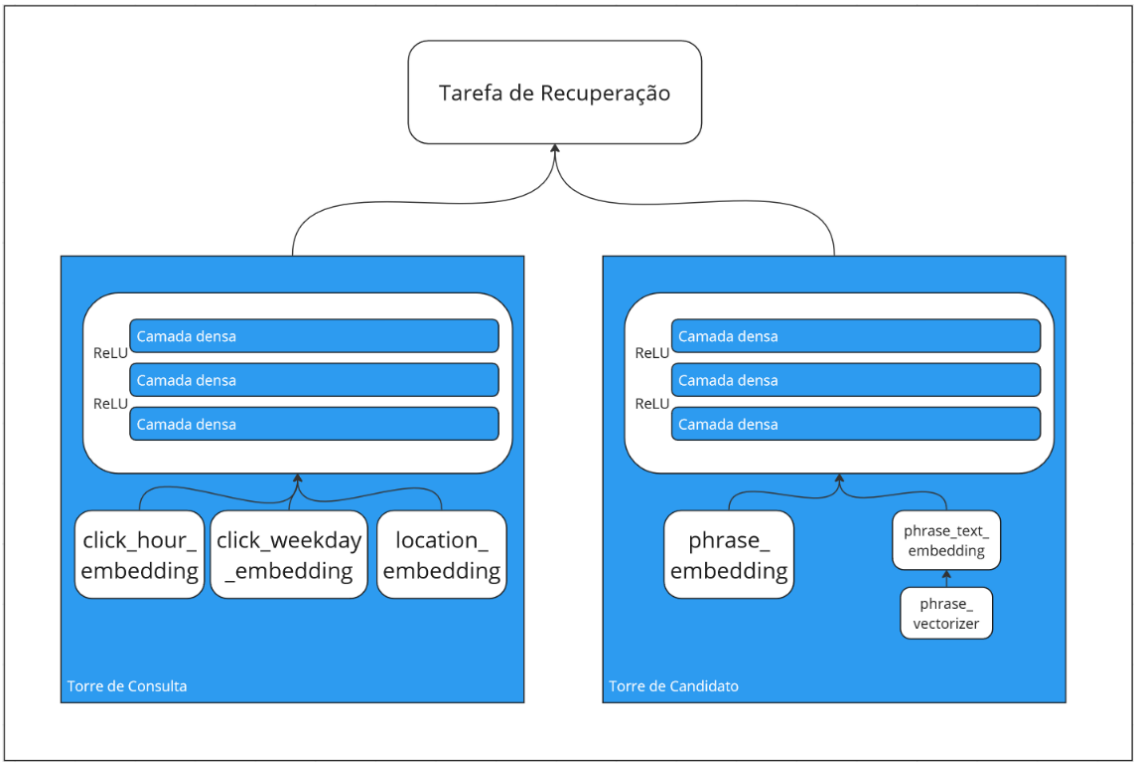

Figura 8 – Arquitetura de Duas Torres do modelo de recuperação

3.2.2 Modelo de Ranqueamento

Conforme visto na documentação do TFRS (TensorFlow Recommenders Authors, 2023b), em tarefas de *ranking*, os modelos não estão tão restritos em termos de eficiência quanto os modelos de recuperação e isso oferece uma maior flexibilidade na escolha das arquiteturas. Uma abordagem comum é a de *feedforward*, empregando várias camadas densas empilhadas. Porém, antes de definir as camadas densas, é criada uma camada de incorporação de maneira análoga às torres do modelo de recuperação descritas anteriormente. Três incorporações são criadas, uma para cada característica do modelo:

- *click_hour_embedding*;
- *click_weekday_embedding*;
- *location_embedding*.

Essas três incorporações, como já mencionado, seguem o mesmo molde estabelecido na torre de consulta do modelo de recuperação. Essas incorporações são concatenadas e uma sequência é criada para a próxima fase que inclui múltiplas camadas densas (totalmente conectadas) com ativação ”ReLU”, permitindo que o modelo aprenda representações mais complexas e hierárquicas à medida que processa as informações.

A última camada usa a função de ativação *softmax*, indicando que o modelo está configurado para realizar previsões de classificação multiclasse. A quantidade de neurônios na última camada corresponde ao número de classes (as frases que representam os texto falados associados aos pictogramas que serão recomendados). Na Figura 9, pode-se ver como ficou a estrutura desse modelo.

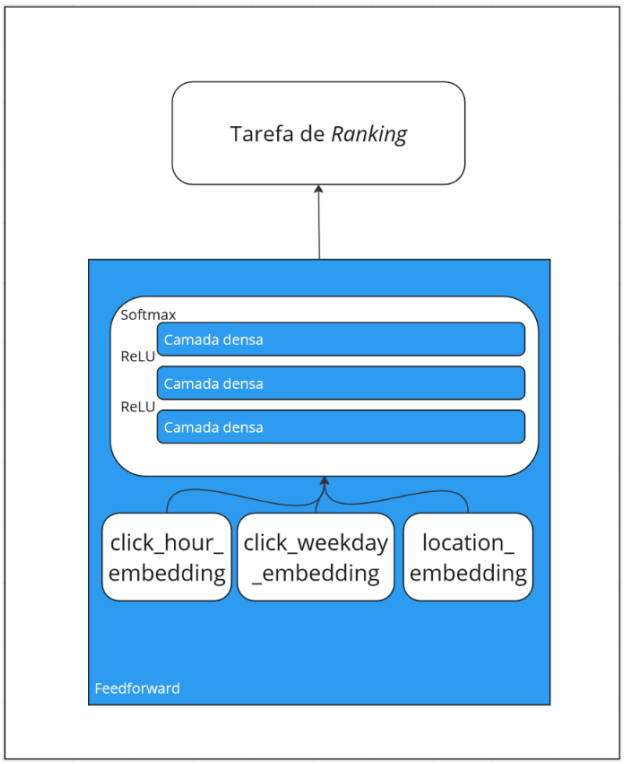

Figura 9 – Arquitetura *feedforward* do modelo de *ranking*

3.2.2.1 Implementação do Modelo de Ranqueamento

Quando uma instância do modelo desse modelo é criada, começa a ser realizadas as incorporações e a computar as previsões finais. Uma tarefa de *ranking* é definida na biblioteca do TFRS para satisfazer a abordagem LTR.

Além disso, utilizou-se a biblioteca TFR que é normalmente empregada para o desenvolvimento de modelos de aprendizagem neural de LTR. Contudo, neste trabalho, foram usadas as implementações das métricas de *recall* e das diversas funções de perdas próprias para o *ranking*, como *pointwise*, *pairwise* e *listwise*, que estão inclusas nessa biblioteca.

3.3 Metodologia de Avaliação

É normal que exista divisão no conjunto de dados tanto para treino, como para teste. Na maioria das vezes, cerca de 70% a 80% dos dados são alocados para o

conjunto de treino. Isso é feito para suavizar o *overfitting*, no qual o modelo pode memorizar as rótulos dos dados de treinamento sem uma boa generalização (GéRON, 2019).

Este trabalho teve 80% dos dados finais separados para os treinamentos dos modelos de recuperação e de *ranking*, e 20% dos dados reservados para avaliar o desempenho dos modelos, sendo esse o conjunto de teste.

Nessa fase, além da separação do conjunto inicial dos dados, também se faz necessária a mudança dos diversos hiperametros para os modelos neurais de recomendação. É de extrema importancia o ajuste nos hiperparâmetros, pois ele também corrobora para a suavização do *overfitting* (ZHANG et al., 2019).

Na Tabela 1 estão listados os hiperparâmetros utilizados, além de fornecer os valores específicos desses hiperparâmetros que foram avaliados para a regularização em cada modelo. É importante esclarecer que, embora diversos hiperparâmetros tenham sido avaliados em cada modelo, esses aqui apresentados foram considerados os mais relevantes durante o processo de regularização.

Modelo	**Hiperparâmetros**	**Valores Analisados**
Modelo de Recuperação ScaNN	Otimizador	Adagrad, Adam, Nadam
	Taxa de Aprendizado	0.1, 0.001, 0.0001
	Camadas (profundidade)	1, 2, 3
	Neurônios (largura)	32, 64, 128
Modelo de Classificação	Otimizador	Adagrad, Adam
	Taxa de Aprendizado	0.5, 0.1, 0.001, 0.0001
	Camadas (profundidade)	2, 3
	Neurônios (largura)	32, 64, 128, 256
	Função de Perda	Pointwise, Listwise, Pairwise

Tabela 1 – Hiperparâmetros e seus valores analisados para os modelos

Como técnica de validação cruzada, foi escolhida o *holdout* (tendo um fluxo de exemplo mostrado na Figura 10), onde se faz necessária a separação entre conjunto treino e teste, como já mencionado. Também no *holdout*, conforme Schorfheide et al. (SCHORFHEIDE; WOLPIN, 2012), diversos treinamentos precisam acontecer, como forma de validar os modelos a partir dos hiperparâmetros indicados na Tabela 1. Dessa forma, todas as possibilidades de hiperparâmetros foram submetidas ao processo de validação cruzada.

Para a escolha dos hiperparâmetros foi usada a abordagem de *grid search* (FEURER; HUTTER, 2019). Para essa abordagem, foram definidos os valores plausíveis para os hiperparâmetros e testados exaustivamente todas as combinações possíveis para os valores. Após os treinamentos e avaliações desses modelos, foram sele-

cionados o melhor modelo de recuperação e de *ranking*, com a pretensão de compará-los.

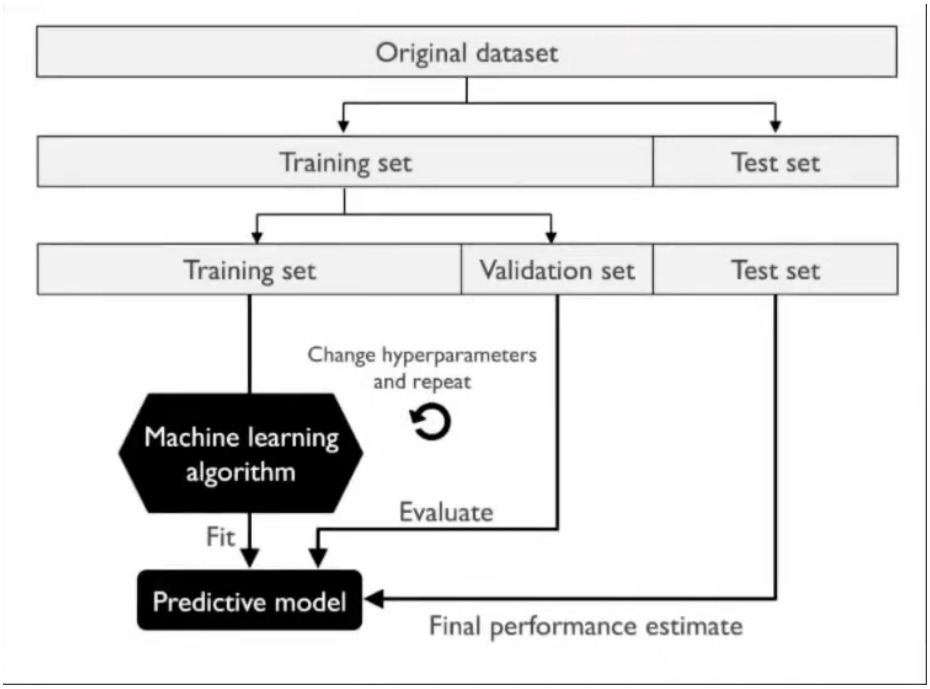

Figura 10 – Exemplo da abordagem *holdout* (RASCHKA, 2024)

O conjunto de dados foi reordenado de maneira que evitasse qualquer viés, seja pela ordenação de itens por usuário ou por alguma outra coluna, ocasionados na extração do *dataset* original. A separação aleatória permitiu que os dados para treino e teste obtivessem a mesma probabilidade de para qualquer linha do conjunto. Após essa separação, a mesma semente foi utilizada em toda fase de validação cruzada.

3.4 Avaliação

A métrica escolhida para a avaliação dos dois modelos foi a *recall*. Essa métrica é comumente usada para avaliação de modelos de classificação e de recuperação (ROSHDI; ROOHPARVAR, 2015; ZEHLIKE; YANG; STOYANOVICH, 2022). O *recall* mede a proporção de exemplos positivos relevantes que foram corretamente identificados pelo modelo em relação ao número total de exemplos positivos na base de dados (MAGDY; JONES, 2010).

Porém, em sistemas de recomendações, é mais interessante, em muitos casos, recomendar apenas os itens mais relevantes para o usuário. Portanto, faz sentido calcular o *recall* nos primeiros ”k” itens em vez de todos os itens, isso também é conhecido como ”*recall* em K” (*recall@k*). A avaliação do desempenho dessa métrica pode ser vista na Figura 11.

$$\mathrm{R@k}(\{y\},\{s\}) = \frac{\sum_i I[\mathrm{rank}(s_i) \leq k]\bar{y}_i}{\sum_j \bar{y}_j}$$

Figura 11 – Fórmula de *recall@k*
(TensorFlow Ranking Authors, 2023)

Neste trabalho foi usada a classe de *recall* já implementada pela biblioteca do TFR. Logo, seguindo a sua documentação (TensorFlow Ranking Authors, 2023) e com base na fórmula apresentada na Figura 5, se tem:

- y sendo a lista de rótulos;
- s sendo a lista de pontuações de classificação;
- k sendo o número de exemplos positivos relevantes a serem recuperados;
- n sendo o número total de exemplos;
- $rank(s_i)$ sendo a posição i após a ordenação pelos scores s com empates quebrados aleatoriamente;
- $I[rank(s_i) \leq k]$ sendo a função indicadora que retorna 1 se $rank(s_i) \leq k$ e 0 caso contrário.

Para os modelos deste trabalho, o valor de k foi definido para 1, 5 e 10, afins de avaliar os resultados mais relevantes, pois em um contexto de aplicação, as recomendações dos pictogramas para os usuários não seria maior do que os valores definidos em k. Logo, com esses valores definidos, pode-se chamar as métricas de *recall@1*, *recall@5* e *recall@10*.

O *recall@1* vai destacar a capacidade de cada modelo de identificar corretamente a informação mais relevante, enquanto o *recall@5* e *recall@10* vai proporcionar uma compreensão mais abrangente da eficácia em casos de maior complexidade, mostrando os valores reais das conquistas de cada modelo em termos de recomendação.

4 Experimentos e Resultados

Este capítulo mostrará como foram realizados os experimentos com os modelos e os resultados encontrados, além de discuti-los, uma vez que foram adotadas as diversas técnicas segundo a literatura, já mencionadas anteriormente.

4.1 Resultados e Discussão

4.1.1 Análise Quantitativa

A Tabela 2 fornece uma visão abrangente do desempenho dos dois modelos descritos no capítulo anterior. Ambos foram submetidos a uma avaliação, utilizando as métricas *recall@1*, *recall@5* e *recall@10*. Ao analisar os resultados apresentados na tabela, é possível discernir o desempenho de cada modelo nas diferentes métricas de *recall* para resultados de caráter quantitivo.

Modelo	***Recall@1***	***Recall@5***	***Recall@10***
ScaNN Retrieval Model	0.0094	0.0762	0.1789
Ranking Model	0.0259	0.1279	0.2394

Tabela 2 – *Recall@k* dos modelos de recuperação e *ranking*

A partir dos resultados da Tabela 2, revelam-se informações importantes sobre o desempenho de cada abordagem, onde a métrica *recall@k* é calculada como a frequência média com que um item está presente nas previsões dos itens definidos em k. Portanto, os valores são as médias dessas frequências para todos os itens de teste (TensorFlow Authors, 2023):

- *recall@1*: Na identificação da melhor opção, o modelo de recuperação apresenta um *recall@1* menor, indicando que, na primeira recomendação, apenas 0.94% das consultas obtêm o item correto. Já o modelo de *ranking* tem um *recall@1* superior, com 2.59% das consultas recebendo a recomendação correta de imediato. Isso sugere que o modelo de *ranking* tem uma capacidade inicial mais robusta de identificar a opção mais relevante;
- *recall@5*: Na amplitude da Recomendação, o modelo de recuperação apresenta um *recall@5* de 7.62%, enquanto o modelo de *ranking* apresenta um valor mais alto de 12.79%, indicando que ele é mais eficaz na apresentação de opções relevantes entre as cinco primeiras;

- *recall@10*: Em uma cobertura extendida, novamente o modelo de *ranking* supera no resultado com um *recall@10* de 23.94% em comparação com 17.89%. Isso sugere que, ao ampliar o escopo da recomendação, o modelo de *ranking* continua a se destacar.

Esses resultados preliminares indicam que o modelo de recuperação teve uma taxa de sucesso menor na identificação imediata da opção correta, uma vez que o modelo de *ranking* se destaca nesse quesito. O segundo modelo também superou o modelo de recuperação em termos de amplitude e de cobertura, o que mostrou uma capacidade mais abrangente de apresentar opções relevantes em diferentes posições do *ranking*.

Contudo, todos os resultados foram relativamente baixos e é interessante analisar do motivo. Uma possível explicação é o baixo valor de k na métrica de *recall*. Conforme já mencionado a fórmula do *recall*, quanto menor for esse valor k, é normal que menor sejam os resultados. Porém, a escolha desses valores foram importantes para a avaliação de um contexto real no aplicativo Livox. Esse baixo valor combinado com o alto número de itens/classes e o baixo valor de informações no *dataset* podem impactar diretamente o desempenho dos modelos (GéRON, 2019).

É importante considerar o contexto específico da aplicação e também compreender que a maneira como os dados são tratados pode alterar significativamente os resultados. Mudanças no conjunto de características, como o tratamento da localização como dados de latitude e longitude, e a inclusão dos idiomas falados pelos usuários, entre outros, poderiam aumentar o número de camadas de incorporações, o que por sua vez impactaria nos resultados. Modelos neurais que se ajustam melhor a informações cada vez mais complexas e extensas tenderiam a apresentar resultados mais promissores.

Outro ponto a ser mencionado é que, como já foi mostrado em outro capítulo, o conjunto de dados selecionado nesse estudo soma quase 53 mil amostras e cerca de dois mil candidatos/classes, segundo a análise exploratória dos dados. Ou seja, uma vez que a literatura implica em mostrar que, para um conjunto de dados massivo, os algoritmos de recuperação tendem a lidar melhor do que algoritmos de classificação (LI et al., 2022). Logo quanto mais itens para recomendar um conjunto tiver, mais dados precisaria para haver uma melhor resposta no *ranking*, logo métricas como *recall* poderiam mudar radicalmente nos resultados (KAMINSKAS; RICCI, 2012).

4.1.2 Análise Qualitativa

A criação de um modelo global faz com que seja muito difícil que os dispositivos de todos os usuários tenham os principais pictogramas recomendados para um

determinado contexto, o que pode resultar na perda de itens recomendados, pois o aplicativo não irá saber lidar com um pictograma inexistente para aquele usuário. Por isso, é interessante comparar as recomendações geradas nas tabelas a seguir, com as frases que representam os pictogramas clicados pelo usuário em questão, a fim de analisar se, em um cenário real, as recomendações se tornarão viáveis para aquele usuário.

Logo, foram feitas duas previsões para ambos os modelos, alterando a hora, o dia da semana e a localização. A Tabela 3 mostra a previsão dos modelos em ordem decrescente por pontuação considerando as caraterísticas do horário das 6 horas, em uma quinta-feira, usando a localização do usuário de identificador ’17186’.

Top	**Modelo de recuperação**	**Modelo de *ranking***
1	”é minha vez”	”secar as mãos”
2	”você está na minha frente”	”faixa de pedestre”
3	”de quem é a vez?”	”família do futuro ”
4	”lixo reciclável ”	”lixo reciclável ”
5	”patrulha canina ”	”velocidade máxima 10 km por hora”
6	”peixinho da maré”	”lápis de cor”
7	”monstros s a”	”você está na minha frente”
8	”você está falando muito alto”	”patrulha canina ”
9	”todo mundo levanta a mão”	”você está falando muito alto”
10	”velocidade máxima 10 km por hora”	”monstros s a”

Tabela 3 – Primeira recomendação dos modelos.

As frases do usuário de identificador ’17186’ foram filtradas e listadas abaixo em ordem de mais cliques:

- ’monstros s a’;
- ’patrulha canina’;
- ’lápis de cor’;
- ’velocidade máxima 10 km por hora’;
- ’você está na minha frente’;
- ’roupas e acessórios’;
- ’você está falando muito alto’;
- ’partes do corpo’;
- ’família do futuro’;
- ’vitamina de banana’;

- ’lixo reciclável’;
- ’secar as mãos’;
- ’faixa de pedestre’;
- ’galera coração’;
- ’é sua vez’;
- ’repete, por favor’;
- ’pão de queijo’;
- ’de quem é a vez?’;
- ’Lavar o rosto’;
- ’é minha vez’;
- ’peixinho da maré’;
- ’formas geométricas’;
- ’todo mundo levanta a mão’;
- ’Cabeça ombro joelho e pé’;
- ’vire a direita’;
- ’eu quero’;
- ’não entendi’;
- ’óculos de sol’;
- ’escovar os dentes’;
- ’lavar as mãos’.

Logo, dadas as principais dez recomendações com as maiores pontuações na Tabela 3, ambos os modelos, em um cenário real e no contexto em que o usuário está inserido, geraram recomendações que correspondem aos pictogramas selecionados pelo usuário durante o uso da aplicação. Portanto, essa análise, embora desconsidere o horário e o dia da semana, é importante para entender se as recomendações atendem à individualidade de cada usuário, de forma que os modelos enfrentem o problema da possível ausência de pictogramas de outros usuários que foram usados no treinamento para um SR global.

Em seguida, a Tabela 4 apresenta a segunda recomendação dos modelos para o horário das 18 horas em um sábado, usando a localização do usuário de identificador ’14725’.

Top	**Modelo de recuperação**	**Modelo de *ranking***
1	”I have a fever”	”M & M”
2	”What are you eating?”	”What is that?”
3	”I miss someone”	”No one wants to play with me”
4	”chocolate chip cookie”	”I want to go to the calm corner”
5	”Please help me to pick an outfit.”	”I want to play in science”
6	”M & M”	”I want to sleep”
7	”please I want to have my hair cut”	”I want to play with puzzles”
8	”Peanut butter and jelly”	”I want to play in the kitchen”
9	”No one wants to play with me”	”I want to take a shower”
10	”When will they arrive?”	”When will they arrive?”

Tabela 4 – Segunda recomendação dos modelos.

Analogamente a previsão anterior, o mesmo foi feito para os resultados da Tabela 4, segue os itens clicados pelo usuário da segunda inferência:

- ’I want to take a shower’;
- ’I want to play in the kitchen’;
- ’I want to play in math’;
- ’I want to play with puzzles’;
- ’I want to go to the calm corner’;
- ’I want to play with sand’;
- ’I want to play in blocks’;
- ’I miss someone’;
- ’Are you alright?’;
- ’I want to drink chocolate milk’;
- ’I want to play in library’;
- ’I want to play in science’;
- ’When will they arrive?’;
- ’I want Apple juice’;
- ’I am afraid’;

- ’please I want to have my hair cut’;
- ’No one wants to play with me’;
- ’Please help me to pick an outfit.’;
- ’I have a fever’;
- ’ask me a question whose answer is yes or no’;
- ’What are you eating?’;
- ’fruit roll up’;
- ’Where are you going?’;
- ’I want to sleep’;
- ’I want to change my diaper’;
- ’I want to eat french fries’;
- ’This is my mama’;
- ’I want to drink juice’;
- ’how are you’;
- ’What is that?’;
- ’I want to drink milk’;
- ’I want to drink water’;
- ’I want to walk on the beach’;
- ’I want to play ball’.

Analisando o segundo exemplo, podemos ver que, eventualmente, os modelos podem gerar recomendações que fujam dos pictogramas clicados pelos usuários, mesmo estando no contexto desse usuário. Provavelmente, para esse segundo caso o modelo pode ter recomendado frases de outro usuário que esteve em um contexto semelhante. Também é necessário ressaltar que esses exemplos não representam a totalidade das recomendações, mas de casos isolados.

Vale ressaltar que a interpretação cuidadosa desses resultados é crucial para uma compreensão aprofundada das capacidades e limitações de cada abordagem.

Este exame crítico das tabelas proporciona uma base sólida para análises mais detalhadas e futuras otimizações, visando aprimorar ainda mais o desempenho desses modelos.

Existe também a possibilidade dos modelos de recuperação e *ranking* serem combinados. Então, por mais que esse trabalho tenha a finalidade de criar e avaliar esses dois modelos, abordagens diferentes podem ser usadas, como pode ser averiguado na documentação do TFRS (TensorFlow Recommenders Authors, 2024c), onde uma tarefa de recuperação é feita para fornecer os itens mais relevantes para um determinado usuário e, a partir desses itens, uma classificação é feita para reproduzir as recomendações finais baseadas nas características fornecidas, almejando melhores resultados.

5 Conclusão

O intuito de incrementar melhorias nas recomendações de pictogramas do aplicativo Livox e o acesso à sua base de dados proporcionaram a elaboração deste trabalho. Contudo, um estudo de temas relacionados a AAC, como educação através de pictogramas e de soluções tecnológicas, além de temas relacionados a AM como recomendação sensível ao contexto e RNAs, convergiu para as propostas aqui estudadas e apresentadas.

Logo, para este trabalho, houve o estudo sobre diversos assuntos relacionados à área de IA, além de suas aplicações, tendo como objetivo geral a construção de dois modelos de recomendação com a novidade da introdução de conceitos e abordagens de RNA, juntamente com a de sensibilidade ao contexto para recomendação de pictogramas em um sistema AAC. Um modelo usando a tarefa de recuperação e outro a tarefa de *ranking* foram criados, treinados e avaliados, utilizando as bibliotecas do TF, principalmente TFRS.

Embora o modelo de recuperação tenha adotado a abordagem de ScaNN, que é relativamente recente e poderosa, a tarefa de *ranking* obteve resultados significativamente melhores nas métricas escolhidas, uma vez que tarefas de recuperação de informações são preferivelmente usadas em conjuntos de dados massivos como mostra o estado da arte.

5.1 Limitações

A criação de um modelo global sensível ao contexto evita que novos usuários fiquem sem recomendações por não serem necessárias a inserção de suas preferências no sistema. A breve análise qualitativa realizada no capítulo anterior nos mostrou que o contexto pode funcionar de maneira eficaz para um usuário já consolidado no sistema. No entanto, caso esse usuário mude seu contexto habitual, os modelos podem recomendar pictogramas que o usuário não possui. A análise exploratória também apresentada neste trabalho tende a amenizar este problema, uma vez que mais da metade dos pictogramas utilizados nos treinamentos são compartilhados por mais de um usuário.

Este trabalho não incluiu uma análise do tempo de inferência dos modelos desenvolvidos. Esse aspecto é muito sensível e relevante, pois influencia diretamente não apenas o desempenho da aplicação que utiliza o SR, mas também a experiência do usuário.

Resultados obtidos por qualquer modelo de aprendizagem são significativamente influenciados pelo conjunto de dados utilizado, logo, a ausência de um CARS com finalidade semelhante aos modelos abordados neste estudo teve um impacto substancial nas discussões sobre os resultados. Um modelo, utilizando uma abordagem mais tradicional ou diferente de AM, ajudaria a definir a complexidade do problema de recomendação em questão. Além disso, possibilitaria uma comparação direta e aprimoraria os resultados das métricas, proporcionando uma análise mais sólida.

5.2 Trabalhos futuros

Afim de avaliar melhor os modelos criados neste trabalho, se faz importante a comparação e análise com outros trabalhos que abarquem um CARS para pictogramas. Uma comparação e avaliação do modelo de recomendação já implementado no Livox atualmente, além de uma melhora no processamento da base de dados dessa aplicação, também são tópicos interessantes a serem estudados.

Outras técnicas de aprendizagem podem ser incrementadas nesses modelos. A aprendizagem incremental para adquirir informações continuamente a partir de dados em constante atualização (LUO; AL., 2020), isso ajudaria a respeitar a sensibilidade do contexto. A aprendizagem federada permite o treinamento local de dados, agregando os modelos no servidor central e distribuindo atualizações para atingir objetivos de aprendizado (ZHANG; AL., 2021). Essa técnica diminuiria a generalização do modelo e ajustaria mais as recomendações para cada usuário mantendo a individualidade e a sensibilidade de contexto. Essas são possíveis abordagens aplicáveis para continuidade deste estudo.

Também, buscando melhores avaliações e como já citado nos capítulos anteriores, estudos como o de Kaminskas et al. (KAMINSKAS; RICCI, 2012) e o de Li et al. (LI et al., 2022), podem ser levados em consideração. A proposta desses trabalhos é a criação de um primeiro modelo (de recuperação) que sirva como um *pre-ranking* para um segundo de *ranking*, onde seria feita a recomendação final. A plaforma do TFRS já abarca essa abordagem e é chamada de ”Multi-task Recommenders” (TensorFlow Recommenders Authors, 2024c). Logo, essa técnica também serviria como um possível trabalho com objetivo de ter melhores resultados nas recomendações de pictogramas.

Referências

ADOMAVICIUS, G.; TUZHILIN, A. Context-aware recommender systems. In: *Recommender Systems Handbook*. Boston, MA: Springer US, 2010. p. 217–253. Citado 2 vezes nas páginas 12 e 15.

ANANDARAJ, A. et al. Book recommendation system with tensorflow. In: IEEE. *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*. [S.l.], 2021. p. 1665–1669. Citado 3 vezes nas páginas 12, 20 e 25.

ANKERST, M. et al. Optics: Ordering points to identify the clustering structure. *ACM Sigmod record*, v. 28, n. 2, p. 49–60, 1999. Citado na página 24.

ARASAAC. *ARASAAC: Portal ARASAAC*. 2024. <https://arasaac.org/>. Acessado em 26 de Fevereiro de 2024. Citado na página 14.

COSTA, F. S. D.; DOLOG, P. Collective embedding for neural context-aware recommender systems. In: *Proceedings of the 13th ACM conference on recommender systems*. [S.l.: s.n.], 2019. p. 201–209. Citado 2 vezes nas páginas 12 e 20.

D'AMICO, E. et al. Leveraging laziness, browsing-pattern aware stacked models for sequential accommodation learning to rank. In: *Proceedings of the Workshop on ACM Recommender Systems Challenge*. [S.l.: s.n.], 2019. p. 1–5. Citado 2 vezes nas páginas 17 e 20.

FEURER, M.; HUTTER, F. Hyperparameter optimization. In: *Automated Machine Learning: Methods, Systems, Challenges*. [S.l.: s.n.], 2019. p. 3–33. Citado na página 32.

GARG, S.; SHARMA, S. Impact of artificial intelligence in special need education to promote inclusive pedagogy. *International Journal of Information and Education Technology*, v. 10, n. 7, p. 523–527, 2020. Citado 2 vezes nas páginas 11 e 14.

GIL, A. C. *Como elaborar projetos de pesquisa*. São Paulo: Atlas, 2002. v. 4. 175 p. Citado na página 21.

GUO, R. et al. Accelerating large-scale inference with anisotropic vector quantization. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2020. p. 3887–3896. Citado na página 29.

GéRON, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 1. ed. [S.l.: s.n.], 2019. Citado 3 vezes nas páginas 22, 32 e 36.

HERNáNDEZ, S. S. et al. User-centric recommendation model for aac based on multi-criteria planning. In: *International Conference on Advances in Human-oriented and Personalized Mechanisms, Technologies and Services*. [S.l.: s.n.], 2014. Citado 2 vezes nas páginas 12 e 19.

HEUMADER, P.; MURILLO-MORALES, T.; MIESENBERGER, K. Ai based recommendation and assessment of at for people with cognitive disabilities. *J. Technol. Pers. Disabil*, v. 10, 2022. Citado na página 11.

HIGGINBOTHAM, D. J. et al. Access to aac: Present, past, and future. *Augmentative and Alternative Communication*, v. 23, n. 3, p. 243–257, 2007. Citado 2 vezes nas páginas 11 e 14.

KAMINSKAS, M.; RICCI, F. Contextual music information retrieval and recommendation: State of the art and challenges. *Computer Science Review*, v. 6, n. 2-3, p. 89–119, 2012. Citado 2 vezes nas páginas 36 e 43.

Keras. *Keras*. 2023. <https://keras.io/>. Acessado em 29 de dezembro de 2023. Citado 3 vezes nas páginas 26, 27 e 29.

KULKARNI, S.; RODD, S. F. Context aware recommendation systems: A review of the state of the art techniques. *Computer Science Review*, v. 37, p. 100255, 2020. Citado 4 vezes nas páginas 11, 12, 15 e 16.

LI, X. et al. Inttower: The next generation of two-tower model for pre-ranking system. In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. [S.l.: s.n.], 2022. p. 3292–3301. Citado 5 vezes nas páginas 16, 17, 28, 36 e 43.

LIGHT, J.; DRAGER, K. Aac technologies for young children with complex communication needs: State of the science and future research directions. *Augmentative and Alternative Communication*, v. 23, n. 3, p. 204–216, 2007. Citado 2 vezes nas páginas 11 e 14.

LIVOX. *Livox - Comunicação Alternativa com IA*. 2023. <https://livox.com.br/br/>. Acessado em 23 de dezembro de 2023. Citado 2 vezes nas páginas 12 e 14.

LUO, Y.; AL. et. An appraisal of incremental learning methods. *Entropy*, v. 22, n. 11, p. 1190, 2020. Citado na página 43.

MAGDY, W.; JONES, G. J. F. Pres: A score metric for evaluating recall-oriented information retrieval applications. In: *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. [S.l.: s.n.], 2010. p. 611–618. Citado na página 33.

MICHALSKI, R. S.; CARBONELL, J. G.; MITCHELL, T. M. (Ed.). *Machine Learning: An Artificial Intelligence Approach*. [S.l.]: Springer Science & Business Media, 2013. Citado na página 22.

MOHAMMED, R.; RAWASHDEH, J.; ABDULLAH, M. Machine learning with oversampling and undersampling techniques: Overview study and experimental results. In: IEEE. *2020 11th International Conference on Information and Communication Systems (ICICS)*. [S.l.], 2020. p. 243–248. Citado na página 23.

MU, R. A survey of recommender systems based on deep learning. *IEEE Access*, v. 6, p. 69009–69022, 2018. Citado 2 vezes nas páginas 12 e 16.

NEAMTU, R. et al. Using artificial intelligence for augmentative alternative communication for children with disabilities. In: *Human-Computer Interaction–INTERACT 2019: 17th IFIP TC 13 International Conference, Paphos, Cyprus, September 2–6, 2019, Proceedings, Part I*. [S.l.]: Springer International Publishing, 2019. p. 234–243. Citado 2 vezes nas páginas 12 e 19.

PASUMARTHI, R. K. et al. Tf-ranking: Scalable tensorflow library for learning-to-rank. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. [S.l.: s.n.], 2019. p. 2970–2978. Citado na página 20.

RAKKAPPAN, L.; RAJAN, V. Context-aware sequential recommendations with stacked recurrent neural networks. In: *The World Wide Web Conference*. [S.l.: s.n.], 2019. p. 3172–3178. Citado na página 20.

RASCHKA, S. *Holdout Technique*. 2024. <https://github.com/rasbt/stat479-machine-learning-fs18/tree/master>. Acessado em 05 de janeiro de 2024. Citado na página 33.

ROSHDI, A.; ROOHPARVAR, A. Information retrieval techniques and applications. *International Journal of Computer Networks and Communications Security*, v. 3, n. 9, p. 373–377, 2015. Citado 3 vezes nas páginas 18, 19 e 33.

SASSI, I. B.; MELLOULI, S.; YAHIA, S. B. Context-aware recommender systems in mobile environment: On the road of future research. *Information Systems*, v. 72, p. 27–61, 2017. Citado 3 vezes nas páginas 11, 12 e 15.

SCHORFHEIDE, F.; WOLPIN, K. I. On the use of holdout samples for model selection. *American Economic Review*, v. 102, n. 3, p. 477–481, 2012. Citado na página 32.

SHANE, H. C. et al. Using aac technology to access the world. *Assistive Technology*, v. 24, n. 1, p. 3–13, 2012. Citado na página 11.

SKALSKI, P. *Feed-forward Architecture*. 2024. <https://towardsdatascience.com/lets-code-a-neural-network-in-plain-numpy-ae7e74410795>. Acessado em 07 de janeiro de 2024. Citado na página 18.

TensorFlow Authors. *Recall Keras*. 2023. <https://www.tensorflow.org/api_docs/python/tf/keras/metrics/Recall>. Acessado em 29 de dezembro de 2023. Citado na página 35.

TensorFlow Ranking Authors. *Recall Ranking*. 2023. <https://www.tensorflow.org/ranking/api_docs/python/tfr/keras/metrics/RecallMetric>. Acessado em 29 de dezembro de 2023. Citado na página 34.

TensorFlow Recommenders Authors. *Building deep retrieval models*. 2023. <https://www.tensorflow.org/recommenders/examples/deep_recommenders>. Acessado em 28 de dezembro de 2023. Citado 3 vezes nas páginas 16, 26 e 28.

TensorFlow Recommenders Authors. *Recommending movies: ranking*. 2023. <https://www.tensorflow.org/recommenders/examples/basic_ranking>. Acessado em 28 de dezembro de 2023. Citado 2 vezes nas páginas 26 e 30.

TensorFlow Recommenders Authors. *TensorFlow Recommenders*. 2023. <https://www.tensorflow.org/recommenders/>. Acessado em 28 de dezembro de 2023. Citado 2 vezes nas páginas 25 e 26.

TensorFlow Recommenders Authors. *Efficient Serving*. 2024. <https://www.tensorflow.org/recommenders/examples/efficient_serving>. Acessado em 04 de janeiro de 2024. Citado na página 29.

TensorFlow Recommenders Authors. *Feature Preprocessing*. 2024. <https://www.tensorflow.org/recommenders/examples/featurization>. Online; accessed 02-February-2024. Citado na página 24.

TensorFlow Recommenders Authors. *Multi-task Recommenders*. 2024. <https://www.tensorflow.org/recommenders/examples/multitask>. Online; accessed 04-January-2024. Citado 2 vezes nas páginas 41 e 43.

TIJUS, C. et al. The design, understanding and usage of pictograms. written documents in the workplace. p. 17–31, 2007. Citado na página 12.

WU, Y. C.; FENG, J. W. Development and application of artificial neural network. *Wireless Personal Communications*, v. 102, p. 1645–1656, 2018. Citado 2 vezes nas páginas 16 e 18.

ZEHLIKE, M.; YANG, K.; STOYANOVICH, J. Fairness in ranking, part ii: Learning-to-rank and recommender systems. *ACM Computing Surveys*, v. 55, n. 6, p. 1–41, 2022. Citado 2 vezes nas páginas 19 e 33.

ZHANG, C.; AL. et. A survey on federated learning. *Knowledge-Based Systems*, v. 216, p. 106775, 2021. Citado na página 43.

ZHANG, S. et al. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, v. 52, n. 1, p. 1–38, 2019. Citado 4 vezes nas páginas 12, 16, 18 e 32.