UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

ERICK CORREIA SILVA

ANÁLISE DE TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA RANQUEAMENTO PARA PRODUTOS

QUIXADÁ

2023

ERICK CORREIA SILVA

ANÁLISE DE TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA RANQUEAMENTO PARA PRODUTOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação do CAMPUS QUIXADÁ da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Computação.

Orientador: Prof. Dr. Victor Aguiar Evangelista de Farias.
Coorientador: Dr. Ivo Paixão de Medeiros.

QUIXADÁ
2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S579a Silva, Erick Correia.
Análise de técnicas de aprendizagem de máquina para ranqueamento para produtos / Erick Correia Silva. – 2023.
44 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Computação, Quixadá, 2023.
Orientação: Prof. Dr. Victor Aguiar Evangelista de Farias.
Coorientação: Prof. Dr. Ivo Paixão de Medeiros.

1. Inteligência Artificial. 2. Métodos de aprendizagem. 3. Recuperação da Informação . I. Título.
CDD 621.39

ERICK CORREIA SILVA

ANÁLISE DE TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA RANQUEAMENTO PARA PRODUTOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação do CAMPUS QUIXADÁ da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Computação.

Aprovada em: 02/06/2023.

BANCA EXAMINADORA

Prof. Dr. Victor Aguiar Evangelista de Farias (Orientador)
Universidade Federal do Ceará (UFC)

Dr. Ivo Paixão de Medeiros (Coorientador)
Instituto Nacional de Pesquisas Espaciais (INPE)

Profa. Dra. Lívia Almada Cruz
Universidade Federal do Ceará (UFC)

Dr. Iago Chaves
Universidade Federal do Ceará (UFC)

À minha família, por sua capacidade de acreditar em mim e investir em mim. Mãe, seu cuidado e dedicação foi que deram, em alguns momentos, a esperança para seguir. Pai, sua presença significou segurança e certeza de que não estou sozinho nessa caminhada.

AGRADECIMENTOS

Meu Deus, obrigado pelos teus planos para minha vida, pois são sempre maiores que meus próprios sonhos.

Agradeço aos meus pais, principalmente a minha mãe Maria Eliane Silva Correia, por seu apoio, incentivo nas cobranças, nas horas difíceis de desânimo e cansaço. Agradeço a meu querido irmão Lucas Correia Silva, por toda a ajuda e acompanhamento em todas fases de minha vida.

Aos meus orientadores, professor Dr. Victor Aguiar Evangelista de Farias, e Dr. Ivo Paixão de Medeiros, por toda dedicação, disposição, auxílio e orientação realizados a mim durante toda a execução deste trabalho.

A todos meus colegas de trabalho da plataforma de inteligência artificial da Americanas S/A, em especial quero agradecer a Lucas Elias Cardoso Rocha que se dispôs e me ensinar muito sobre o tema deste trabalho

A todos os meus colegas e amigos da graduação, por tudo, aprendizados e amizades construídas ao longo destes anos e todos os momentos compartilhados. Em especial quero agradecer aos meus amigos do laboratório de robótica, Samuel Henrique, Anderson Silva Sousa, Antonio César, Gabriel Moreira Nunes e Abdul Moreira que estiveram comigo desde o início desta jornada, proporcionando inúmeros momentos que irei recordar para sempre em minha vida.

A todos os professores da Universidade Federal do Ceará, Campus Quixadá, que colaboraram no processo de minha formação profissional e pessoal, com muito empenho e profissionalismo, compartilhando seus conhecimentos e vivências.

A todos que fizeram parte deste ciclo vivido ao longo desses 5 anos e contribuíram de alguma forma na minha formação profissional e pessoal.

"O perigo de verdade não é que computadores passem a pensar como humanos, mas sim que humanos passem a pensar como computadores."
(Sydney Harris)

RESUMO

Propõe-se com este trabalho a análise de técnicas de aprendizagem de ranqueamento utilizando como base o conjunto de dados Shopping Queries Dataset disponibilizado pela Amazon, na competição KDD Cup 2022. Para isso foi realizado o experimento com quatro modelos de aprendizagem de ranqueamento que foram escolhidos com base nos artigos escritos após a competição. Os experimentos foram avaliados através do ganho cumulativo com desconto normalizado (NDCG), métrica de classificação de várias etiquetas utilizada em problemas de aprendizagem de máquina. Por fim é discutido quais os principais desafios para implementação dos modelos em ambiente de produção real utilizando como referência a empresa de vendas online Americanas SA.

Palavras-chave: Apredizagem de Maquina; Aprendizagem de Ranqueamento; Recuperação de Informação.

ABSTRACT

This work proposes the analysis of ranking learning techniques using as a basis the Shopping Queries Dataset provided by Amazon, in the KDD Cup 2022 competition. For this, an experiment was carried out with four ranking learning models that were chosen based on the articles written after the competition, and were evaluated based on. The experiments were evaluated using the cumulative gain with normalized discounting (NDCG), a multi-label classification metric used in machine learning problems. Finally, the main challenges for implementing the models in a real production environment are discussed, using the online sales company Americanas SA as a reference.

Keywords: Machine Learning; Learning to Rank; Information Retrieval.

LISTA DE TABELAS

Tabela 1 – Quantidade de atributos, consultas e documentos das coleções utilizadas. . . 25
Tabela 2 – Avaliação comparativa entre Random Forest e Deep Forest. 26
Tabela 3 – Avaliação comparativa do Deep Forest, Deep Forest-LambdaMART e LambdaMART 26
Tabela 4 – Avaliação comparativa entre Deep Forest, Deep Forest-LambdaMART e Rede Neural Profunda 26
Tabela 5 – Pontuação dos 4 primeiros da tarefa 1. 28
Tabela 6 – Tabela comparativa. 30
Tabela 7 – Resumo do conjunto de dados de consultas 34

SUMÁRIO

1	**INTRODUÇÃO**	11
1.1	**Problematização**	11
1.2	**Objetivos**	12
2	**FUNDAMENTAÇÃO TEÓRICA**	13
2.1	**Motores de Busca**	13
2.2	**Motores de Busca Centralizados**	13
2.3	**Motores de Busca Descentralizados**	14
2.4	**Aprendizagem de Máquina**	15
2.5	**Aprendizado Supervisionado**	15
2.6	**Learning to Rank**	16
2.7	**Abordagens de *Leaning to Rank***	18
2.8	**Transformers**	19
2.9	**Cross Encoders**	20
2.10	**BERT**	21
2.11	**Métricas de Avaliação em LeToR**	22
2.12	**NDCG**	23
3	**TRABALHOS RELACIONADOS**	25
3.1	**Análise da Técnica Deep Forest para o Problema de Aprendizado de Ranqueamento**	25
3.2	**A Winning Solution of KDD CUP 2022 ESCI Challenge for Improving Product Search**	27
3.3	**A Semantic Alignment System for Multilingual Query-Product Retrieval**	28
3.4	**Análise Comparativa**	29
4	**PROCEDIMENTOS METODOLÓGICOS**	31
4.1	**Análise Exploratória dos Dados**	31
4.2	**Pré-processamento dos dados**	31
4.3	**Critérios de Escolha e Seleção dos Modelos**	32
4.4	**Treinamento dos modelos**	32
4.5	**Avaliação dos modelos**	32
4.6	**Desafios para deploy dos modelos**	33

5 EXPERIMENTOS E RESULTADOS 34
5.1 Análise Exploratória 34
5.2 Pré-processamento dos Dados 35
5.3 Modelos e Treinamento 35
5.3.1 Experimento 01: MS Marco 36
5.3.2 Experimento 02: DeBERTa-v3 37
5.3.3 Experimento 03: LambdaMART 38
5.3.4 Experimento 04: LambdaMART + TF-IDF 39
5.3.5 Análise dos Resultados 39
5.4 Principais Desafios 40
5.4.1 Obtenção dos dados 40
5.4.2 Ambiente de Desenvolvimento e Implatação 40
6 CONCLUSÕES E TRABALHOS FUTUROS 42
6.1 Trabalhos Futuros 42
REFERÊNCIAS 43

1 INTRODUÇÃO

Motores de busca são ferramentas utilizadas na área da Recuperação da Informação (RI) que tem como finalidade ajudar na busca de informações mantidas em ambientes computacionais, ou seja, são sistemas encarregados de reportar as informações almejadas, com um alto índice de adequação, de forma que quanto mais rápido se apresentar o resultado, mais útil se tornará para quem faz a busca (CAMOSSI *et al.*, 2022). Com isso, dentro do campo da recuperação da informação há uma área chamada *Learning to Rank* (Aprendizagem de Ranqueamento) que aborda ranqueamento de documentos considerando a relevância destes nas consultas.

Learning to Rank (LeToR) é uma área de pesquisa que aborda ranqueamento de documentos considerando a relevância destes nas consultas. A principal responsabilidade de LeToR é aplicar técnicas de aprendizagem de maquina para construção de modelos capazes de gerar permutações de documentos baseado nos relacionamentos aprendido entre os seus atributos e o grau de relevância dos mesmos para uma consulta (ROCHA *et al.*, b).

Melhorar a relevância dos resultados de busca de produtos pode auxiliar significativamente a experiência do cliente e a chance de comprar aquele produto (CHOUDHARY *et al.*, a). Apesar dos avanços recentes no campo do aprendizado de máquina, retornar corretamente os itens para uma determinada busca de produto ainda é um desafio. A presença de informações ruidosas nos resultados, a dificuldade de compreensão da intenção da consulta e a diversidade dos itens disponíveis são alguns dos motivos que contribuem para a complexidade desse problema (CHOUDHARY *et al.*, b).

Existem diversas atividades em RI onde algoritmos de LeToR podem ser considerados parte central das soluções, como, por exemplo, Sistemas de Recomendação (KARATZOGLOU *et al.*, 2013) e Sistemas de Pergunta e Resposta (PîRTOACă *et al.*, 2019). Porém, o principal problema abordado com LeToR é a recuperação de documentos por meio de motores de busca.

1.1 Problematização

O problema que este trabalho pretende resolver é de busca de produtos, ou seja, dado uma consulta do usuário deve-se retornar uma lista de produtos, que sejam mais relevantes para aquela pesquisa dentro do catálogo de produtos, preferencialmente, que essa lista seja ordenada por relevância.

Algumas das soluções utilizadas atualmente para isso são RankNet, LambdaRank e LambdaMART. Em todas as três técnicas, o ranqueamento é transformado em um problema de classificação ou regressão. Isso significa que os modelos propostos analisam pares de itens, no caso deste trabalho e feito a analise do par *consulta* / *produto*, essa análise e feita uma de cada vez, para no fim criar uma ordem ideal para o par, depois a usa para chegar ao ranque final de todos os resultados.

Neste trabalho pretende-se investigar a aplicação de abordagem baseada em aprendizagem de maquina. Para isso será utilizado a primeira tarefa da KDD Cup 2022 Workshop: ESCI Challenge for Improving Product Search (CUP,) que tem como objetivo ranquear os produtos para que os produtos relevantes sejam ranqueados acima dos não relevantes. Isso é semelhante às tarefas padrão de recuperação de informações, mas especificamente no contexto de pesquisa de produtos no comércio eletrônico

1.2 Objetivos

Dessa forma, o objetivo geral deste trabalho é gerar um comparativo entre algoritmos de LeToR (comparando métricas de desempenho) para desenvolver modelos de busca de produto baseado em machine learning.

Como objetivos específicos estão:

- Selecionar e implementar modelos de LeToR utilizando o dataset “Shopping Queries Data Set” para classificar os produtos, para que os produtos relevantes sejam classificados acima dos não relevantes.
- Gerar uma análise com objetivo de listar quais os principais desafios e soluções para implementação do projeto em produção.

O restante do trabalho está organizado da seguinte maneira. No Capítulo 2 são apresentados os conceitos necessários para o entendimento do trabalho. O Capítulo 3 apresenta trabalhos relacionados ao presente trabalho. Em seguida, no Capítulo 4 estão os procedimentos metodológicos a serem empregados para o desenvolvimento deste trabalho. No Capítulo 5 estão os resultados obtidos neste trabalho. Por fim, as considerações finais são apresentadas no Capítulo 6.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, serão apresentados os principais conceitos necessários para o entendimento e desenvolvimento do projeto proposto neste trabalho.

2.1 Motores de Busca

Segundo (GABRIEL, 2020) motores de busca são sistemas de recuperação de informações mantidas em ambientes computacionais na Web, isso significa que tais sistemas são designados para buscar a informação desejada e, portanto quanto mais preciso e rápido o resultado for entregue, mais útil se tornará para quem faz a busca. O objetivo dos motores de busca é usar o significado da consulta para melhorar a experiência de pesquisa do usuário. Existem quatro abordagens para a busca semântica, são eles: Análise Contextual, Raciocínio, Compreensão de Linguagem Natural, Representação de Conhecimento Baseada em Ontologia.

A Figura 1 especifica uma arquitetura básica de um motor de busca para URLs na internet. Primeiro o cliente fornece a consulta (palavras-chaves) ao mecanismo de consulta, em paralelo o rastreador busca os dados da WWW e armazena no banco de dados, em seguida o módulo de índice indexa os dados, armazenando texto no banco de dados e armazenando a estrutura no banco de dados de estrutura. O módulo de análise de coleções recolhe os elementos do banco de dados e dos dados indexados. Por fim o módulo de ranqueamento retorna ao cliente a consulta ordenada de acordo com a exatidão.

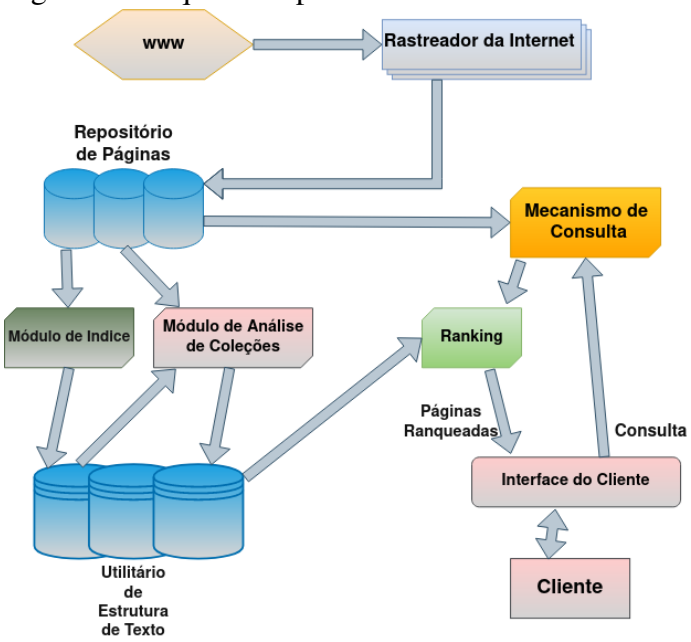
De acordo com (REZAEE, 2021), os motores de busca classificam-se como centralizados (padrão) e descentralizados . A abordagem dominante para a busca convencional é centralizada. Essa abordagem possui algumas limitações que são descritas a seguir.

2.2 Motores de Busca Centralizados

A abordagem primária para sistemas de busca convencionais é centralizada (REZAEE, 2021). A centralização neste contexto refere-se aos métodos de controle. Os sistemas centralizados são controlados por um único grupo (REZAEE, 2021). A maioria das arquitetura dos motores de busca consiste em três módulos, sendo elas:

- Rastreador web: que tem como objetivo coletar informações
- Sistema de Arquivos ou banco de dados: Responsável por armazenar informações
- Subsistema de busca: responsável por encontrar o documento relevante para uma consulta

Figura 1 – Arquitetura padrão de um motor de busca



Fonte: (SINGH, 2012))

– Representação de Conhecimento Baseada em Ontologia

A seguir, são descritos com mais detalhes os dois principais módulos de motores de busca.

O rastreador: é um subsistema que recupera páginas da Web e cria um índice baseado em texto. Um rastreador é um tipo de software que, dado um ou mais URLs de semente, baixa as páginas da web associadas a essas URLs, extrai quaisquer hiperlinks e, em seguida, baixa as páginas da web recursivamente (TROTMAN; ZHANG, 2013).

O Subsistema de buscar: é um componente que processa um fluxo de consultas. O algoritmo de pesquisa ranqueia os documentos de acordo com vários parâmetros, incluindo a quantidade de ocorrências de texto âncora. Contagens de ocorrências de diferentes tipos em vários níveis de proximidade são calculadas para cada documento correspondente. Essas contagens são então roteadas através de muitas tabelas de pesquisa antes de serem convertidas em uma classificação. O objetivo da pesquisa é fornecer resultados de alta qualidade rapidamente.

2.3 Motores de Busca Descentralizados

Um mecanismo de pesquisa descentralizado não possui um único servidor. Nesse tipo de sistema de busca, as tarefas como rastreamente, indexação e processamento de consultas são distribuídas entre vários pontos de forma descentralizada, sem ponto único de controle, ao contrário do centralizado convencional.

FAROO e YaCy são dois exemplos de mecanismos de pesquisa distribuídos. Ambos

são baseados em conceitos de rede ponto a ponto (P2P). O Mecanismo de pesquisa descentralizado propõe resolver os problemas de privacidade que existem em sistemas de busca centralizados (REZAEE *et al.*, 2021) . Os desafios de privacidade incluem censura, exposição de consultas pessoais à pesquisa, falta de transparência dos algoritmos de pesquisa proprietários e do servidor e quaisquer terceiros com quem o operador do serviço de pesquisa possa compartilhar dados.

2.4 Aprendizagem de Máquina

O Aprendizado de Máquina é a ciência e, também, a arte da programação de computadores de forma a possibilitar que eles sejam capazes de aprender com os dados (GÉRON, 2019). No nível mais alto de abstração, podemos pensar em aprendizagem de máquina como um conjunto de ferramentas e métodos que tentam inferir padrões e extrair percepções de um registro do mundo observável (CONWAY, 2012). Ou seja, dentro de algum contexto podemos fazer um registro das ações de nossos problemas, fazendo com que aprenda com esse registro e depois crie um modelo dessas atividades que informe nossa compreensão desse contexto daqui para frente. Na prática, isso requer dados,e em aplicações contemporâneas isso geralmente significa muitos dados.

Duas categorias importantes do Aprendizado de Máquina são o aprendizado não supervisionado e o aprendizado supervisionado. Neste trabalho será utilizado aprendizado supervisionado, na aplicação de algoritmos de *Learning to rank*.

2.5 Aprendizado Supervisionado

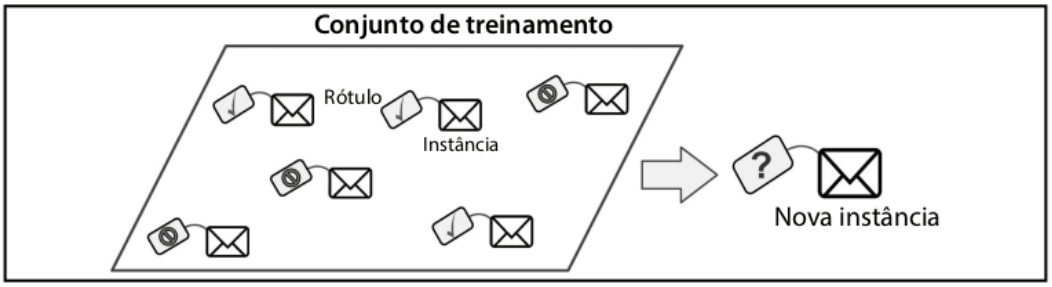
O aprendizado supervisionado, é uma subcategoria de aprendizado de máquina e inteligência artificial, que pode ser definido pelo uso de conjuntos de dados rotulados para treinar algoritmos que classificam dados ou predizem resultados com precisão (HUB, 2020).

O aprendizado supervisionado usa um conjunto de treinamento para ensinar os modelos a produzir a saída desejada. Esse conjunto de dados de treinamento inclui entradas e saídas corretas, que permitem que o modelo aprenda ao longo do tempo (HUB, 2020). O algoritmo mede sua precisão através da função de perda, ajustando até que o erro tenha sido suficientemente minimizado.

O aprendizado supervisionado ajuda as organizações a resolver uma variedade de problemas do mundo real em escala, como classificar spam em uma pasta separada de sua caixa

de entrada, como mostrado na figura a seguir.

Figura 2 – Um conjunto de treinamento rotulado para aprendizado supervisionado, por exemplo, classificação de spam



Fonte: (GÉRON, 2019))

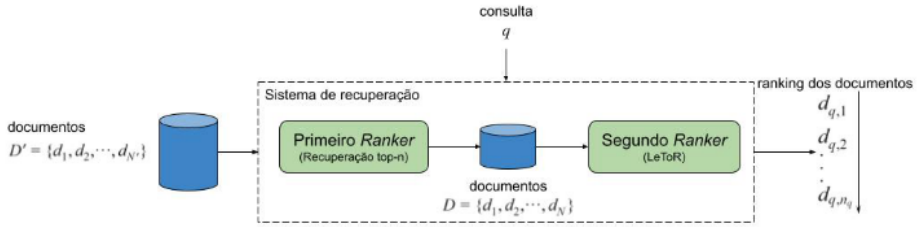
Aprendizagem é uma investigação através do espaço de hipóteses com grandes chances de acontecer por aquele que terá um bom desempenho, mesmo em novos exemplos, excluindo os do conjunto de treinamento. Para calcular a precisão de uma hipótese, é fornecido um conjunto de testes de exemplos diferentes do conjunto de treinamento (RUSSELL, 2013).

2.6 Learning to Rank

Learning to rank é o processo de construção de um modelo para ranquear documentos ou objetos. É útil para muitas aplicações, como recuperação de informações (RI). Segundo (ROCHA *et al.*, b) uma máquina de busca tem por objetivo responder a uma consulta de um usuário com um conjunto de documentos ordenados por relevância em relação à própria consulta. Como podemos observar na figura 2, uma máquina de busca possui dois processos consecutivos de ranqueamento. O primeiro é um ranqueamento base que não envolve LeToR que visa recuperar o maior número de documentos, mesmo que alguns não sejam relevantes à consulta, prezando o recall, enquanto o segundo é um ranqueamento que procura aumentar a precisão, obtido por LeToR.

A Figura 3 ilustra um sistema de Recuperação da Informação utilizando *Learning to Rank*. Inicialmente temos o conjunto de treinamento dado como entrada. Um exemplo desse conjunto consiste em consultas, onde os documentos associados são representados pelo vetor de ncaracterísticas, no qual "n"representa o número de documentos associados à consulta e o grau de relevância correspondente.. Como trata-se de um processo de aprendizado supervisionado, precisamos também de um conjunto de teste que é estruturado da mesma forma que o conjunto de treinamento. Por fim temos um modelo de aprendizado que aprenderá a função de ranking

Figura 3 – Visão global de uma máquina de busca com destaque para os processos de ranqueamento.



Fonte: (ROCHA *et al.*, b))

capaz de predizer a relevância dos documentos para as novas consultas.

Considerando um conjunto D' contendo uma grande quantidade de documentos, nem todos são relevantes para uma consulta qualquer 'q'. Assim, o primeiro ranqueamento ocorre sobre a coleção de documentos D' = d1, d2,... , dN' e visa selecionar um subconjunto D de D' que reúna N documentos possivelmente relevantes à consulta 'q' realizada pelo usuário. O objetivo dessa primeira etapa é maximizar o recall do sistema de recuperação (ROCHA *et al.*, b). Ou seja, o sistema irá retornar a maioria dos documentos relevantes à consulta, mesmo que alguns documentos não relevantes sejam selecionados. Um algoritmo bastante utilizado para esse primeiro ranqueamento é o BM25 (ROCHA *et al.*, b) que, de forma geral, tenta recupera e ordenar os documentos em função do número de vezes que os termos da consulta ocorrem nos documentos.

O segundo ranqueamento tem como objetivo combinar os valores dos diversos atributos dos documentos, visando maximizar a precisão de relevância do ordenamento final dos documentos de D. Contudo, há relacionamentos complexos entre os atributos que dependem fortemente dos documentos de D, o que torna difícil projetar uma função direta ou determinística para o segundo ranqueamento que tenha boa precisão para distintas consultas. Com isso, na prática, prefere-se utilizar uma função de ranqueamento derivada automaticamente por meio de alguma técnica de aprendizado de máquina supervisionado. Uma vez treinado o modelo, na fase de teste o sistema de ranqueamento é capaz de, dado um conjunto de documentos D relacionados a uma consulta q, criar um ranqueamento (permutação) dos documentos.

Há diferentes abordagens para construir a função f (q, D). Essas abordagens se diferenciam pela forma de relacionar os documentos de uma mesma consulta durante o treinamento dos modelos. Os detalhes dessas abordagens serão discutidos a seguir.

2.7 Abordagens de *Leaning to Rank*

Segundo descreve Tie-Yan Liu (LIU, 2009) algoritmos de aprendizagem de máquina possuem quatro componentes gerais, sendo eles :

1. Representação do Espaço de entrada: forma de representação dos objetos, geralmente dado por vetores de atributos;
2. Espaço de saída: formado pelas pontuações alvo da aprendizagem em relação aos objetos do espaço de entrada;
3. Hipótese: função de mapeamento do espaço de entrada para o espaço de saída,sendo o objetivo da aprendizagem. Assim, a hipótese corresponde à função f (q, D) e utiliza o vetor de atributos dos documentos gerando uma predição de acordo com o espaço de saída;
4. Função de perda: modo de avaliação da capacidade de generalização da hipótese gerada pelo treinamento

Através desses quatro componentes é possível classificar a abordagem dos algoritmos de LeToR em três categorias distintas que descrevem diferentes perspectivas com que esses algoritmos podem trabalhar os espaços de entrada e saída, assim como a definição da hipótese e a escolha da função de perda para avaliação do modelo.

Abordagem *Pointwise:* A abordagem Pointwise aplica as técnicas de aprendizado de máquina diretamente para resolver problemas de ranking (LIU, 2009). O espaço de entrada é constituído por um documento representado por um vetor de atributos, enquanto o espaço de saída é o grau de relevância do documento. O vetor de atributos pondera a relação do documento com a consulta, assim, muitas vezes é descrito como vetor par documento-consulta. Essa abordagem trata LeToR como um problema de classificação ou regressão. Assim, o modelo treinado tenta predizer a relevância de um documento dado seus atributos. Aplicando o modelo em diferentes documentos, obtêm-se a relevância individual deles e assim podemos gerar uma ordenação final

Abordagem *Pairwise:* A abordagem pairwise diferente da pointwise não foca em obter o grau de relevância de cada documento. Nessa abordagem busca-se obter o *ranking* reduzindo a um problema de classificação em pares de documentos. O objetivo nesse caso é classificar os pares de documentos predizendo qual dos dois é o mais importante dada a consulta e, minimizar o número de pares de documentos classificados de modo errôneo. Desta forma, se todos os pares de documento forem classificados corretamente, temos a solução ótima para o ranking (LIU, 2009). Como os pares de documentos são dependentes uns dos

outros, não conseguimos aplicar algoritmos de classificação diretamente para solucionar o problema. A função de perda utilizada nessa abordagem considera somente a ordem relativa entre dois documentos e o objetivo principal é minimizar os pares de documentos não classificados corretamente.

Abordagem *Listwise:* A abordagem listwise examina todos os documentos em busca da ordenação ideal. Diferente da abordagem pointwise que analisa a relevância de cada documento perante a consulta e da abordagem pairwise que analisa a relevância relativa entre dois documentos dado a consulta, essa abordagem analisa a relevância de todo o conjunto de documentos para encontrar o ranking. O espaço de entrada é dado pelo conjunto de documentos D relacionado à consulta q sendo que cada documento em D é representado pelo seu vetor de atributos, enquanto o espaço de saída é definido por uma lista já ranqueada dos documentos.

2.8 Transformers

A arquitetura de *transformers* tornou-se rapidamente a arquitetura dominante para processamento de linguagem natural, superando modelos neurais alternativos, como redes neurais convolucionais e recorrentes em desempenho para tarefas de compreensão de linguagem natural e geração de linguagem natural. A arquitetura é dimensionada com dados de treinamento e tamanho do modelo, facilita o treinamento paralelo eficiente e captura recursos de sequência de longo alcance. Modelo de pré-treinamento permite que os modelos sejam treinados em corpora genéricos e posteriormente facilmente adaptados a tarefas específicas com alto desempenho. A arquitetura do Transformer é particularmente favorável ao pré-treinamento em grandes corpora de texto, levando a grandes ganhos de precisão em tarefas posteriores, incluindo classificação de texto, compreensão da linguagem.

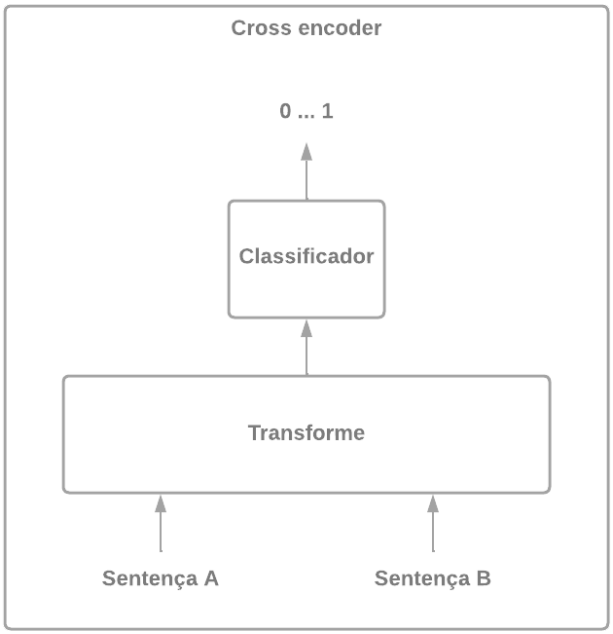
Esse avanço leva a uma ampla gama de desafios práticos que devem ser enfrentados para que esses modelos sejam amplamente utilizados. O uso onipresente do *Transformer* exige sistemas para treinar, analisar, dimensionar e aumentar o modelo em uma variedade de plataformas. A arquitetura é usada como um bloco de construção para projetar extensões cada vez mais sofisticadas e experimentos precisos. A adoção generalizada de métodos de pré-treinamento levou à necessidade de distribuir, ajustar, implantar e compactar os principais modelos pré-treinados usados pela comunidade.

2.9 Cross Encoders

Cross encoder (codificador cruzado) é um tipo de arquitetura de rede neural usada em tarefas de processamento de linguagem natural (NLP), como similaridade de sentença ou classificação de texto. Ao contrário de outros modelos NLP que usam um codificador para mapear sequências de entrada em um vetor de tamanho fixo, um codificador cruzado considera pares de sequências como entrada e produz uma única pontuação ou rótulo de saída.

O modelo de codificador cruzado pega duas sequências de entrada e as codifica separadamente usando o mesmo conjunto de parâmetros e, em seguida, combina as duas codificações em uma representação conjunta. Essa representação conjunta é então alimentada em um classificador ou regressor para produzir uma previsão ou pontuação, assim como mostrado na figura abaixo.

Figura 4 – Exemplo da aquitetura de cross encoder



Fonte: Elaborado pelo próprio autor)

Os codificadores cruzados são particularmente úteis em tarefas que exigem a comparação de duas sequências, como resposta a perguntas, detecção de paráfrase ou análise de sentimentos, pois podem capturar as interações e dependências entre as duas sequências. Eles demonstraram alcançar desempenho de ponta em uma variedade de benchmarks e aplicativos de PNL.

2.10 BERT

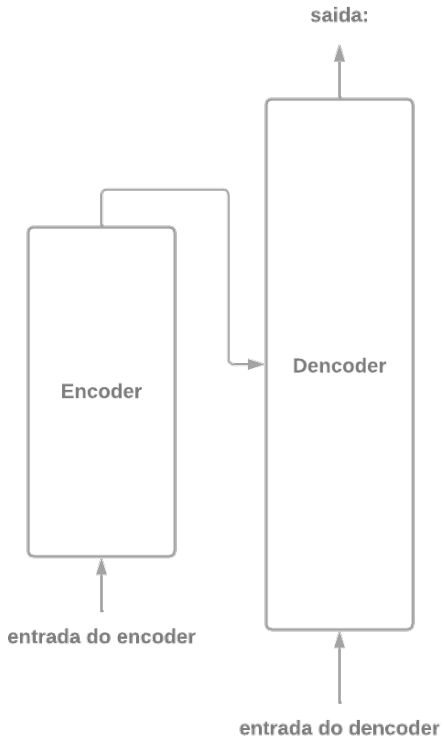
As tarefas de processamento de linguagem natural incluem uma ampla gama de aplicativos, desde bots de conversação e tradução automática até assistentes de voz e tradução de fala online. Nos últimos anos, essa indústria experimentou um rápido crescimento, tanto quantitativamente, no volume de aplicações e produtos do mercado, quanto qualitativamente, na eficácia dos modelos mais recentes e na proximidade com o nível humano de compreensão da linguagem.

No final de 2018, um grupo de cientistas do laboratório Google AI Language sob a liderança de J. Devlin apresentou um novo modelo linguístico chamado BERT (DEVLIN *et al.*, 2019). Este modelo destina-se ao aprendizado preliminar profundo da representação de texto bidirecional para uso subsequente em modelos de aprendizado de máquina. A vantagem deste modelo é sua facilidade de uso, que envolve adicionar apenas uma camada de saída à arquitetura neural existente para obter modelos de texto que superam a imprecisão de todos os existentes em diversos problemas de processamento natural de texto.

A arquitetura BERT é baseada no transformador bidirecional multicamadas descrito em 2017 por A. Washwani (DEVLIN *et al.*, 2019). Os autores treinaram duas versões da rede neural - uma padrão com 12 camadas e 768 coordenadas na exibição (110 milhões de parâmetros treinados no total) e uma grande com 24 camadas e 1024 coordenadas (340 milhões de parâmetros). O BERT usa incorporações de texto para representar uma sequência de entrada. Uma sequência é um conjunto arbitrário de tokens de texto contíguos. Por exemplo, o modelo usa as mesmas representações para uma frase e para um par, o que permite que o BERT seja usado para uma ampla gama de tarefas. Usando o processo descrito de treinamento adicional para uma tarefa específica, o BERT foi testado em vários conjuntos de dados padrão para comparar seu desempenho com outros modelos publicados. Então, no teste GLUE (Avaliação de Compreensão Geral da Linguagem, um conjunto de tarefas e conjuntos de dados que testam a compreensão da linguagem natural), o modelo baseado em BERT mostrou uma superioridade média de 4,5% e 7% (para redes neurais padrão e grandes, respectivamente) em comparação com os modelos mais conhecidos.

A arquitetura do transformador utiliza o modelo de sequência, que possui um codificador e um decodificador. Para incorporar a entrada, usamos o codificador e, para transformar a saída incorporada em uma string, podemos usar um decodificador. É semelhante a qualquer algoritmo de codificação-decodificação.

Figura 5 – Exemplo da aquitetura de *transformer* BERT



Fonte: Elaborado pelo próprio autor)

A arquitetura BERT possui uma estrutura diferente de um transformador tradicional. Dependendo do caso de uso, o modelo empilhará codificadores uns sobre os outros. Os embeddings usados na entrada serão alterados e enviados para um novo classificador, que ocorrerá de maneira específica da tarefa.

2.11 Métricas de Avaliação em LeToR

Em problemas tradicionais de aprendizagem de máquina (classificação e regressão), a avaliação de resultados é feita verificando se a predição dada para uma instância está correta, ou seja, comparando a predição para a instância com o seu valor de relevância real. Por outro lado, em LeToR, a avaliação deve qualificar a ordenação de documentos produzida. Isto é, indicar o quanto a ordenação gerada pelo modelo se aproxima da melhor ordenação possível do mesmo conjunto de documentos. Assim, LeToR demanda métricas específicas para avaliar qualidade de ranqueamentos.

Há diversas métricas para avaliar o desempenho de um modelo para LeToR. Nessa

seção é descrita a métrica NDCG (Normalized Discounted Cumulative Gain - Ganho Cumulativo Descontado Normalizado).

2.12 NDCG

Os Autores de (ROCHA *et al.*, a) argumentam que documentos altamente relevantes são mais valiosos do que documentos marginalmente relevantes.Assim, eles definem a métrica de ganho acumulado (CG) para produzir um vetor de ganho. O ganho cumulativo é a soma dessas pontuações de relevância e pode ser calculado como:

$$CG = \sum_{i=1}^{n} (relevancia)_i \tag{2.1}$$

Com base na lista classificada recuperada, em que cada documento é representado por sua pontuação de relevância (possivelmente ponderada) até uma posição classificada n definida para o experimento. Os autores argumentam que quanto maior a posição ranqueada de um documento relevante, menos valioso ele é para o usuário, pois é menos provável que o pesquisador examine o documento.

Devido ao tempo, esforço e informações acumuladas de documentos já vistos. Isso leva a “corrigir” as leituras fornecidas pelo ganho acumulado por um fator de desconto baseado em classificação, o logaritmo da classificação de cada documento. O ganho acumulado normalizado é calculado como a parcela do desempenho ideal que uma técnica de IR alcança. O ganho cumulativo descontado pode ser calculado pela fórmula:

$$DCG = \sum_{i=1}^{n} \frac{(relevanciai)}{log_2(i+1)} \tag{2.2}$$

Chegamos ao NDCG utilizando G-1 max,i (k) para realizar uma normalização do DCG. Gmax,i (k) é o fator de normalização definido de acordo com o ranqueamento baseado na ordenação pelos valores de relevância real, onde o NDCG para a posição k é 1 (LI, 2011). Logo, chegamos à seguinte função para o NDCG

$$NDCG(k) = G^{-1} max, i(k) * \sum_{j:\pi i \leq k} \frac{2^{yi,j} - 1}{\log 2(1 + \pi i(j))} \tag{2.3}$$

A NDCG avalia a qualidade do ranqueamento πi em uma determinada posição k, onde πi (j) é a posição do documento di,j em i, é yi,j é o valor de referência para a relevância do documento di,j. Para efeito interpretativo, NDCG varia no intervalo [0, 1] e indica, assim como o DCG, valores mais altos quando documentos relevantes estão mais ao topo da permutação. Portanto, para ranqueamentos baseados na ordenação pelos valores de relevância real, NDCG atribui 1 à todas as posições.

3 TRABALHOS RELACIONADOS

Nesta seção, serão apresentados alguns trabalhos relacionados com o projeto proposto neste trabalho.

3.1 Análise da Técnica Deep Forest para o Problema de Aprendizado de Ranqueamento

Em (ROCHA *et al.*, b) é apresentado uma análise de técnicas de deep learning Deep Forest para o problema de aprendizado de ranqueamento, com o objetivo principal de responder a seguinte pergunta "O aprendizado profundo com módulos não diferenciáveis, seguindo a estratégia Deep Forest, é capaz de melhorar a efetividade nas tarefas de LeToR?". Onde o autor configura alguns hiperparâmetros, e algoritmos para testar sua hipótese, sendo eles o Deep Forest, LambdaMART, e redes neurais.

Como parte da metodologia o autor utiliza a abordagem pointwise com procedimentos de ensemble que é uma estratégia de aprendizado em que é aplicado uma combinação de múltiplos algoritmos de aprendizagem de máquina, denominados base learners em uma estrutura única, dessa forma cada base learner produz um resultado preditivo.

Para isso foram utilizadas três coleções de dados de LeToR para o experimento: WEB10K , YAHOO! e MQ2007. Cada coleção foi dividida em cinco grupos de mesmo tamanho separados em conjuntos de treino, validação e teste, seguindo a proporção 60%, 20% e 20%, respectivamente. Os conjuntos de validação foram utilizados para verificação de hiperparâmetros, enquanto os conjuntos de teste foram utilizados para avaliação dos modelos, onde também foi aplicado t-teste pareado e os valores indicados por * apresentam significância estatística com 95% de confiança. Na tabela 1 encontram-se os detalhes referentes à quantidade de atributos, documentos e consultas das coleções.

Tabela 1 – Quantidade de atributos, consultas e documentos das coleções utilizadas.

	Atributos	Consultas	Documentos
WEB10K	136	10.000	1.200.192
YAHOO!	700	6.295	171.988
MQ2007	46	1.700	69.623

Como resultados (ROCHA *et al.*, b) traz a comparação entre Random Forest e Deep Forest, verificando a NDCG nos datasets utilizados.

Tabela 2 – Avaliação comparativa entre Random Forest e Deep Forest.

	WEB10K	**Yahoo!**	**MQ2007**
Random Forest	0,4417	0,7140	0,4132
Deep Forest	**0,4522**	**0,7241**	**0,4376**

Concluindo que, o Deep Forest é capaz de superar Random Forest, inclusive mantendo uma diferença. Isso mostra a eficácia e justifica o uso do método ensemble aplicado com Deep Forest no contexto de Learning to Rank.

Outro resultado significativo para este trabalho, que o autor relata, são os resultados de efetividade ao utilizar o LambdaMART como base learner do Deep Forest. A tabela apresenta os resultados obtidos para os três modelos envolvidos: Deep Forest, Deep Forest com LambdaMART como base learner, e o LambdaMART.

Tabela 3 – Avaliação comparativa do Deep Forest, Deep Forest-LambdaMART e LambdaMART

	WEB10K	**Yahoo!**	**MQ2007**
Deep Forest	0,4522	0,7241	0,4376
LambdaMART	0,4578	0,7188	0,4372
Deep Forest - LambdaMART	**0,4655**	**0,7286**	**0,4500**

Como pode ser observado, o LambdaMART, como um algoritmo do estado da-arte de LeToR, apresenta melhores resultados que o Deep Forest. Porém, quando utilizado como base learner do Deep Forest, visto que é um modelo ensemble baseado em árvores.

Por fim (ROCHA *et al.*, b) mostra que o Deep Forest não supera redes neurais profundas na tarefa de LeToR. Em seus resultados o Deep Forest apresenta valores superados estatisticamente pela Rede Neural Profunda em até 3,98%. Por sua vez, DF-LM alcança NDCG mais próximo ao da Rede Neural Profunda -1% menor na WEB10k e 1,34% menor na Yahoo!, aproximadamente – chegando a um NDCG maior na coleção MQ2007, porém sem relevância estatística.

Tabela 4 – Avaliação comparativa entre Deep Forest, Deep Forest-LambdaMART e Rede Neural Profunda

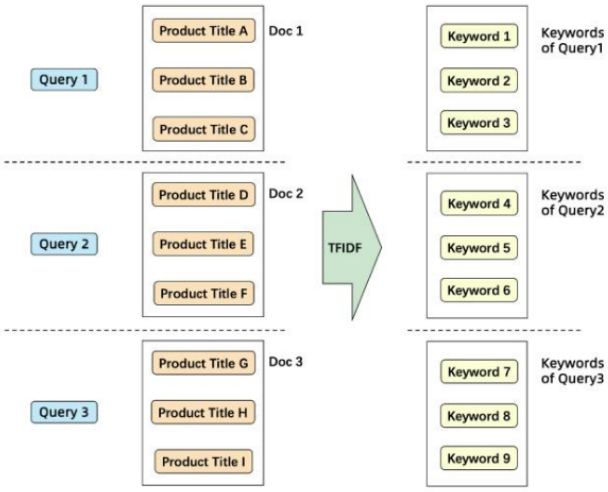
	WEB10K	**Yahoo!**	**MQ2007**
Deep Forest	0,4522	0,7241	0,4376
Deep Forest - LambdaMART	0,4655	0,7286	**0,4500**
Rede Neural Profunda	**0,4702**	**0,7384**	0,4490

3.2 A Winning Solution of KDD CUP 2022 ESCI Challenge for Improving Product Search

Neste relatório técnico (LIN *et al.*,), é apresentada uma solução premiada para o desafio. Os autores perceberam que consultas curtas possuem um grande impacto na precisão da classificação, usando um procedimento de TF-IDF, que é uma medida estatística que tem o intuito de indicar a importância de uma palavra de um documento em relação a uma coleção de documentos. Os autores utilizaram o TF-IDF baseado em consulta para extrair palavras-chave de títulos de produtos, marcadores de produtos e descrições de produtos. Em seguida, usam essas palavras-chave como recursos de consulta. Essa técnica de aumento de consulta pode melhorar muito a classificação e a precisão da classificação. Também usaram auto destilação, pós-processamento, ensemble e algumas outras tecnologias.E por fim, essa solução, da equipe "day-day-up" conquistou o 1º lugar na tarefa 2 e tarefa 3, e o 3º lugar na tarefa 1, entre 1699 participantes.

A figura 3 explica melhor a metodologia usada para extração das palavras-chave do texto dos produtos utilizando o TF-IDF, os autores pegam os títulos de todos os produtos correspondente à consulta como um documento e, em seguida, use TF-IDF para extrair palavras-chave. Também obtem as palavras-chave de productBulletPoint e productDescription para cada consulta. Desta forma, o extrato das palavras-chave podem ser usadas como o recurso de consulta. E pode ser colocado como texto de entrada.

Figura 6 – Procedimento TF-IDF baseado em consulta



Fonte: (LIN *et al.*,)

Os autores escolheram o algoritmo InfoXLM large como modelo principal, pois os

conjuntos de dados são multilíngues e o InfoXLM-large é um modelo pré-treinado com excelente compreensão entre idiomas. Como os dados de todas as três tarefas possuem rótulos com a mesma definição, a equipe concatenou o conjunto de treinamento de todas as três tarefas como um novo conjunto de treinamento. A equipe implementou o modelo com a biblioteca PyTorch e transformadores huggingface. A Tarefa 1 (Ranking): Ordena os produtos pela fórmula abaixo.

$$score = Pexact + 0.1 * Psubstitute + 0.01 * Pcompletion \tag{3.1}$$

Por fim a equipe conseguiu atingir uma métrica de NDCG igual a 0,9035 no dataset privado e 0,9056 no dataset publico, a seguir e mostrado a classificação geral da equipe.

Tabela 5 – Pontuação dos 4 primeiros da tarefa 1.

Rank	**Modelo**	**NDCG (Privado**	**NDCG (Público**
1	www	0,9043	0,9057
2	qinpersevere	0,9036	0,9047
3	day-day-up	**0,9035**	**0,9056**
4	GraphMIRAcles	0,9028	0,9036

3.3 A Semantic Alignment System for Multilingual Query-Product Retrieval

Neste artigo é descrito a solução vencedora pela equipe “www” (ZHANG *et al.*,) ao Amazon ESCI Challenge do KDD CUP 2022, que alcançou uma pontuação NDCG de 0,9043 e ganhou o primeiro lugar na tarefa 1. Os autores focaram principalmente na tarefa 1 e propuseram um sistema de alinhamento semântico para recuperação multilíngue de produtos de consulta. Modelos de linguagem multilíngue (LM) pré-treinados são adotados para obter a representação semântica de consultas e produtos. Os modelos desenvolvidos pela equipe são todos treinados com perda de entropia cruzada para classificar os pares de consulta-produto nas categorias ESCI 4 primeiro e, em seguida, usaram a soma ponderada com as probabilidades de 4 classes para obter a pontuação para classificação. Para impulsionar ainda mais o modelo, também fizeram pré-processamento de dados, aumento de dados por tradução, especialmente manipulação de textos em inglês com LMs em inglês, treinamento de adversários com AWP e FGM, auto destilação, pseudo-rotulagem, suavização de rótulos e ensemble.,A solução desenvolvida supera outras tanto no ranking público quanto no privado.

3.4 Análise Comparativa

Assim como realizado em (LIN *et al.*,) e (ZHANG *et al.*,) este trabalho pretende realizar o ranqueamento de produtos, utilizando técnicas de LeToR. Pretende-se utilizar algumas soluções semelhantes às usadas pelos principais participantes do desafio, além de utilizar técnicas de busca por similaridade, com ferramentas específicas para isso, como as bibliotecas Faiss, e Annoy.

Além disso a base de dados utilizada neste trabalho se difere da que foi utilizada por (LIN *et al.*,) e (ZHANG *et al.*,) visto que ambos utilizam o data set (Shopping Queries Data Set) multilingue, já este trabalho utilizará apenas as linhas referentes ao idioma Inglês. Outra mudança feita em comparação aos outros trabalhos, e a análise da métrica ERR, pois deseja-se ver o tempo recíproco esperado que o usuário levará para encontrar um documento

Na Tabela asseguir, é apresentado um resumo das características encontradas nos trabalhos relacionados descritos anteriormente, e a comparação com o projeto apresentado neste trabalho.

Tabela 6 – Tabela comparativa.

Trabalho	Algoritmos	Data set	Problema	Métricas
(ROCHA *et al.*, b)	Deep Forest Random Forest Redes Neurais Profundas Deep Forest (LambdaMART)	WEB10K YAHOO! MQ2007	Ranqueamento de documentos	NDCG
(LIN *et al.*,)	Info XML	Shopping Queries Data Set (Completo)	Ranqueamento de produtos	NDCG
(ZHANG *et al.*,)	Info XML DeBERTa RemBERT.	Shopping Queries Data Set (Completo)	Ranqueamento de produtos	NDCG
Este Trabalho	MS Marco DeBERTa v3-large LambdaMART TF-IDF + LambdaMART	Shopping Queries Data Set (busca em inglês)	Ranqueamento de produtos	NDCG

4 PROCEDIMENTOS METODOLÓGICOS

Para alcançar o primeiro objetivo proposto neste trabalho, se faz necessário um conjunto de etapas a serem seguidas, conforme pode ser observado no fluxograma apresentado na Figura abaixo

Figura 7 – Fluxograma de procedimentos metodológicos

4.1 Análise Exploratória dos Dados

A primeira etapa para realizar o desenvolvimento deste trabalho consiste em salvar em um repositório local a base de dados utilizado na competição (citar) KDD cup, e realizar uma análise exploratória dos dados, com o objetivo de descobrir padrões, ou valhas nos dados a fim de serem solucionados na próxima etapa.

4.2 Pré-processamento dos dados

Como segunda etapa do processo, tem-se o pré-processamento dos dados com o intuito de garantir que os dados disponibilizados aos algoritmos são completos e consistentes, de forma a obter o melhor desempenho e os melhores resultados.

Por norma, a etapa de pré-processamento incorpora as seguintes fases (Sivakumar e Gunasundari 2017):

1. Limpeza dos dados
2. Integração dos dados
3. Transformação dos dados
4. Redução dos dados

Dessa forma, neste trabalho será incorporado na primeira fase de pré-processamento a realização da remoção de *stop words*, *low case* e caracteres especiais, nas colunas do tipo string, e a redução dos dados faltantes.

4.3 Critérios de Escolha e Seleção dos Modelos

Os modelos escolhidos serão selecionadas com base nos critérios abaixo:

- Modelos referenciados no estado da arte
- Modelos citados como trabalhos futuros dos artigos da competição
- Modelos não citados em artigos da competição
- Modelos com desempenho satisfatório em relação a métrica NDCG com dados públicos

4.4 Treinamento dos modelos

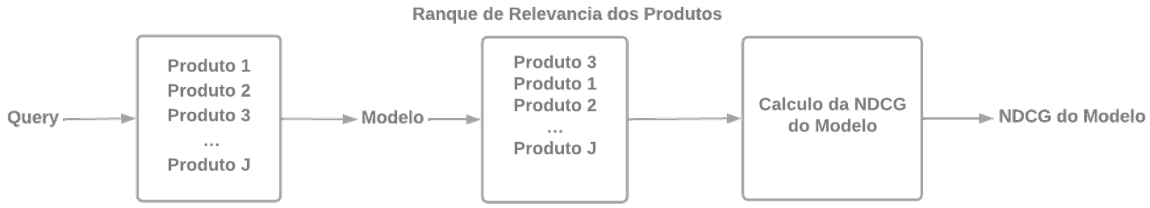
Na terceira etapa, levando em consideração o que foi apresentado no Capítulo 2, o presente trabalho utilizará modelos de aprendizado, especificamente os mostrados na Tabela 6, que podem ser encontrados na biblioteca scikit-learn e PyTorch, que é são bibliotecas da linguagem Python desenvolvidas especificamente para aplicações e práticas de machine learning.

A própria competição disponibilizou os dados separados em treino e teste, sendo separados da seguinte maneira, o conjunto de dados de treinamento contém uma lista de pares de resultados de consulta com rótulos E/S/C/I anotados. Os dados são multilíngues e incluem consultas dos idiomas inglês , japonês e espanhol, porém neste trabalho será utilizado apenas os dados no idioma inglês. Os exemplos no conjunto de dados têm os seguintes campos: exampleId, query, queryId, product, productTitle, productDescription, productBulletPoint, productBrand, productColor, productLocalee, esciLabel. O conjunto de dados de teste foi estruturado de forma semelhante, exceto que o último campo (esciLabel) que foi retido.

4.5 Avaliação dos modelos

Por fim, os dados de testes serão submetidos aos modelos, com objetivo de verificar seu comportamento em relação a ndcg, e comparar o desempenho entre si. A figura 8 mostra um exemplo de como será o fluxo de entrada e saída dos modelos, sendo a entrada referente ao conjunto de dados de teste, ou seja uma lista de consultas com seus respectivos produtos, e a saída o valor da ndcg do modelo.

Figura 8 – Fluxo de entrada e saida dos modelos



4.6 Desafios para deploy dos modelos

Como citado no segundo objetivo deste trabalho, essa etapa irá descrever quais os desafios e soluções para implementação de um dos modelos criados em um ambiente de produção real, para isso será imaginado o ambiente de vendas da Americas SA, que foi parceira deste trabalho. Assim como a Amazon, que foi a fornecedora dos dados da competição, a Americanas possui o desafio de ranquear os produtos entregues em seus aplicativos de vendas online.

5 EXPERIMENTOS E RESULTADOS

Neste capítulo são relatados os experimentos realizados para analisar a métrica dos modelos propostos.

5.1 Análise Exploratória

O Shopping Queries Data Set é um conjunto de dados anotado manualmente em grande escala composto por consultas desafiadoras de clientes. Para primeira tarefa da competição foi divulgado três conjuntos de dados sendo eles, um dataset de treinamento, um dataset de teste, e um dataset com o catálogo dos produtos. O conjunto de dados de treinamento contém uma lista de pares de resultados de consulta com rótulos E/S/C/I anotados. Os dados são multilíngues e incluem consultas nos idiomas inglês , japonês e espanhol, porém este trabalho só utilizou as amostras no idioma inglês. Os exemplos no conjunto de dados têm os seguintes campos: example_id, query, query_id, product_id, product_locale, esci_label, small_version, large_version, split, product_title, product_description, product_bullet_point, product_brand, product_color e source

Em cada conjunto de dados fornecido pela competição, foi realizada uma análise exploratória dos dados a fim de identificar a quantidade de linhas referente ao idioma inglês, e os tipos de dados referentes às colunas de cada dataset. Além disso, foi verificada a quantidade de dados faltantes, que se somaram 100 linhas referentes a coluna “product title” essas linhas foram excluídas do conjunto de treinamento. A seguir e mostrado o resumo do conjunto de dados de consultas do dataset para a tarefa 1.

Tabela 7 – Resumo do conjunto de dados de consultas .

Total		
Consultas	Quantidade de Produtos	Média de Profundidade
29,844	601,354	20.15
Treinamento		
20,888	419,653	20.09
Teste		
8,956	181,701	20.29

5.2 Pré-processamento dos Dados

Como o dataset possui majoritariamente colunas referentes a texto, foi utilizado técnicas de limpeza como remoção de *stop words* da língua inglesa que são palavras que podem ser consideradas irrelevantes para o conjunto de resultados. Também foi feita uma remoção de caracteres especiais como símbolos de pontuação, números, e emoticons. Além disso foi padronizado todos os textos com as palavras em minúsculas. A seguir, é mostrado um exemplo de consulta antes e depois das técnicas de processamento aplicadas.

Figura 9 – Exemplo de consulta original e pré-processada (Fonte: Elaborado pelo próprio autor)

5.3 Modelos e Treinamento

Neste capítulo são relatados os experimentos realizados para analisar a métrica dos modelos propostos. Foram escolhidos quatro modelos, sendo eles descritos a seguir

MS Marco: Este modelo foi escolhido pois se trata de um modelo de cross encoder que possui um corpus de recuperação de informações em larga escala que foi criado com base em consultas de pesquisa de usuários reais usando o mecanismo de pesquisa Bing. Os modelos fornecidos podem ser usados para pesquisa semântica, ou seja, dadas palavras-chave, ou uma frase de pesquisa, ou até mesmo uma pergunta, o modelo encontrará passagens relevantes para a consulta de pesquisa. Os dados de treinamento do MS Marco consistem em mais de 500 mil exemplos, enquanto o corpus completo consiste em mais de 8,8 milhões de passagens. Além disso, não foi encontrado nenhum registro de uso do MS Marco nos artigos da competição.

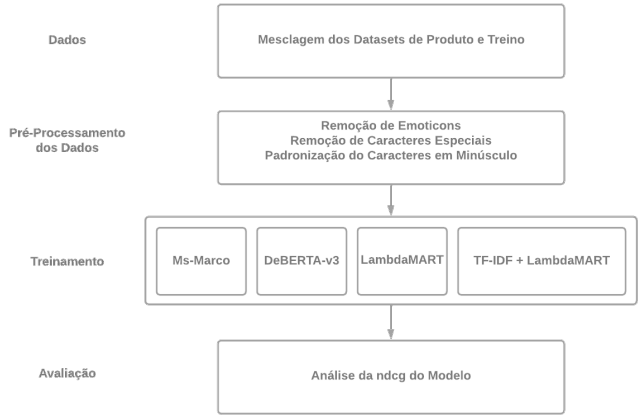
DeBERTa-v3: Este modelo de cross encoder foi escolhido pois na maioria dos artigos relacionado a competição havia referências sobre tal modelo. Além disso o DeBERTa e um modelo de BERT melhorado, pois ao contrário do BERT onde cada palavra na camada de entrada é representada por um vetor que é a soma de sua incorporação de palavra e incorporação de

posição, cada palavra no DeBERTa é representada por dois vetores que codificam seu conteúdo e posição, respectivamente, e os pesos de atenção entre as palavras são calculadas usando matrizes desemaranhadas com base em seus conteúdos e posições relativas, respectivamente, tornando-se assim um modelo apto a experimentar.

LambdaMART: Este modelo foi escolhido pois é um algoritmo clássico de ranqueamento pairwise. O LambdaMART é uma combinação de LambdaRank e MART (*Multiple Additive Regression Trees*). O MART usa árvores de decisão com aumento de gradiente para tarefas de ranqueamento. No entanto, o LambdaMART melhora isso usando árvores de decisão com aumento de gradiente com uma função de custo derivada do LambdaRank para ordenar qualquer situação de classificação. Além disso, não foi encontrado nenhum registro de uso do LambdaMART nos artigos da competição.

TF-IDF + LambdaMART: Essa combinação de técnicas foi escolhida com base no trabalho futuro citado por (ZHANG *et al.*,) que em seu artigo relata o desejo de unir as duas técnicas, sendo elas o TF-IDF para extrair as palavras chaves das consultas e o algoritmo LambdaMART para ranquear.
A figura abaixo resume as etapas e procedimentos realizados em cada modelo.

Figura 10 – Pipeline geral de treinamento dos modelos (Fonte: Elaborado pelo próprio autor)



5.3.1 Experimento 01: MS Marco

O primeiro modelo treinado foi um modelo de *cross encoder* utilizando a abordagem pairwise chamado ms-marco-MiniLM-L-12-v2 disponibilizado no Hugging Face. Por ser

um modelo de cross encoder o ms-marco-MiniLM-L-12-v2 recebe como entrada além dos parâmetros as sequências de consultas, e o próprio algoritmo se responsabiliza pela separação das sentenças. Esse modelo pode ser usado para recuperação de informações, já que dada uma consulta, o modelo codifica essa consulta com todas as passagens possíveis, em seguida, classifica as passagens em ordem decrescente. O modelo ms-marco-MiniLM-L-12-v2 funciona usando a arquitetura de transformer, que é um modelo de linguagem baseado em redes neurais. Ele segue um processo de pré-treinamento e ajuste fino para aprender a compreender e gerar respostas para consultas. Durante o ajuste fino, o modelo é treinado usando uma técnica chamada "supervisão por classificação". Ele recebe uma consulta e um produto e é treinado para gerar uma resposta que seja adequada para aquela consulta específica. Como parâmetros para esse modelo foi utilizado os seguintes parâmetros.

- num_epochs = 1
- num_labels = 1
- max_length = 512
- loss_fct = MSELoss
- optimizer_params='lr': lr
- evaluation_steps = 5000
- warmup_steps = 5000

O modelo obteve uma ndcg igual a 0.8300, um resultado muito inferior se comparado com outros modelos de *cross encoder* como o DeBERTa-v3, que foi utilizado na segunda experimentação deste trabalho.

5.3.2 Experimento 02: DeBERTa-v3

O segundo modelo treinado foi um modelo de cross encoder utilizando a abordagem pairwise chamado DeBERTa-v3. Por ser um modelo de *cross encoder* assim como o ms marco, o modelo DeBERTa-v3 recebe como entrada além dos parâmetros a sequência de consultas, e o próprio algoritmo se responsabiliza pela separação das sentenças. O DeBERTa melhora o BERT com dois novos componentes: DA (*Disentangled Attention*) e um decodificador de máscara aprimorado. Ao contrário das abordagens existentes que usam um único vetor para representar o conteúdo e a posição de cada palavra de entrada, o mecanismo DA usa dois vetores separados: um para o conteúdo e outro para a posição. Enquanto isso, os pesos de atenção do mecanismo DA entre as palavras são calculados por meio de matrizes desemaranhadas em

seus conteúdos e posições relativas. Assim como o BERT, o DeBERTa é pré-treinado usando modelagem de linguagem mascarada. O mecanismo DA já considera o conteúdo e as posições relativas das palavras do contexto, mas não as posições absolutas dessas palavras, que em muitos casos são cruciais para a previsão. Combinando mecanismo de atenção, incorporações contextuais, representações em camadas e considerando as informações em nível de token e em nível de sequência, o DeBERTa efetivamente recupera e processa as informações da entrada. Esse processo de recuperação permite que o modelo entenda as relações entre os tokens, capture o contexto e gere representações significativas que podem ser usadas para várias tarefas, incluindo ranqueamento. Como parâmetros para esse modelo foi utilizado os seguintes parâmetrLambdaMARTos.

- maxlength = 128
- batch_size = 64
- learning_rate = 3e-5
- load_accumulation = 8

O modelo obteve uma ndcg igual a 0.9016, um resultado superior ao primeiro modelo de cross encoder.

5.3.3 Experimento 03: LambdaMART

O terceiro experimeto criado foi um modelo utilizando o ranqueador LambdaMART utilizando a abordagem pairwise. O LambdaMART é a versão em árvore aprimorada do LambdaRank, baseada no RankNet. As árvores impulsionadas, especialmente o LambdaMART, provaram ser muito bem-sucedidas na resolução de problemas de aprendizado do mundo real para ranqueamento, porém não foi encontrado nenhuma referência de utilização desse modelo na competição, somente (LIN *et al.*,) que o cita como trabalho futuro. O LambdMART por meio do conjunto de árvores de decisão, treinamento com aumento de gradiente e agregação de previsões de árvore, o LambdMART efetivamente recupera informações dos recursos de entrada e aprende a ranquear pares de documentos de consulta com base em sua relevância. Ele aproveita o poder das árvores de decisão e da otimização baseada em gradiente para capturar padrões complexos e tomar decisões de ranqueamneto precisos. Com isso foi desenvolvido um modelo utilizando a biblioteca LightGbm que possui um método de ranqueamento com Lambdamart, utilizando os seguintes parâmetros.

- objective = "lambdarank"

- nestimators = 20
- maxdepth = -100
- numleaves = 100
- learningrate = 0.001

LambdaMART

O modelo obteve uma ndcg igual a 0.8027, o pior resultado comparado aos outros modelos

5.3.4 Experimento 04: LambdaMART + TF-IDF

Por fim foi treinado um modelo que foi pensado como trabalho futuro no artigo de (LIN *et al.*,), foi utilizada a técnica de TF-IDF junto com o ranqueador LambdaMART. Para o ranqueador foi utilizado os mesmos parâmetros do modelo 03. Com isso o modelo obteve uma ndcg de 0.9003

5.3.5 Análise dos Resultados

Ao analisar a métrica dos modelos treinados podemos ranquear de forma decrescente as soluções como mostrado abaixo:

Posição	**Modelo**
1	DeBERTa-v3
2	LambdaMART + TF-IDF
3	MS Marco
4	LambdaMART

Podemos observar com os resultados dos experimentos que o modelo DeBERTA-v3 obteve o melhor resultado se comparado aos outros, até mesmo o modelo utilizado no experimento 4 que foi proposto por um dos medalhistas da competição (referência) em seu artigo. Esse resultado do experimento com DeBERTA pode ser explicado pela construção deste modelo, O DeBERTa-v3 melhora os modelos BERT e RoBERTa usando atenção desemaranhada e decodificador de máscara aprimorado. Com essas duas melhorias, DeBERTa executa o modelo RoBERTa na maioria das tarefas de compreensão de linguagem natural com dados de treinamento de 80 GB.

Apesar do modelo proposto por (LIN *et al.*,) não ter sido superior às abordagens documentadas em outros artigos, a solução obteve um resultado superior ao outro modelo de

cross encoder utilizado no experimento 3. Isso comprova a eficiência do lambadaMART, em tarefas de ranqueamento,com a adição de outra técnicas como o TF-IDF ou o *deep forest* utilizado por (ROCHA *et al.*, b)

5.4 Principais Desafios

Essa seção relata quais os desafios e soluções para implementação de um dos modelos criados em um ambiente de produção real, para isso foi imaginado o ambiente de vendas da Americas SA, que foi parceira deste trabalho. Os desafios para implementação dos modelos são listados abaixo

1. Armazenamento/obtenção dos dados de forma segura.
2. Ambiente de desenvolvimento e implantação que suporte a alta carga de dados.

Para isso foi desenvolvido uma arquitetura baseada em nuvem, que solucionaria tais desafios como mostrado na t figura abaixo

5.4.1 Obtenção dos dados

O primeiro desafio a ser solucionado é a obtenção de dados de produtos e consultas de forma segura. Para isso foi pensado na utilização do bigquery que é o armazenamento de dados totalmente gerenciado e sem servidor do Google que permite análise escalonável em petabytes de dados. Além de ser uma plataforma como serviço que suporta consultas SQL. Dessa forma ficaria fácil e seguro criar conjuntos de dados iguais aos fornecidos pela Amazon na competição.

5.4.2 Ambiente de Desenvolvimento e Implatação

Para solucionar o desafio de um ambiente de desenvolvimento que suporte a alta carga de dados, serÁ utilizado a plataforma de inteligência artificial da americanas chamada IA Platform.

A IA platform possui várias ferramentas para o desenvolvimento de projetos de IA, a principal ferramenta dela e o jupytherhub, ambiente gerenciável que o desenvolvedor pode codificar seus projetos. Com essa plataforma o desenvolvimento pode selecionar 3 tipos de máquinas virtuais com CPU's ou GPU's, que dão suporte ao desenvolvimento de projetos em várias linguagens, como python, R, e notebooks Pythons. Além de facilitar a criação de pipelines

de machine learning com a ferramenta Elyra. Além dos 3 tipos de máquinas que o usuário pode escolher, através do elyra o usuário pode escalonar mais recursos para o pipeline, que será executado em outra ferramenta da plataforma, o KubeFlow pipelines que é uma ferramenta para criar e implantar fluxos de trabalho de aprendizado de máquina portáteis e escalonáveis com base em contêineres do Docker.

6 CONCLUSÕES E TRABALHOS FUTUROS

A proposta deste trabalho foi analisar as técnicas de aprendizagem de ranqueamento utilizando como base a competição realizada pela Amazon no ano de 2022 chamada amazon kdd cup 2022, para isso foi utilizado o conjunto de dados fornecido pela competição chamada *Shopping Queries Dataset*. Foi executado quatro modelos diferentes, e realizado a análise de seus desempenhos através da métrica de ganho acumulado normalizado, com isso foi observado que o modelo de cross encoder DeBERTA-v3 obteve o melhor resultado se comparado ao outros modelos desenvolvidos.

6.1 Trabalhos Futuros

Como trabalhos futuros pretende-se realizar os testes dos modelos citados neste trabalho com um conjunto de dados específico da empresa Americanas SA, para isso necessita-se a compilação de um dataset específico. Além disso planeja-se realizar novos testes com outros modelos como o uso da biblioteca Faiss que é uma biblioteca para busca de similaridade eficiente e de agrupamento de vetores densos. Contém algoritmos que buscam em conjuntos de vetores de qualquer tamanho, até aqueles que possivelmente não cabem na memória RAM. Outra biblioteca que pretende-se utilizada é o annoy biblioteca em python que procura pontos no espaço que estejam próximos a um determinado ponto de consulta. Ele também cria grandes estruturas de dados baseadas em arquivo de somente leitura que são mapeadas na memória para que muitos processos possam compartilhar os mesmos dados.

REFERÊNCIAS

CAMOSSI, G.; TEIXEIRA, H. D.; RODAS, C. M.; ALVES, R. C. V. Técnicas de search engine optimization (seo) aplicado para o comércio eletrônico. Disponivel em: =https://www.revistas.usp.br/incid/article/view/187500.Acesso em: 18 out, 2022.

CHOUDHARY, N.; RAO, N.; KATARIYA, S.; SUBBIAN, K.; REDDY, C. K. **ANTHEM**: Attentive hyperbolic entity model for product search. In: . [*S. l.*]: Disponivel em: =https://doi.org/10.1145/3488560.3498456.Acesso em 19 de out 2022.

CHOUDHARY, N.; RAO, N.; SUBBIAN, K.; REDDY, C. K. **Graph-based multilingual language model**: Leveraging product relations for search relevance. In: . [*S. l.*]: Disponivel em: = https://www.amazon.science/publications/graph-based-multilingual-language-model-leveraging-product-relations-for-search-relevance.Acesso em 19 de out 2022.

CONWAY, J. M. W. e D. **Machine Learning for Hackers**. [*S. l.*]: OReilly Media. Acesso em: 19 out. 2022, 2012.

CUP, A. K. **Shopping Queries Dataset: A Large-Scale ESCI Benchmark for Improving Product Search**. [*S. l.*]: Disponivel em: https://www.aicrowd.com/challenges/esci-challenge-for-improving-product-search Acesso em: 19 out. 2022.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. **ArXiv**, . Acesso em: 19 out. 2022, abs/1810.04805, 2019.

GABRIEL, R. K. e M. **Marketing na Era Digital - conceitos, plataformas e estratégias**. [*S. l.*]: Atlas, Acesso em 19 out 2022, 2020. v. 2.

GÉRON, A. **Aprendizado de Máquina com Scikit-Learn & TensorFlow**. [*S. l.*]: Starlin Alta Editora e Consultoria Eireli. Acesso em: 19 out. 2022, 2019.

HUB, I. C. L. Supervised learning [s.l: s.n]. Disponivel em: https://www.ibm.com/cloud/learn/supervised-learnin. Acesso em: 19 out. 2022, 2020.

KARATZOGLOU, A.; BALTRUNAS, L.; SHI, Y. Learning to rank for recommender systems. In: . [*S. l.*]: Disponivel em: = https://doi.org/10.1145/2507157.2508063.Acesso em 18 de out 2022, 2013.

LI, H. A short introduction to learning to rank. **IEICE Transactions on Information and Systems**, Acesso em: 19 out. 2022, E94.D, n. 10, p. 1854–1862, 2011.

LIN, J. L. J. *et al.* A winning solution of kdd cup 2022 esci challenge for improving product search. Disponivel em: =https://amazonkddcup.github.io/papers/3782.pdf. Acesso em: 19 out. 2022.

LIU, T.-Y. Learning to rank for information retrieval. **Found. Trends Inf. Retr**, Now Publishers Inc. Acesso em: 19 out. 2022, Hanover, MA, USA, v. 3, n. 3, 2009. Disponível em: https://doi.org/10.1561/1500000016.

PîRTOACă, G.-S.; REBEDEA, T.; RUSETI, S. **Answering questions by learning to rank – Learning to rank by answering questions**. [*S. l.*]: Disponivel em: =https://arxiv.org/abs/1909.00596.Acesso em: 18 de out 2022, 2019.

REZAEE, E. A survey on blockchain-based search engines. **Applied Sciences**, Disponivel em: =https://www.mdpi.com/2076-3417/11/15/7063. Acesso em: 19 out. 2022, v. 11, 2021.

REZAEE, E.; SAGHIRI, A. M.; FORESTIERO, A. A survey on blockchain-based search engines. **Applied Sciences**, Acesso em: 19 out. 2022, v. 11, 2021. Disponível em: https://www.mdpi.com/2076-3417/11/15/7063.

Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II. Berlin, Heidelberg: Springer-Verlag. Acesso em: 19 out. 2022. ISBN 978-3-030-99738-0.

ROCHA, L. E. C. *et al.* Análise da técnica deep forest para o problema de aprendizado de ranqueamento. Universidade Federal de Goiás. Acesso em: 19 out. 2022.

RUSSELL, P. N. S. **Inteligência Artificial**. [*S. l.*]: Elsevier Editora Ltda. Acesso em: 19 out. 2022, 2013.

SINGH, P. A novel architecture of ontology based semantic search engine. **International Journal of Science and Technology**, Acesso em: 19 out. 2022, v. 1, p. 650–654, 12 2012.

TROTMAN, A.; ZHANG, J. Future web growth and its consequences for web search architectures. **arXiv preprint arXiv:1307.1179**, Acesso em: 19 out. 2022, 2013.

ZHANG, Q. Z. Q. *et al.* A semantic alignment system for multilingual query-product retrieval. Disponivel em: =https://amazonkddcup.github.io/papers/9517.pdf. Acesso em: 19 out. 2022.