UNIVERSIDADE ESTADUAL DO OESTE DO PARANÁ - UNIOESTE

CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS - CCET

Programa de Pós-Graduação em Ciência da Computação - PPGComp

Aplicação de Tecnologia *Matching* para realizar a Correspondência entre os Usuários e Especialistas na Prestação de Serviços de Assessoria Científica

Dissertação (Mestrado)

Elielson Nogueira de Souza

Cascavel–PR

2025

UNIVERSIDADE ESTADUAL DO OESTE DO PARANÁ - UNIOESTE

CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS - CCET

Programa de Pós-Graduação em Ciência da Computação - PPGComp

Elielson Nogueira de Souza

Aplicação de Tecnologia *Matching* para realizar a Correspondência entre os Usuários e Especialistas na Prestação de Serviços de Assessoria Científica

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre em Ciência da Computação, do Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual do Oeste do Paraná - Campus de Cascavel.

Orientador(a): Dr. Fabio Alexandre Spanhol

Cascavel–PR

2025

Ficha de identificação da obra elaborada através do Formulário de Geração Automática do Sistema de Bibliotecas da Unioeste.

Nogueira de Souza, Elielson
Aplicação de Tecnologias Matching para realizar a Correspondência entre os Usuários e Especialistas na Prestação de Serviços de Assessoria Científica / Elielson Nogueira de Souza; orientador Fabio Alexandre Spanhol. -- Cascavel, 2025.
130 p.

Dissertação (Mestrado Acadêmico Campus de Cascavel) -- Universidade Estadual do Oeste do Paraná, Centro de Ciências Exatas e Tecnológicas, Programa de Pós-Graduação em Ciências da Computação, 2025.

1. Correspondência Semântica. 2. Processamento de Linguagem Natural. 3. Assessoria Científica. 4. Inteligência Artificial. I. Alexandre Spanhol, Fabio, orient. II. Título.

ELIELSON NOGUEIRA DE SOUZA

APLICAÇÃO DE TECNOLOGIA *MATCHING* PARA REALIZAR A CORRESPONDÊNCIA ENTRE OS USUÁRIOS E ESPECIALISTAS NA PRESTAÇÃO DE SERVIÇOS DE ASSESSORIA CIENTÍFICA

Monografia apresentada como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação, pela Universidade Estadual do Oeste do Paraná, Campus de Cascavel, aprovada pela Comissão formada pelos professores:

Prof. Dr. Fabio Alexandre Spanhol (Orientador)
Programa de Pós-Graduação em Ciência da Computação, UNIOESTE

Prof. Dr. André Luiz Brun
Programa de Pós-Graduação em Ciência da Computação, UNIOESTE

Prof. Dr. Sidgley Camargo de Andrade
Programa de Pós-Graduação em Ciência da Computação, UNIOESTE

Prof. Dr. Thiago Henrique Pereira da Silva
Ciência da Computação, IFMG

Cascavel, 15 de agosto de 2025

Este trabalho é dedicado às crianças adultas que,
quando pequenas, sonharam em se tornar cientistas.

Agradecimentos

Agradeço primeiramente a Deus e a Jesus Cristo, pela força, fé, sabedoria e saúde concedidas ao longo desta jornada, que me permitiram superar desafios e chegar até aqui.

À minha família, em especial à minha mãe, *Leni Maria Nogueira de Souza (in memoriam)* e ao meu pai, *Edmilson Nogueira de Souza*, pelos ensinamentos, valores, fé e força transmitidos, que me sustentaram nos momentos difíceis e me impulsionaram a seguir em frente com coragem e propósito. À minha esposa e filhas *Allice* e *Guadhalupy*, por todo o amor, compreensão e apoio incondicional ao longo dessa caminhada.

Ao meu irmão, *Rogerio*, e aos amigos que me incentivaram e acreditaram no meu potencial, mesmo nos momentos mais difíceis.

Agradeço com carinho a todos os professores do Programa de Pós-Graduação em Ciência da Computação da UNIOESTE, pela dedicação, paciência e pelo conhecimento compartilhado durante toda a minha formação acadêmica. Em especial, agradeço ao meu orientador, *Dr. Fabio Alexandre Spanhol*, pelas orientações atenciosas, apoio técnico e incentivo constante ao longo desta jornada.

À empresa SciBees, pela oportunidade de aplicar meu conhecimento em um projeto real e desafiador, contribuindo diretamente para o meu crescimento profissional e pessoal. Em especial, agradeço à *Dra. Melina Oliveira Melito*, pela confiança, incentivo e apoio técnico ao longo do desenvolvimento deste trabalho.

A todos que, direta ou indiretamente, fizeram parte dessa jornada: o meu muito obrigado.

What we call chaos is just patterns we haven't recognized.
What we call random is just patterns we can't decipher.
(Chuck Palahniuk)

Resumo

SOUZA, Elielson Nogueira. **Aplicação de Tecnologia *Matching* para realizar a Correspondência entre os Usuários e Especialistas na Prestação de Serviços de Assessoria Científica**. Orientador: Dr. Fabio Alexandre Spanhol. 2025. 130f. Dissertação (Mestrado em Ciência da Computação) – Universidade Estadual do Oeste do Paraná, Cascavel – Paraná, 2025.

Os avanços recentes em Processamento de Linguagem Natural (PLN), especialmente com o uso de Redes Neurais Profundas, têm promovido melhorias significativas na interação entre humanos e máquinas. Esses desenvolvimentos possibilitam que computadores compreendam e respondam à linguagem humana com alta precisão, impulsionando a automação e a eficiência de processos operacionais em instituições públicas e privadas. Este trabalho teve como objetivo desenvolver um modelo de matching baseado em PLN, para determinar a correspondência entre usuários e especialistas científicos, aplicado em uma plataforma especializada em revisão acadêmica e assessoria científica. Os testes foram realizados com dados reais, utilizando a similaridade do cosseno entre vetores de embeddings para identificar as correspondências mais relevantes. A validação dos resultados foi conduzida por meio de métricas como precision, Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain (nDCG) e Hit Ratio, além da comparação com um ground truth estabelecido manualmente pela equipe técnica da SciBees. Isso permitiu avaliar a capacidade dos modelos em recuperar correspondências relevantes de forma precisa. Foram testados os modelos Bertimbau nas versões base e large e os modelos OpenAI small e large, dos modelos testados, o OpenAI-large apresentou melhor desempenho, alcançando valores médios de precision de 0,5667 e MRR de 0,8833, além de uma taxa Hit Ratio de 100% no Top-3 ou seja, em todos os casos avaliados, pelo menos um expert relevante figurou entre os três primeiros recomendados. Esses resultados evidenciam a consistência das recomendações geradas. Adicionalmente, a metodologia demonstrou flexibilidade e potencial de replicação, reforçando sua aplicabilidade em diversos domínios, como saúde, recursos humanos e agricultura.

Palavras-chave: pareamento; processamento de linguagem natural; consultoria científica.

Abstract

SOUZA, Elielson Nogueira. **Application of Matching Technologies to Perform the Correspondence Between Users and Experts in the Provision of Scientific Advisory Services**. Orientador: Dr. Fabio Alexandre Spanhol. 2025. 130f. Dissertação (Mestrado em Ciência da Computação) – Universidade Estadual do Oeste do Paraná, Cascavel – Paraná, 2025.

Recent advances in Natural Language Processing (NLP), especially through the use of Deep Neural Networks, have promoted significant improvements in the interaction between humans and machines. These developments enable computers to understand and respond to human language with high accuracy, driving automation and operational efficiency in public and private institutions. This study aimed to develop an NLP-based matching model focused on achieving precise correspondence between users and scientific experts, applied within a platform specialized in academic review and scientific advisory services. The experiments were conducted using real-world data, applying cosine similarity between embedding vectors to identify the most relevant matches. The validation of results was performed using metrics such as Precision, Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain (nDCG), and Top-k Hit Ratio, in addition to comparison with a ground truth manually established by the SciBees technical team. This approach enabled a precise assessment of the models' ability to retrieve relevant matches. The Bertimbau base and large versions of the models, as well as the OpenAI small and large versions, were tested. Of the models tested, the OpenAI-large model performed best, achieving average precision of 0.5667 and MRR of 0.8833, as well as a 100% hit ratio in the top three. This means that in all cases evaluated, at least one relevant expert was among the top three recommended. These results demonstrate the precision and consistency of the generated recommendations. Additionally, the methodology showed flexibility and replication potential, reinforcing its applicability in various domains such as healthcare, human resources, and agriculture.

Keywords: pairing; natural language processing; scientific consulting.

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

ABCNN *Attention-Based Convolutional Neural Network*. 61

API *Application Programming Interface*. 91, 94, 99, 101, 114–116, 118

AraBERT *Arabic Bidirectional Encoder Representations from Transformers*. 60, 69, 71

AraElectra *Arabic Efficiently Learning an Encoder that Classifies Token Replacements Accurately*. 60

AraElectra-SQuAD *AraElectra fine-tuned on the Stanford Question Answering Dataset*. 60

ATM Android Test Matching. 63

AUROC *Area Under the Receiver Operating Characteristic Curve*. 62

BAC *BiLSTM-Attention-CRF*. 58

BERT *Bidirectional Encoder Representations from Transformers*. 20, 42–45, 57–63, 65, 69–72, 74, 86

BERT-QAnet *BERT-based Question Answering Network*. 59

BERTimbau *Bidirectional Encoder Representations from Transformers* Pretrained Model for Portuguese. 45, 91, 93, 94, 97, 101, 103, 105, 106, 110, 112–115, 118

BERTimbau-base *BERT Pretrained Model for Portuguese base*. 45, 81, 85, 86, 91, 93–99, 103–106, 110, 112, 113, 117, 118

BERTimbau-large *BERT Pretrained Model for Portuguese large*. 45, 81, 85, 86, 94, 97–99, 104–108, 110, 112, 113, 117, 118

BETO *model bert pre-trained on Spanish corpora*. 59

BFM *Back and Forth Matching*. 56

Bi-GRU *Bidirectional Gated Recurrent Unit*. 58, 61, 62

BiGRU-SF *Bidirectional Gated Recurrent Unit with Semantic Fusion*. 62

BiLSTM *Bidirectional Long Short-Term Memory*. 57, 58, 61, 62, 69, 70, 74

BiMM *Biomedical Text Segmentation*. 61

BioBERT *Bidirectional Encoder Representations from Transformers for Biomedical Text Mining*. 57, 60, 69, 71

BLEU *Bilingual Evaluation Understudy*. 58

BM25 *Best Matching* 25. 57, 58, 60–62

BPE *Byte Pair Encoding*. 49

BPTT *Backpropagation Through Time*. 28, 29, 31

BT-TPF *lightweight intrusion detection model*. 63

C4 *Colossal Clean Crawled Corpus*. 46, 47

CA-RNN *Context-Aware Recurrent Neural Network*. 63

CAMeLBERT *Contextualized Arabic Model Embedding Language BERT*. 60, 71

CAPES Coordenação de Aperfeiçoamento de Pessoal de Nível Superior. 79, 81

CBOW *Continuous Bag of Words*. 40, 41

CEBM25CAT *Cross-lingual Embedding-based BM25 Context-Aware Transformer*. 62

CEC *Constant Error Carousel*. 30

CFO *Computation Flow Orchestrator*. 57

CIDEr *Consensus-based Image Description Evaluation*. 58

CLEF *Cross-Language Evaluation Forum*. 60

CNN *Convolutional Neural Network*. 25, 57, 59–62, 69, 70

ColBERT *Contextualized Late Interaction over BERT*. 60

ColBERT-X *Contextualized Late Interaction over BERT with eXpansion*. 60, 71

ConvBERT *Convolutional Bidirectional Encoder Representations from Transformers*. 58

ConvNet *Convolutional Neural Network*. 61

COVID-19 *Coronavirus Disease 2019*. 60, 61

Craftdroid *Context-Aware Test Case Recommendation Framework for Android*. 63

DBSCAN *Density-Based Spatial Clustering of Applications with Noise*. 56

Deep-BERT *Deep Bidirectional Encoder Representations from Transformers*. 62

DenseNet *Densely Connected Convolutional Network*. 59

DistilBERT *Distilled Bidirectional Encoder Representations from Transformers*. 44, 45, 61, 69, 71

DL *Deep Learning*. 25, 42

DLAM *Deep Learning Approximate Matching*. 62

DSSM *Deep Structured Semantic Model*. 61

EEG *Electroencephalography*. 57

EF-SBERT *Element-Focused Sentence-BERT*. 60

ELMo *Embeddings from Language Models*. 57, 59, 71

EMA *European Medicines Agency*. 59

ESIM *Enhanced Sequential Inference Model*. 61

FastT5 *Fast Text-to-Text Transfer Transformer*. 60

FBAB *Fine-Tuning BERT-Attention-BiLSTM*. 61

GloVe *Global Vectors for Word Representation*. 40–42, 56, 69, 71

GLOW *Global Weighted Self-Attention Network*. 58

GLUE *General Language Understanding Evaluation*. 45

GPT *Generative Pre-trained Transformer*. 48, 49

GPT-3.5 *Generative Pre-trained Transformer 3.5*. 61

GPT-4 *Generative Pre-trained Transformer 4*. 61

GPU *Graphics Processing Unit*. 85, 113, 114

GRU *Gated Recurrent Unit*. 30, 31, 69

GUI *Graphical User Interface*. 63

HC4 *Headlines Corpus 4*. 60

Himu-QAAN *Hierarchical Multi-Granularity Question-Aware Attention Network*. 59

HR *Hit Ratio*. 49, 52, 92–95, 97, 99, 101, 102, 106–109, 117

HRRP *High-Resolution Range Profile*. 60

HWR *Handwriting Recognition*. 61

I3D *Inflated 3D ConvNet*. 57

IA Inteligência Artificial. 21, 35, 55

ICD-10 *International Classification of Diseases*, 10th Revision. 61

IDF *Inverse Document Frequency*. 39

JSON *JavaScript Object Notation*. 35

KG-BERT *Knowledge Graph Bidirectional Encoder Representations from Transformers*. 58, 69, 71

LiSA *Literature Search Application*. 59

LSTM *Long Short-Term Memory*. 29–31, 57, 58, 61, 62, 69

MAP *Mean Average Precision*. 60, 62, 73

MemNet *Memory Network*. 58

ML *Machine Learning*. 24, 25, 55

MLM *Masked Language Model*. 42, 45

MLP *Multilayer Perceptron*. 27, 59

MRR *Mean Reciprocal Rank*. 49–51, 60, 73, 92–95, 97–102, 106, 109, 110, 117

MSMARCO *Microsoft Machine Reading Comprehension*. 60, 62

MTSM *Multi-Task Semantic Matching*. 58, 71

nDCG *Normalized Discounted Cumulative Gain*. 49, 51, 62, 73, 92–94, 96, 98, 100, 102, 106, 111, 112, 117

NER *Named Entity Recognition*. 56, 58, 59, 70

NICT-BERT *National Institute of Information and Communications Technology BERT*. 61, 71

NLI *Natural Language Inference* . 61

NSP *Next Sentence Prediction*. 42, 44

OCR *Optical Character Recognition*. 57, 62

OLCBot *Online Legal Consultation Bot*. 59

OpenAI-large *OpenAI Embedding Model text-embedding-3-large*. 81, 85, 91, 94, 101–103, 105–107, 110, 112, 114, 115, 117

OpenAI-small *OpenAI Embedding Model text-embedding-3-small*. 81, 85, 91, 94, 99–101, 104–106, 112, 114, 115, 117

PICOC *Population, Intervention, Comparison, Outcome e Context*. 55

PLM *pre-trained language model*. 70

PLN Processamento de Linguagem Natural. 20–25, 28, 29, 31, 34, 35, 38, 42, 45, 46, 48, 54–58, 67, 69, 74, 75, 85, 88, 93, 117

PubMedBERT *Bidirectional Encoder Representations from Transformers pre-trained on PubMed abstracts and full-text articles*. 61

QA *Question Answering*. 50, 59, 60, 71

R-Drop *Regularized Dropout*. 61

RAG *Retrieval-Augmented Generation*. 118

ResNet *Residual Network*. 57

RNA Rede Neural Artificial. 25–27

RNN *Recurrent Neural Network*. 25, 28, 29, 31, 69

RoBERTa *Robustly Optimized BERT Approach*. 44, 58, 62, 63, 69, 71

ROUGE-L *Recall-Oriented Understudy for Gisting Evaluation – Longest Common Subsequence*. 59

SBERT *Sentence-BERT*. 60, 61, 71

SciBERT *Scientific Bidirectional Encoder Representations from Transformers*. 69, 71

SciFive *Scientific Text-to-Text Transfer Transformer based on T5 for biomedical and scientific tasks*. 61

SEED-IV *SJTU Emotion EEG Dataset IV*. 57

SFTM *Similarity-based Flexible Tree Matching*. 60

Skip-Gram *Skip-Gram Architecture*. 40, 41

SQL *Structured Query Language*. 56

SQuAD *Stanford Question Answering Dataset*. 45

SVM *Support Vector Machine*. 62

t-SNE *t-distributed Stochastic Neighbor Embedding*. 56

T5 *Text-to-Text Transfer Transformer*. 46, 47, 58, 60, 61, 63, 71

TabTransformer *Transformer-based Model for Tabular Data*. 62

TED *Tree Edit Distance*. 60

TEMdroid *Tool for Evaluating Mobile Droid-based applications*. 63

TF *Term Frequency*. 39

TF-IDF *Term Frequency – Inverse Document Frequency*. 34, 39, 57, 59, 60, 62, 65, 67, 69

TREC DL'19 *Text REtrieval Conference Deep Learning 2019*. 62

TREC DL'20 *Text REtrieval Conference Deep Learning 2020*. 62

UNBERT *User-News Matching BERT*. 59

UNIOESTE Universidade Estadual do Oeste do Paraná. 76, 81, 89

UTH-BERT *User–Topic–Hashtag BERT*. 61, 71

VIST *Visual Storytelling Dataset*. 58

ViT *Vision Transformer*. 63

WHCR *Word Hashing and Convolutional Representation*. 61

WoBERT *Word-order-preserving BERT*. 61

Word2Vec *Word to Vector*. 40, 41, 59, 61, 62, 65, 67, 69, 71

XLM-RoBERTa *Cross-lingual Language Model - Robustly Optimized BERT Approach*. 60, 71

XML *Extensible Markup Language*. 35

ZSP *Zero-Shot Prediction*. 61

Sumário

1 Introdução

Avanços em Processamento de Linguagem Natural (PLN) estão impulsionando as principais empresas de tecnologia do mundo a introduzirem modelos com alto impacto em seus negócios, como estratégias que utilizam arquitetura *transformer* (VASWANI et al., 2017) e *Bidirectional Encoder Representations from Transformers* (BERT) (DEVLIN et al., 2019), conforme destacado por Chen et al. (2023). Esses modelos possibilitaram a geração de aplicações inovadoras que estão transformando operações empresariais e acadêmicas, impulsionando resultados, aumentando a produtividade e economizando tempo.

Várias rotinas operacionais estão sendo automatizadas nas empresas e ambientes acadêmicos. Por exemplo, a utilização de *software* para detectar plágio em trabalhos de pesquisa (KHADILKAR; KULKARNI; BONE, 2018). No contexto da tecnologia da informação, a automação na detecção de *bugs* em sistemas tem sido aprimorada por meio de algoritmos baseados em PLN e aprendizado de máquina, aumentando a precisão na triagem de relatórios duplicados e reduzindo o tempo de resposta (PATIL; JADON, 2023).

Abordagens híbridas que combinam redes neurais e modelos como BERT estão sendo aplicadas na moderação automática de conteúdo em mídias sociais, ajudando a identificar textos ofensivos e melhorar a segurança digital em plataformas *online* (WADUD et al., 2023). Portanto, os avanços em PLN não facilitam apenas a automação em contexto empresarial, mas também estabelecem uma base sólida para o desenvolvimento de algoritmos de *matching*.

O *matching* é um processo de emparelhamento em que dois ou mais elementos são combinados de forma a garantir uma correspondência adequada, seja por semelhança ou por uma relação específica entre eles. Problemas de *matching* são classificados como explícitos, quando há listas de preferências previamente definidas, e implícitos, quando essas preferências precisam ser inferidas a partir de dados disponíveis (REN et al., 2021). Os algoritmos de *matching* são utilizados na alocação de recursos em diversas áreas, desde operações de mercado até assistentes virtuais pessoais.

Na Ciência da Computação, esses algoritmos abrangem diversos nichos de soluções de problemas, como a correspondência de perguntas e respostas em sistemas de recuperação de informação (YANG et al., 2016), a correspondência de usuários e itens em sistemas de recomendação (ZHANG et al., 2019) e o emparelhamento de entidades e relações em grafos de conhecimento (BORDES et al., 2014). O *implicit matching* refere-se à busca por métodos capazes de emparelhar dois ou mais objetos com características similares, concentrando-se no processo de cálculo de pontuação de similaridade (REN et al., 2021). Essa pontuação representa numericamente o grau de correspondência entre dois objetos.

Um dos modelos baseados em *implicit matching* é o *retrieval matching*. Nessa abordagem, as consultas feitas pelos usuários em um buscador refletem suas necessidades, as quais são registradas e classificadas em um banco de dados, gerando assim os resultados da pesquisa (REN et al., 2021). Algoritmos que utilizam o *retrieval matching* são empregados em *chatbots*, programas capazes de simular interações naturais com usuários por meio de técnicas de PLN (AMALIA et al., 2022).

Os modelos de PLN baseados em *retrieval matching* são utilizados, desde a simples compreensão de diálogos até sistemas de perguntas e respostas em grande escala conectando-se com um banco de dados de grande porte (KIM et al., 2021). Um dos tipos de algoritmos de *implicit matching* é o *expertise matching* que pode ser definido como o processo de encontrar dois ou mais indivíduos com as condições, conhecimento e habilidades necessárias ao contexto do emparelhamento (ELGAMMAL et al., 2021). Um sistema de correspondências de empregos, por exemplo, combina candidatos aos empregos com base em uma série de fatores, tornando a triagem do processo de seleção muito mais fácil e ágil para recrutadores.

Apesar dos avanços em modelos de *implicit matching*, o emparelhamento ainda apresenta limitações devido à adoção limitada de métodos baseados em *deep learning*. A ausência de modelos como *transformers* e abordagens de *embeddings* semânticos limita a capacidade do sistema de capturar nuances nos dados, resultando em correspondências menos precisas. Como resultado, esses modelos operam majoritariamente com dados explicitamente estruturados, sem a capacidade de inferir relações contextuais mais profundas entre as entidades (SHIMADA; YAMAZAKI; TAKANO, 2020). Diante dessas limitações, esta pesquisa propõe o desenvolvimento de uma solução que integra tecnologias avançadas de PLN e Inteligência Artificial (IA), com o objetivo de aprimorar o *matching* entre usuários e especialistas, superando os desafios identificados.

Este projeto integra o Edital nº 68/2022 CNPq, vinculado ao Programa de Mestrado e Doutorado Acadêmico para Inovação MAI/DAI, que tem como finalidade fomentar a inovação tecnológica por meio de parcerias entre universidades e empresas. A pesquisa propõe-se a atender à demanda por um sistema de *matching* preciso e eficiente, voltado à conexão entre usuários e *experts* da rede de apoio da sciBees, com base na aplicação de dados reais.

A sciBees [1] é uma empresa inovadora que atua na intermediação entre pesquisadores e *experts*, oferecendo serviços que viabilizam e facilitam a colaboração científica, desde mentorias até formatação de documentos acadêmicos e apoio a projetos científicos. Seu objetivo é otimizar a conexão entre profissionais da área de pesquisa, promovendo um maior impacto na produção científica.

Atualmente, a seleção do *expert* mais adequado ocorre de forma manual. Essa atividade requer tempo e o envolvimento de profissionais com conhecimento técnico e experiência para identificar qual combinação entre pesquisador e *expert* resultará em maior aproveitamento. O desenvolvimento de um modelo baseado em abordagem de *matching* para a sciBees apresenta-se como uma solução para aprimorar a eficiência e a precisão dessas conexões. A proposta consiste em extrair e estruturar dados relevantes para o processo de correspondência, levando em consideração as necessidades e características dos pesquisadores, bem como as habilidades e experiências dos *experts*, a fim de garantir maior assertividade na recomendação.

Neste estudo, desenvolvemos um modelo de *matching* baseado na abordagem de *implicit matching*, *retrieval* e *expertise matching*. O objetivo foi realizar o emparelhamento automatizado entre usuários (pesquisadores) e *experts* cadastrados na plataforma sciBees. Este estudo contribui na área dos modelos de *implicit matching* com o desenvolvimento de um modelo de *matching* com a utilização de PLN e aplicar modelos pré-treinados ajustados para a correspondência entre usuários e *experts* científicos.

1.1 Objetivos Geral

Desenvolver e validar modelo de *matching* utilizando PLN para realizar a correspondência entre os usuários e *experts* na prestação de serviços de assessoria científica.

1.2 Objetivos Específicos

- Compreender o estado da arte sobre os algoritmos de *matching* através de uma revisão da literatura;
- Desenvolver um modelo de *matching* que incorpore técnicas de PLN e modelo pré-treinado, adaptado as especificidades dos serviços de assessoria científica;
- Validar o modelo de *matching* através de testes com dados reais.

1.3 Organização do Trabalho

O restante deste trabalho está estruturado da seguinte forma.

[1] <www.scibees.com.br/>

O Capítulo 2 apresenta a fundamentação teórica, abordando os principais conceitos relacionados à pesquisa, tais como PLN, algoritmos de *matching* e técnicas de aprendizado de máquina.

Na sequência, o Capítulo 3 contempla a revisão da literatura, discutindo os estudos mais relevantes sobre o tema, as abordagens metodológicas empregadas, os resultados alcançados e as limitações identificadas.

O desenvolvimento do modelo proposto é detalhado no Capítulo 4, que descreve os materiais utilizados, a metodologia adotada, o *dataset*, as ferramentas e bibliotecas aplicadas, bem como o delineamento experimental.

Os resultados obtidos e a análise das evidências observadas a partir dos experimentos são apresentados no Capítulo 5, com base nas métricas de avaliação quantitativa.

Por fim, o Capítulo 6 traz as conclusões do trabalho, destacando as contribuições, limitações e possíveis direções para pesquisas futuras.

2

Fundamentação Teórica

Neste capítulo são apresentados os principais conceitos teóricos que fundamentam o desenvolvimento do sistema de *matching* entre pesquisadores e *experts*. O foco está nas técnicas de PLN e aprendizado de máquina ou *Machine Learning* (ML), com ênfase no uso de modelos de *embeddings* e na arquitetura *transformer*. Além disso, discute-se também conceitos como similaridade semântica e abordagens de pré-processamento de texto, além de métodos de avaliação de correspondência entre perfis.

2.1 Aprendizado de Máquina

O ML utiliza métodos estatísticos e computacionais para criar modelos capazes de identificar padrões e realizar predições para tomadas de decisão com base em dados observados (SILVA, 2005). Esses modelos são desenvolvidos para aprender a executar tarefas complexas, processar dados em larga escala e realizar análises descritivas e preditivas (SOUTO et al., 2003).

Os algoritmos de ML podem ser amplamente categorizados em três classes principais (MAHESH, 2020):

- **Aprendizado Supervisionado**: o modelo é treinado com dados rotulados, onde cada entrada está associada a uma saída esperada. Exemplos incluem regressão linear e árvores de decisão, frequentemente utilizadas em tarefas como previsão de preços e classificação de imagens.
- **Aprendizado Não Supervisionado**: aplicado para explorar estruturas latentes em dados não rotulados. Técnicas como *clustering* (agrupamento) e redução de dimensionalidade são usadas em segmentação de clientes e visualização de grandes conjuntos de dados.
- **Aprendizado por Reforço**: uma abordagem na qual um agente aprende a interagir com o ambiente por meio de tentativa e erro, otimizando uma função de recompensa cumulativa.

Essa técnica é utilizada em áreas como robótica e jogos.

Além das categorias de aprendizado supervisionado, não supervisionado e por reforço, os algoritmos de aprendizado de máquina são comumente aplicados em três tarefas fundamentais: classificação, regressão e agrupamento (AYODELE, 2010). A classificação é uma tarefa em aprendizado supervisionado, na qual o objetivo é construir um modelo capaz de associar instâncias de entrada a categorias discretas com base em exemplos rotulados. O processo envolve a indução de um classificador a partir de um conjunto de treinamento, com o intuito de generalizar o comportamento observado e realizar predições sobre novas instâncias (KOTSIANTIS et al., 2007).

A regressão tem como objetivo estimar a esperança condicional de uma variável dependente contínua dada uma ou mais variáveis explicativas. No contexto de aprendizado de máquina, o foco recai na minimização do erro de previsão fora da amostra, sem impor previamente uma forma funcional fixa para a relação entre as variáveis nem hipóteses sobre a distribuição dos erros; a escolha do modelo é guiada pelo desempenho preditivo (ATHEY; IMBENS, 2019).

O agrupamento (ou *clustering*) é uma técnica de aprendizado não supervisionado cujo objetivo é particionar um conjunto de dados em *clusters*, ou seja, grupos cujos elementos apresentem alta similaridade entre si e baixa similaridade com elementos de outros grupos (XU; WUNSCH, 2005). Diferente da classificação supervisionada, o agrupamento não depende de rótulos previamente atribuídos, buscando identificar estruturas naturais ou latentes nos dados.

As técnicas clássicas de ML geralmente requerem a extração manual de atributos a partir dos dados e são eficazes em contextos com dados estruturados e conjuntos de características bem definidos. No entanto, essas abordagens podem enfrentar limitações ao lidar com dados complexos ou não estruturados, como imagens, áudios ou textos em linguagem natural.

O aprendizado profundo ou *Deep Learning* (DL) tem capacidade de resolver problemas de alta dimensionalidade e processar dados não estruturados, como imagens, áudio. Essas abordagens englobam arquiteturas, como redes convolucionais ou *Convolutional Neural Network* (CNN) para processamento de imagens e redes neurais recorrentes ou *Recurrent Neural Network* (RNN) para dados sequenciais. Essas arquiteturas permitem a extração de representações hierárquicas diretamente dos dados brutos. O treinamento dessas redes é realizado por meio de algoritmos como a retropropagação (*backpropagation*), que ajusta os parâmetros da rede para minimizar os erros preditivos (LECUN; BENGIO; HINTON, 2015).

A Rede Neural Artificial (RNA) se enquadra como uma classe de algoritmos supervisionados ou não supervisionados, capazes de modelar relações não lineares entre variáveis. Sua arquitetura, inspirada em mecanismos neurais biológicos, permite o processamento de dados de alta dimensionalidade e não estruturados, pode ser aplicada em tarefas como classificação de imagens, reconhecimento de voz, análise de séries temporais e PLN. A seguir, apresenta-se o funcionamento das RNA, suas estruturas fundamentais e o papel que desempenham na construção

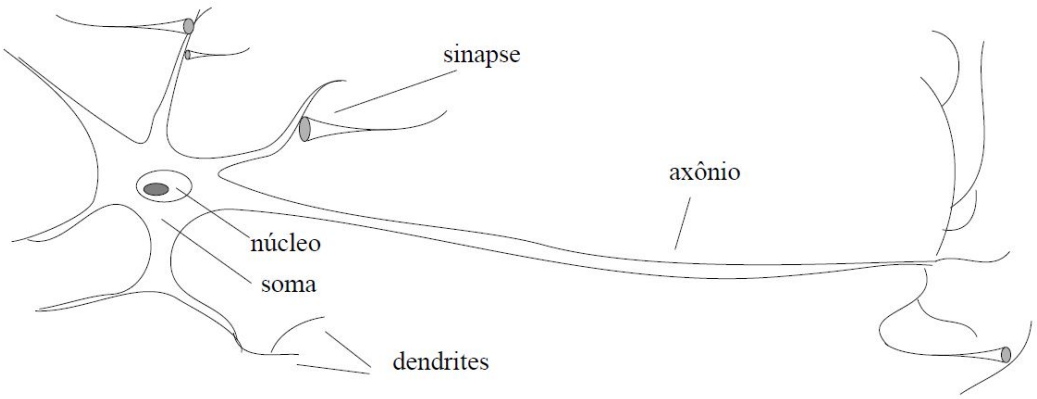
de modelos preditivos.

2.1.1 Redes Neurais Artificias (RNAs)

Na Figura 1 é apresentado um modelo simplificado de um neurônio biológico humano. O neurônio é uma célula altamente especializada composta por um corpo celular (soma), que abriga o núcleo, e por estruturas como dendritos e axônio. Os dendritos recebem estímulos provenientes de outros neurônios ou do ambiente, enquanto o axônio transmite sinais elétricos para outros neurônios por meio de sinapses. Esses sinais, baseados em reações químicas e elétricas, representam o processamento e a transmissão de informações, que são fundamentais para as funções cognitivas e motoras do organismo.

As RNAs são modelos computacionais inspirados na estrutura e no funcionamento do cérebro humano. Elas são formadas por camadas de neurônios artificiais organizados hierarquicamente, simulando a forma como os neurônios biológicos processam informações e se comunicam entre si (RAUBER, 2005). As RNAs utilizam essa analogia para replicar o comportamento dos neurônios biológicos, criando sistemas computacionais capazes de aprender, reconhecer padrões e tomar decisões. Cada neurônio artificial recebe entradas ponderadas, realiza operações matemáticas (como somas e ativações não lineares) e transmite saídas para os neurônios da próxima camada, contribuindo para o processamento da informação de forma distribuída e paralela.

Figura 1 – Representação de um neurônio biológico



Fonte: Rauber (2005)

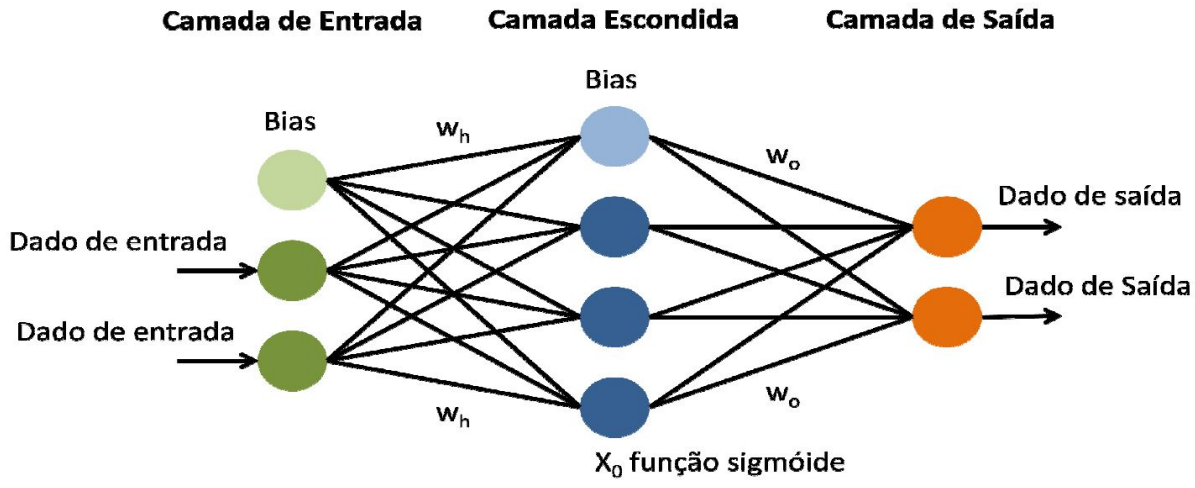
As RNAs são algoritmos computacionais que apresentam um modelo matemático inspirado na estrutura de organismos inteligentes. Dessa forma, a RNA é capaz de aprender e tomar decisões baseadas em seu próprio aprendizado (KOVÁCS, 2006). Um modelo básico de RNA possui diferentes componentes (BRAGA; LUDERMIR; CARVALHO, 2000), dentre os quais:

- **Conjunto de sinapses**: conexões entre os neurônios da RNA. Cada uma delas possui um peso sináptico;
- **Integrador**: realiza as somas dos sinais de entrada da RNA, ponderados pelos pesos sinápticos;
- **Função de ativação**: restringe a amplitude do valor de saída de um neurônio;
- ***Bias***: valor aplicado externamente a cada neurônio e tem o efeito de aumentar ou diminuir a entrada líquida da função de ativação.

O *Perceptron* é um dos modelos mais simples de RNA e foi desenvolvido por Rosenblatt (1958). Ele consiste em uma única camada de neurônios que realiza operações lineares seguidas de uma função de ativação. Apesar de suas limitações, como a incapacidade de resolver problemas não linearmente separáveis, o *Perceptron* serviu de base para o desenvolvimento de arquiteturas mais avançadas. Uma dessas arquiteturas é a Rede Neural *Perceptron* multicamadas ou *Multilayer Perceptron* (MLP), que supera essas limitações ao introduzir camadas ocultas entre a entrada e a saída, permitindo o aprendizado de relações não lineares.

Uma MLP com ligações unidirecionais entre os neurônios na direção das entradas para as saídas são normalmente chamadas de rede neural progressiva ou *feedforward* (GOMES, 2005). Na Figura 2 apresenta-se a estrutura de uma MLP *feedforward*.

Figura 2 – Rede Neural *Feedforward* Multicamadas



Fonte: Tissot, Camargo e Pozo (2012)

Esse tipo de rede é composto por três camadas principais: camada de entrada, camada escondida (intermediária) e camada de saída. Cada neurônio da camada de entrada conecta-se a todos os neurônios da camada intermediária por meio de pesos sinápticos w_h, e esses, por sua vez, conectam-se aos neurônios da camada de saída com pesos w_o. A rede pode incluir neurônios adicionais chamados *bias*, que possuem saída constante igual a 1 e são conectados às camadas

subsequentes com seus próprios pesos. A função de ativação nesse caso representada pela função sigmóide é aplicada nos neurônios das camadas escondida e de saída, permitindo a introdução de não linearidades e o aprendizado de representações complexas (TISSOT; CAMARGO; POZO, 2012).

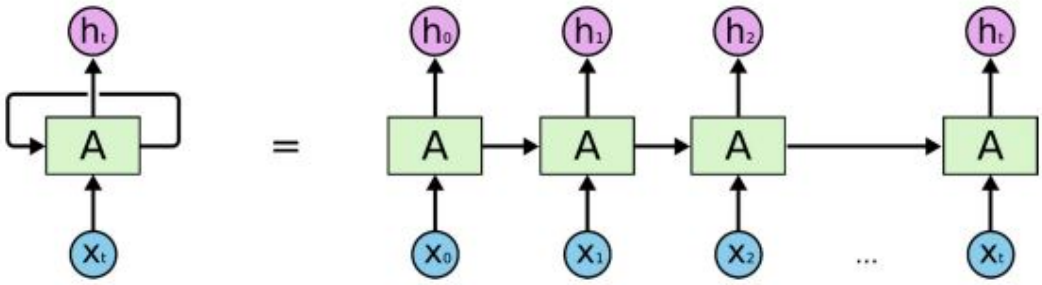
Embora eficazes em tarefas com dados estáticos, redes *feedforward* não conseguem capturar dependências temporais, pois processam as entradas de forma independente. Para superar essa limitação, foram desenvolvidas as RNNs, que incorporam conexões recorrentes e permitem o armazenamento de informações ao longo do tempo.

2.1.2 Redes Neurais Recorrentes (RNN)

As *Recurrent Neural Networks* (RNNs) são um tipo de rede neural projetada para processar dados sequenciais, como séries temporais e textos, sendo utilizadas no PLN. A principal característica das RNNs é sua capacidade de manter informações de estados anteriores, permitindo capturar dependências temporais em uma sequência de dados (YANG et al., 2020).

Na Figura 3 é apresentado o funcionamento de uma RNN ao longo de diferentes passos temporais. Cada unidade A representa uma célula recorrente da rede, responsável por processar uma entrada x_t no tempo t, combinando-a com o estado oculto da etapa anterior h_{t-1} para produzir um novo estado h_t. Esse estado é, então, propagado para a próxima unidade da sequência, criando um encadeamento temporal que permite à rede reter informações sobre o histórico da entrada.

Figura 3 – Processamento sequencial em uma RNN



Fonte:Le et al. (2019)

A arquitetura de uma RNN inclui uma camada de entrada, uma ou mais camadas ocultas com conexões recorrentes e uma camada de saída. A principal característica dessas redes é o uso de conexões de realimentação (*feedback*), permitindo que a saída de uma etapa influencie as próximas. Essa estrutura torna as RNNs adequadas para modelar dados sequenciais, como texto, séries temporais e sinais de voz, pois possibilita capturar dependências ao longo do tempo (LE et al., 2019).

Entretanto, as RNNs convencionais apresentam limitações significativas. Durante o treinamento, elas utilizam um método chamado retropropagação através do tempo *Backpropagation*

Through Time (BPTT), em que os pesos da rede são ajustados iterativamente, da camada de saída até a entrada, levando em consideração a influência de etapas temporais passadas (HOCHREITER; SCHMIDHUBER, 1997). Embora o BPTT funcione bem para capturar dependências curtas, ele encontra dificuldades em reter padrões de longo prazo.

O problema de explosão de gradientes ocorre quando os valores dos gradientes crescem exponencialmente, gerando instabilidade no treinamento. Em contrapartida, o problema de desaparecimento de gradientes se manifesta quando os valores dos gradientes diminuem progressivamente até quase zero, o que impede que a rede aprenda relações entre eventos distantes no tempo (PASCANU; MIKOLOV; BENGIO, 2013). Esses problemas são agravados pelo uso contínuo do *feedback* entre as camadas internas, levando a atualizações excessivas ou insuficientes nos pesos. Devido a essas limitações, as RNNs tradicionais não são adequadas para capturar dependências de longo alcance, o que impulsionou o desenvolvimento de arquiteturas mais sofisticadas, como as *Long Short-Term Memory* (LSTMs).

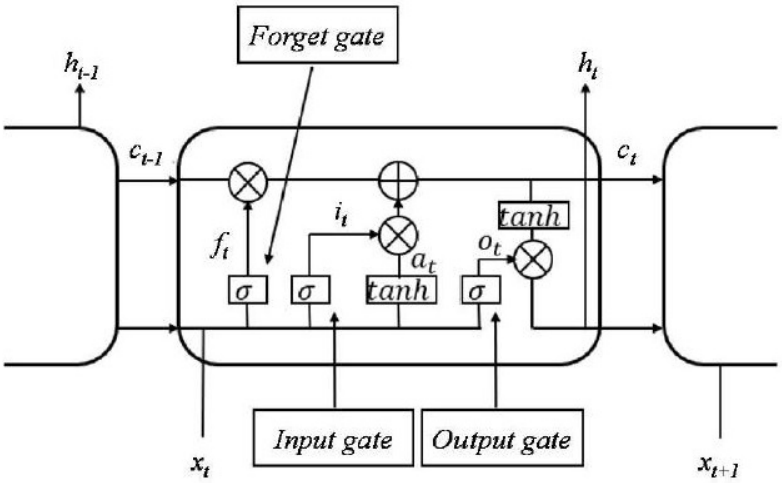
2.1.3 *Long Short-Term Memory* (LSTM)

As redes LSTMs foram desenvolvidas para superar as limitações das RNNs tradicionais, notadamente os problemas de desaparecimento e explosão do gradiente. Tais dificuldades ocorrem durante a propagação de gradientes em sequências longas, prejudicando a capacidade da rede de aprender dependências de longo prazo (BENGIO; SIMARD; FRASCONI, 1994). Para mitigar essas limitações, as LSTMs introduzem uma célula de memória interna dotada de mecanismos de controle, denominados *gates*, que regulam o fluxo de informações ao longo do tempo (STAUDEMEYER; MORRIS, 2019). No contexto de tarefas que envolvem dados sequenciais como o PLN e a análise de séries temporais, a capacidade das LSTMs de manter e manipular informações relevantes por períodos estendidos as torna uma ferramenta extremamente eficaz (HOCHREITER; SCHMIDHUBER, 1997).

A Figura 4 ilustra a estrutura interna de uma célula LSTM, destacando visualmente os principais fluxos de informação entre os componentes do modelo. Nessa figura, o elemento h_{t-1} representa a saída gerada pela célula no instante anterior, a qual contém informações processadas anteriormente e serve como base para o próximo estado. Já x_t é a entrada atual da sequência, ou seja, o dado observado no tempo t, que será combinado com a saída anterior para gerar as ativações internas da célula. O vetor a_t, localizado no centro da figura, representa as ativações intermediárias produzidas pelos *gates* internos — portas que controlam o que será esquecido, armazenado e transmitido. A variável c_{t-1} indica o estado interno de memória da célula no tempo anterior, enquanto c_t representa o novo estado de memória atualizado com base nas decisões dos *gates*. O elemento h_t, por sua vez, é a nova saída gerada pela célula no tempo atual, que será utilizada como entrada no próximo passo temporal, juntamente com o novo dado da sequência x_{t+1}.

A célula LSTM é composta por três mecanismos principais de controle, conhecidos

Figura 4 – Estrutura da célula LSTM



Fonte:Hu, Zhang e Sun (2023)

como *gates*: esquecimento, entrada e saída. Esses componentes atuam de forma coordenada para regular o fluxo de informações ao longo da sequência, permitindo que a rede retenha, atualize ou descarte dados de maneira eficiente. A porta de esquecimento decide quais informações do estado de memória anterior devem ser descartadas, com base na entrada atual e na saída do passo anterior. Já a porta de entrada determina quais novas informações devem ser incorporadas à memória da célula, avaliando a relevância dos dados atuais e gerando candidatos à atualização do estado interno. Por fim, a porta de saída define quais partes da memória atualizada serão propagadas como saída da célula, funcionando como resposta imediata e entrada para o próximo passo temporal.

Esses três mecanismos permitem que a LSTM mantenha informações relevantes por períodos prolongados, superando os problemas de desvanecimento e explosão do gradiente característicos das redes recorrentes tradicionais (HOCHREITER; SCHMIDHUBER, 1997).

Um dos principais diferenciais das LSTMs é o mecanismo denominado *Constant Error Carousel* (CEC), que permite a atualização do estado de memória por meio de operações aditivas, reduzindo os efeitos negativos sobre os gradientes durante o treinamento (HOCHREITER; SCHMIDHUBER, 1997). Esse mecanismo é complementado por uma conexão auto-recorrente com peso fixo igual a 1.0, que introduz um *feedback* com atraso de um passo de tempo, promovendo estabilidade na retenção de informações (KRATZERT et al., 2018).

2.1.4 *Gated Recurrent Unit* (GRU)

A *Gated Recurrent Unit* (GRU) é uma variação das LSTMs, proposta por Cho et al. (2014) para oferecer uma estrutura mais simples, mas ainda eficaz, para o processamento de sequências. Diferente das LSTMs, o GRU combina os portões de atualização e de esquecimento em um único *gate*, o que reduz a quantidade de parâmetros e torna o treinamento mais rápido e

eficiente em termos computacionais (DEY; SALEM, 2017). Essa simplificação permite que a GRU obtenha um desempenho similar ao das LSTMs em muitas tarefas de séries temporais e PLN, com menor custo de processamento.

As GRUs são mais simplificada em comparação com a LSTM (SHEN et al., 2018). No modelo GRU, o *gate* de atualização (z) controla a quantidade de informação da etapa anterior que será carregada para o estado atual, enquanto o *gate* de *reset* (r) decide a quantidade de informação antiga que será esquecida ao computar o novo estado ($\tilde{h}$) (CHUNG et al., 2014). Essa estrutura reduz significativamente o número de operações internas e de parâmetros, ajudando a evitar os problemas de desaparecimento e explosão de gradiente. Com menos portas que as LSTMs, a GRU simplifica o fluxo de informações dentro da rede, o que a torna tão eficiente quanto o LSTM, ideal para ser utilizado em tarefas com recursos computacionais limitados, em aplicações como previsões financeiras (SHEN et al., 2018).

2.1.5 Limitações das Redes Neurais Recorrentes e a Transição para Modelos *Transformers*

Os métodos de redes neurais, incluindo redes profundas, são geralmente baseados em métodos de gradiente, como o BPTT, para identificar os pesos apropriados da rede. Esses métodos são sensíveis à inicialização de parâmetros e tendem a ficar presos em mínimos locais. Os pesos iniciais podem ter um efeito maior no desempenho da rede neural do que a própria arquitetura da rede (MORAVVEJ et al., 2022). Outra dificuldade que as RNNs possuem é a falta de capacidade de processar dados em paralelo, sendo que as RNNs seguem um processamento sequencial e consomem mais recursos computacionais em grandes conjuntos de dados (WANG et al., 2022b). A eficiência dos *Transformer* pelo uso do mecanismo de *Multi-Head Attention*, que é eficaz em capturar correlações complexas entre características, o que aumenta significativamente a precisão na detecção de intrusões. Esse mecanismo supera a abordagem sequencial das RNNs, incluindo LSTM e GRU, que apresentam limitações em termos de paralelismo e eficiência computacional. Em contraste, os *Transformer* possibilitam uma arquitetura escalável, utilizando tokenização e camadas de atenção para processar dados em paralelo, tornando-os ideais para sistemas em larga escala (WANG et al., 2024).

Uma das inovações centrais trazidas pelos modelos *Transformers* é o mecanismo de *Self-Attention*, que permite ao modelo examinar diferentes partes da sequência de entrada simultaneamente, capturando relações contextuais complexas entre elementos distantes. Esse mecanismo é relevante para captar dependências de longo alcance, uma área em que as RNNs, incluindo LSTM e GRU, são limitadas devido à sua natureza sequencial. Nas RNNs, cada elemento depende do estado anterior, o que pode resultar em uma perda gradual de contexto para sequências longas. Em contraste, o *Transformer* calcula todas as relações de atenção em paralelo, permitindo que cada palavra ou *token* na sequência se conecte a qualquer outra parte do contexto sem a necessidade de processamento sequencial. O *Transformer* pode ser

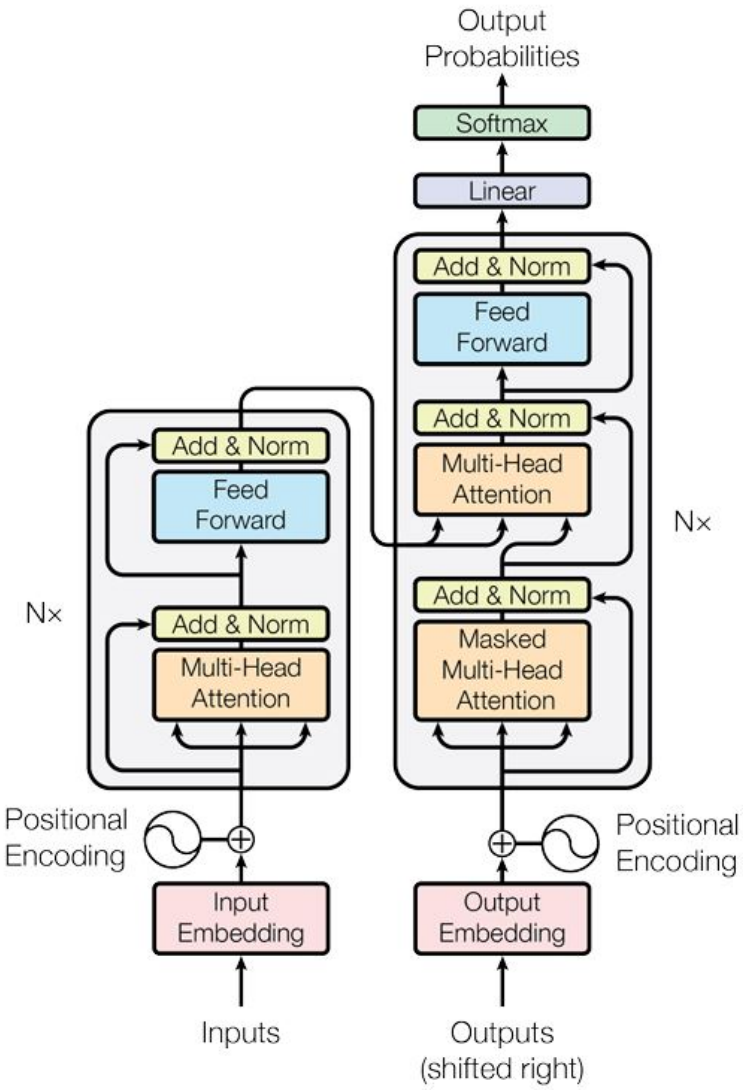
treinado significativamente mais rápido do que arquiteturas baseadas em camadas recorrentes ou convolucionais (VASWANI et al., 2017).

A Figura 5 apresenta a arquitetura do modelo *Transformer*, composta por dois blocos principais: o *encoder* e o *decoder*. O *encoder* é responsável por processar a entrada da sequência, convertendo cada *token* em uma representação vetorial contextualizada. Ele é formado por uma pilha de camadas idênticas, compostas por dois subcomponentes: uma camada de atenção multi-cabeça (*Multi-Head Attention*), que permite ao modelo considerar diferentes relações entre os *tokens* simultaneamente, e uma rede *feed-forward* aplicada ponto a ponto. Ambas as camadas são acompanhadas por conexões residuais e normalização (*Add & Norm*), o que melhora a estabilidade do treinamento. O *Positional Encoding* é somado aos *embeddings* de entrada, fornecendo à rede informações sobre a posição de cada *token* na sequência.

O *decoder*, tem como função gerar a saída da sequência de forma autoregressiva, isto é, prevendo o próximo *token* com base nos anteriores. Assim como o *encoder*, também é composto por múltiplas camadas contendo atenção multi-cabeça e redes *feed-forward*, mas com uma diferença essencial: cada camada do *decoder* inclui uma atenção mascarada (*Masked Multi-Head Attention*) que impede o acesso a posições futuras da sequência, garantindo que a geração de texto seja feita de forma ordenada. O *decoder* incorpora uma atenção cruzada (*cross-attention*) que se conecta às saídas do *encoder*, permitindo que o modelo integre o contexto da entrada na geração da resposta.

As representações produzidas pelo *decoder* passam por uma camada linear e uma função *softmax*, que converte os vetores em probabilidades associadas aos possíveis *tokens* de saída. Essa arquitetura paralelizável é utilizada em tarefas como tradução automática, sumarização de textos e modelagem de linguagem (VASWANI et al., 2017).

Figura 5 – Arquitetura do Modelo *Transformer*



Fonte:Vaswani et al. (2017)

2.2 Mineração de Texto

A mineração de texto é uma área fundamental da Ciência de Dados cujo principal objetivo é extrair informações relevantes e padrões ocultos a partir de grandes volumes de dados textuais não estruturados. De acordo com Aranha e Passos (2006), essa abordagem difere da mineração de dados tradicional, pois lida diretamente com textos em linguagem natural, os quais apresentam características específicas, como ambiguidade, polissemia e variação linguística. Essas propriedades exigem o uso de técnicas avançadas para que os dados possam ser devidamente interpretados e analisados.

A Figura 6 apresenta as principais fases da metodologia de mineração de texto, organizada em cinco fases: coleta, pré-processamento, indexação, mineração e análise. Cada fase desempenha um papel essencial na transformação de textos brutos em informações estruturadas que apoiam a

tomada de decisão.

Figura 6 – Diagrama da Metodologia de Mineração de Texto

Fonte: Aranha e Passos (2006)

A primeira fase, denominada **coleta**, envolve a construção de um corpus textual, que pode ser composto por documentos, páginas *web*, redes sociais ou outras fontes digitais. Técnicas automatizadas, como *web crawling* e *scraping*, são frequentemente utilizadas para reunir grandes volumes de dados textuais em ambientes diversos. Em seguida, ocorre o **pré-processamento**, fase relevante para a preparação dos dados. Nessa fase, são aplicadas diversas técnicas de PLN, como a tokenização, que segmenta o texto em unidades léxicas; a remoção de *stopwords*; a lematização; e a normalização linguística. Tais procedimentos são fundamentais para reduzir ruídos e padronizar os dados, facilitando o processamento posterior.

A fase de **indexação** consiste na conversão dos textos em representações numéricas, permitindo que os dados possam ser processados por algoritmos computacionais. Abordagens como *Term Frequency – Inverse Document Frequency* (TF-IDF) ou vetores gerados por modelos baseados em *transformers* são utilizadas para representar semanticamente os textos.

A fase de **mineração** compreende a aplicação de algoritmos estatísticos e de aprendizado de máquina com o objetivo de identificar padrões e gerar conhecimento a partir dos dados tratados. Entre as principais técnicas destacam-se a classificação de documentos, o agrupamento de textos semelhantes e a extração de tópicos. Essa fase representa o núcleo da mineração de texto, pois concentra a descoberta de informações inéditas.

A fase de **análise** envolve a interpretação dos resultados gerados. Diferentemente das fases anteriores, que são predominantemente automáticas, a análise exige intervenção humana para validação dos achados e extração dos *insights* mais relevantes. Os resultados podem ser apresentados por meio de visualizações, como gráficos e tabelas, facilitando sua compreensão e aplicação prática.

Muitas das técnicas utilizadas nas etapas de pré-processamento, indexação e análise na mineração de texto são derivadas do PLN. Isso porque textos não estruturados exigem transformações linguísticas para que possam ser interpretados por algoritmos. O PLN, portanto, oferece as ferramentas necessárias para lidar com a ambiguidade, complexidade e variabilidade da linguagem humana, sendo essencial para o sucesso da mineração textual.

A seguir, apresenta-se uma visão geral sobre o PLN, suas principais técnicas e aplicações, destacando sua relevância na extração de conhecimento a partir de dados textuais.

2.3 Processamento de Linguagem Natural (PLN)

O PLN é um campo interdisciplinar que integra conhecimentos de linguística, Ciência da Computação e IA, com o objetivo de desenvolver métodos e sistemas capazes de compreender, interpretar e gerar linguagem humana em sua forma textual (CASELI; NUNES, 2023).

Os dados manipulados em aplicações de PLN podem ser classificados em três categorias: estruturados, semiestruturados e não estruturados. Dados estruturados são organizados segundo esquemas fixos, como tabelas relacionais, onde cada campo possui um tipo e um formato bem definido, permitindo fazer o armazenamento e consulta das informações. Dados semiestruturados não seguem um esquema rígido, mas marcadores que conferem estrutura, como em arquivos nos formatos *Extensible Markup Language* (XML) e *JavaScript Object Notation* (JSON). Já os dados não estruturados não apresentam organização formal padronizada, como textos livres, artigos, *e-mail*, comentários em postagens em redes sociais.

Dentre essas categorias, os dados textuais não estruturados são os mais frequentemente utilizados em PLN, especialmente em contextos de análise semântica, mineração de texto, análise de sentimentos e sistemas de recomendação.

A aplicação de técnicas de PLN permite converter grandes volumes de texto em representações estruturadas e compreensíveis para algoritmos computacionais, viabilizando tarefas como extração de palavras-chave, análise de sentimentos e reconhecimento de entidades nomeadas. Tran (2017) apresenta aplicação de PLN na predição de condições mentais de pacientes a partir da descrição textual contida em notas clínicas psiquiátricas.

Além disso, Bucur, Cosma e Dinu (2021) destacam que textos informais, como postagens em redes sociais, representam um desafio para o PLN, pois as ferramentas existentes são treinadas com textos formais e não lidam bem com ruídos textuais, abreviações e erros gramaticais comuns nesses contextos. Assim, as técnicas como tokenização, normalização, lematização e vetorização são essenciais para tornar esses dados legíveis e úteis para tarefas analíticas.

Portanto, o PLN desempenha um papel central no sucesso de projetos que envolvem análise textual, sendo peça-chave para o funcionamento eficiente de processos de mineração de texto e sistemas inteligentes baseados em linguagem.

2.3.1 Tokenização

A tokenização constitui uma etapa de importância significativa no processo de pré-processamento de dados, segmentando os textos de entrada em unidades denominadas *tokens* que são as unidades básicas em que um texto é segmentado durante o pré-processamento. Um *token* pode corresponder a uma palavra, parte de uma palavra, número, símbolo ou até mesmo um caractere, dependendo da técnica de tokenização adotada.

Existem três métodos primordiais de tokenização empregados pelos participantes: (1) tokenização baseada em regras; (2) tokenização baseada em dicionário; e (3) tokenização baseada em subpalavras. No contexto da tokenização baseada em regras, são aplicadas diretrizes predefinidas para efetuar a segmentação dos textos em *tokens*. Tais diretrizes podem variar desde abordagens tão elementares quanto a tokenização por espaço em branco, até combinações mais intricadas de regras meticulosamente concebidas, como aquelas fundamentadas em gramática específica do idioma e prefixos comuns.

A tokenização baseada em dicionário demanda a construção de um vocabulário, e a segmentação do texto ocorre por meio da combinação do texto de entrada com os *tokens* contidos no vocabulário previamente elaborado. No que tange à tokenização de subpalavras, permite-se que um token seja uma subsequência de uma palavra, ou seja, unidades de subpalavras. Este método proporciona uma abordagem para lidar com palavras que não se encontram presentes no vocabulário. Um *tokenizer* de subpalavras aprende um conjunto de subpalavras comuns com base na distribuição de palavras no conjunto de dados de treinamento (HE et al., 2021).

2.3.2 *Stemming*

O *stemming* é uma técnica fundamental de pré-processamento no contexto da mineração de textos, pois permite a remoção de sufixos (ou outras partes morfológicas) para reduzir palavras ao seu radical. Essa simplificação do vocabulário contribui para melhorar o desempenho de sistemas de análise textual, recuperação de informação e classificação, além de reduzir a dimensionalidade dos dados (PORTER, 1980).

A Tabela 1 apresenta exemplo prático da aplicação do *stemming*. Observa-se que diferentes formas morfológicas da palavra “run” — como “running” (gerúndio), “runner” (substantivo derivado), “ran” (pretérito) e “runs” (presente da terceira pessoa) — são todas reduzidas ao mesmo radical “run”. Esse processo de normalização permite que essas palavras sejam tratadas como semanticamente equivalentes durante a análise, mesmo apresentando variações gramaticais distintas. Dessa forma, sistemas computacionais podem identificar relações de significado e contexto com maior eficiência.

Tabela 1 – Exemplo de aplicação do *stemming*

Palavra Original	**Radical (Stemming)**
running	run
runner	run
ran	run
runs	run

Fonte: Adaptado de Porter (1980)

Ao reduzir o vocabulário e agrupar palavras semanticamente próximas sob uma mesma forma canônica, o *stemming* melhora a eficiência de sistemas de recuperação de informações e mineração de textos. No entanto, a técnica pode apresentar limitações, como a geração de radicais inexistentes ou a fusão de termos semanticamente distintos, o que pode comprometer a precisão em algumas aplicações (WILLETT, 2006).

2.3.3 Lematização

A lematização é uma técnica que utiliza vocabulário e análise morfológica para reduzir palavras à sua forma base ou de dicionário. Diferente do *stemming*, que realiza cortes mecânicos nas palavras, a lematização considera o contexto linguístico e a classe gramatical para identificar o *lema* correspondente. Isso proporciona maior precisão na vinculação das palavras ao seu significado raiz (BALAKRISHNAN; LLOYD-YEMOH, 2014).

A Tabela 2 apresenta exemplos práticos da aplicação da lematização em diferentes palavras. Observa-se que o termo “running”, quando identificado como verbo, é reduzido corretamente ao lema “run”. Da mesma forma, “was” (forma flexionada do verbo) é mapeado para “be”, sua forma base no dicionário. Já “cars”, plural de “car”, é reduzido à forma singular. Além disso, casos mais complexos, como “better” (comparativo de “good”) e “hot” (com sentido de temperatura elevada), são tratados semanticamente: “better” é reduzido ao lema “good”, e “hot” ao sinônimo “warm”, evidenciando a capacidade da lematização de lidar com relações semânticas mais profundas.

Tabela 2 – Exemplo de aplicação da Lematização

Palavra Original	**Classe Gramatical**	**Lema (Forma Base)**
running	Verbo	run
was	Verbo	be
cars	Substantivo	car
better	Adjetivo	good
hot	Adjetivo	warm

Fonte: Adaptado de Balakrishnan e Lloyd-Yemoh (2014)

A lematização é relevante para normalizar palavras e lidar com variações morfológicas

em sistemas de recuperação de informações. Por meio de regras gramaticais e morfológicas, ela garante maior precisão na transformação das palavras em sua forma base, sendo especialmente útil em contextos que exigem interpretação semântica mais profunda (STANKOVIĆ et al., 2016).

2.3.4 *Stopwords*

Stopwords, ou palavras de ruído, são termos que aparecem com alta frequência em textos como artigos, preposições, conjunções e pronomes mas que carregam pouco ou nenhum valor semântico relevante para tarefas de análise textual (BARION; LAGO, 2008). Sua remoção é uma etapa essencial no pré-processamento de texto em PLN, especialmente em atividades como recuperação de informações, análise de sentimentos e classificação de documentos (KAUR; BUTTAR, 2018). Ao eliminar essas palavras, o vocabulário se torna mais enxuto e informativo, o que contribui para a eficiência computacional e para o foco em termos mais representativos (SARICA; LUO, 2021).

A Tabela 3 apresenta como a remoção de *stopwords* simplifica frases e evidência os termos mais relevantes para a análise. Por exemplo, na sentença original "A destruição das florestas tropicais da Amazônia", são removidas palavras como "a", "das" e "da", resultando em "destruição florestas tropicais amazônia", que preserva o núcleo semântico da frase. Da mesma forma, em "Nós fomos ao parque para passear", termos funcionais como "nós", "ao" e "para" são excluídos, e a frase reduzida "fomos parque passear" mantém o sentido essencial da ação. Essa redução de ruído melhora a relação sinal-ruído do texto, facilita a vetorização e torna o conteúdo mais informativo para algoritmos de PLN.

Tabela 3 – Exemplo de Remoção de *Stopwords*

Texto Original	**Texto após Remoção de Stopwords**
A destruição das florestas tropicais da Amazônia	destruição florestas tropicais amazônia
O gato está na casa	gato está casa
Nós fomos ao parque para passear	fomos parque passear
A criança brinca com os amigos no quintal	criança brinca amigos quintal

Fonte: Adaptado de Barion e Lago (2008)

A remoção de *stopwords* contribui para o aumento da relação sinal-ruído em textos não estruturados, permitindo que termos mais relevantes para uma tarefa específica ganhem destaque (SARICA; LUO, 2021). Essa prática também pode reduzir significativamente a dimensionalidade do vocabulário, chegando a economias de até 65% em conjuntos de dados extensos (SAIF et al., 2014).

2.3.5 *Term Frequency - Inverse Document Frequency* (TF-IDF)

O TF-IDF é uma técnica utilizada em tarefas de recuperação de informações e categorização de texto. Ele pondera a relevância de palavras individuais em documentos, equilibrando a frequência local do termo em um documento específico *Term Frequency* (TF) e sua raridade global no corpus *Inverse Document Frequency* (IDF) (YUN-TAO; LING; YONG-CHENG, 2005). O componente TF reflete a probabilidade de um termo aparecer em um documento, enquanto o componente IDF está relacionado à informatividade do termo, com base em sua distribuição no corpus (AIZAWA, 2003).

No contexto do Modelo de Espaço Vetorial, os documentos são representados como vetores em um espaço multidimensional, onde cada dimensão corresponde a uma palavra. Os pesos atribuídos às palavras são calculados com base no TF-IDF, permitindo identificar termos mais representativos para a classificação e análise de documentos. O TF-IDF é fundamental para capturar a relevância entre palavras, documentos e categorias específicas (YUN-TAO; LING; YONG-CHENG, 2005).

A fórmula clássica do TF-IDF é apresentada na Equação 1, onde o peso de um termo t em um documento d é calculado pelo produto entre sua frequência no documento TF e a frequência inversa do termo no corpus IDF (PERREAULT-JENKINS, 2020):

$$\text{TF-IDF}\,(t, d) = \text{TF}\,(t, d) \cdot \text{IDF}\,(t) \tag{1}$$

onde:

- $\text{TF}\,(t, d)$ é a frequência do termo t no documento d.
- $\text{IDF}\,(t)$, conforme definido na Equação 2, mede a raridade do termo em relação ao corpus e é calculado como:

  $$\text{IDF}\,(t) = \log \frac{N}{1 + \text{DF}\,(t)} \tag{2}$$

  sendo N o número total de documentos no corpus e $\text{DF}\,(t)$ o número de documentos que contêm o termo t.

A Equação 1 demonstra como o TF-IDF pondera termos relevantes em documentos específicos, enquanto a Equação 2 enfatiza a raridade de termos no corpus, destacando palavras distintivas para tarefas de categorização e recuperação de informações.

2.4 Modelos *Word-Embeddings* Tradicionais

A incorporação de palavras, ou *Word Embedding*, é uma representação que viabiliza a similaridade de significados entre palavras, proporcionando uma representação análoga. Esse método capacita as máquinas a aprimorarem sua compreensão vocabular, sendo uma abordagem

robusta para extrair informações semânticas latentes da linguagem. Essas informações podem ser aplicadas em diversas tarefas, como classificação de texto, ao capturar padrões de coocorrência entre palavras e integrar aspectos de raciocínio sobre o uso e significado das palavras (SOUZA; GONÇALVES; SOUZA, 2020).

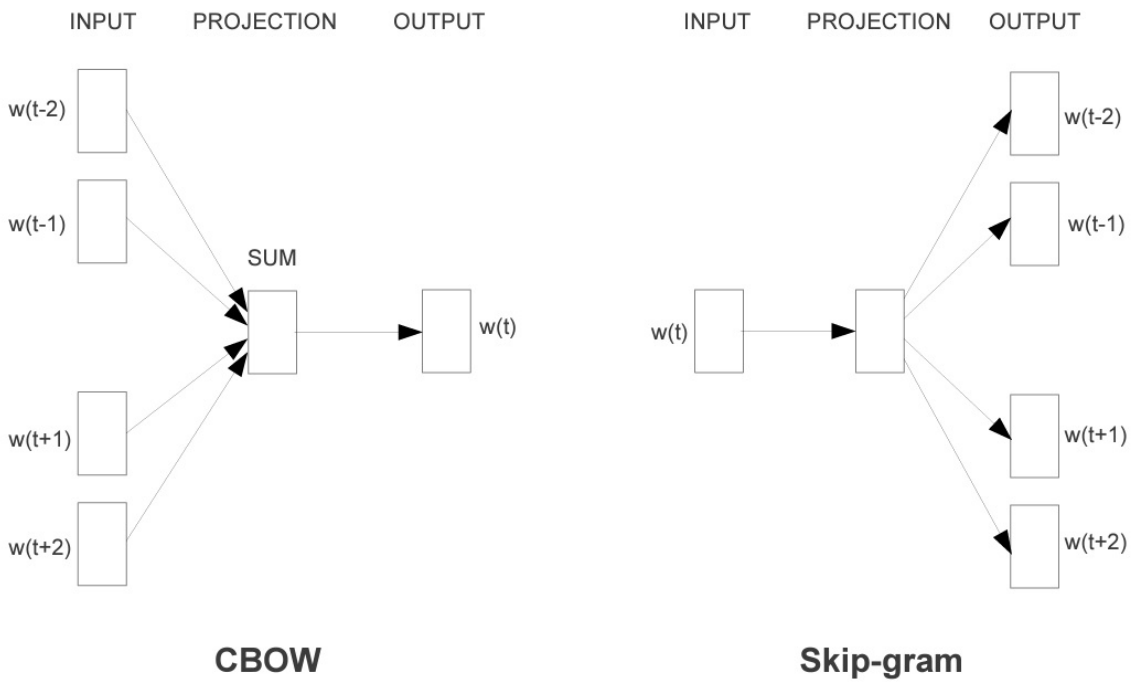
Os modelos de *word-embeddings* tradicionais, como *Word to Vector* (Word2Vec), *Global Vectors for Word Representation* (GloVe) e *FastText*, representam palavras como vetores fixos em um espaço contínuo de baixa dimensão, onde palavras semanticamente semelhantes ocupam posições próximas. Cada palavra é associada a um único vetor estático, independentemente do contexto em que aparece, o que torna o processamento computacional eficiente. Contudo, essa característica apresenta limitações importantes, como a incapacidade de lidar com ambiguidades semânticas e de capturar variações de significado em diferentes contextos (GARDAZI et al., 2025).

2.4.0.1 *Word2Vec*

No modelo Word2Vec as palavras são representadas como vetores densos em um espaço contínuo de baixa dimensão, onde palavras semanticamente semelhantes estão próximas entre si. Esse modelo utiliza duas arquiteturas principais para o treinamento dos *embeddings*: o *Continuous Bag of Words* (CBOW) e o *Skip-Gram Architecture* (Skip-Gram) (MIKOLOV et al., 2013):

A Figura 7 apresenta as duas arquiteturas principais do modelo Word2Vec o CBOW e o Skip-Gram. Ambas são responsáveis por gerar *embeddings* de palavras em um espaço vetorial contínuo, onde palavras semanticamente semelhantes estão mais próximas entre si.

Figura 7 – Arquitetura *Word2vec*



Fonte: Mikolov et al. (2013)

Na abordagem CBOW, o modelo utiliza as palavras de contexto, como $w(t-2), w(t-1), w(t+1), w(t+2)$, para prever a palavra central $w(t)$. Essa arquitetura é computacionalmente eficiente e indicada para corpora de grande dimensão, pois prioriza o aprendizado de palavras frequentes. O Skip-Gram utiliza a palavra central $w(t)$ como entrada para prever as palavras de contexto, como $w(t-2), w(t-1), w(t+1), w(t+2)$. Essa abordagem é mais adequada para capturar relações semânticas em palavras menos frequentes, devido ao seu foco em generalizar para dados com menor representatividade no corpus. Ambas as arquiteturas compartilham a mesma estrutura de projeção, onde as palavras são mapeadas para vetores densos.

2.4.0.2 Representação Vetorial com Subpalavras (*FastText*)

O modelo *FastText*, proposto por Bojanowski et al. (2017), apresenta uma extensão do modelo Skip-Gram tradicional, com o objetivo de melhorar a representação vetorial de palavras, especialmente em contextos com vocabulários extensos e idiomas morfologicamente complexos. Ao invés de associar um vetor único a cada palavra, como nos modelos Word2Vec e GloVe, o *FastText* representa cada palavra como a soma dos vetores dos seus *n-gramas* de caracteres, incluindo a própria palavra como um *token* especial. Isso permite capturar regularidades morfológicas e generalizar para palavras raras ou mesmo não vistas durante o treinamento (*out-of-vocabulary*).

A arquitetura mantém a base do Skip-Gram com amostragem negativa, mas redefine a função de *score* entre palavra e contexto ao considerar os vetores dos *n-gramas* que compõem

a palavra. A implementação utiliza *hashing* (FNV-1a) para mapear n-gramas a índices fixos, limitando o uso de memória. A abordagem permite o compartilhamento de parâmetros entre palavras com estrutura semelhante, o que melhora a semântica.

2.4.0.3 *Global Vectors for Word Representation* (GloVe)

O GloVe, desenvolvido por Pennington, Socher e Manning (2014), apresenta uma abordagem baseada na combinação de informações globais e locais para criar representações vetoriais de palavras. O modelo constrói uma matriz de coocorrência palavra-contexto, utilizando os logaritmos das frequências de coocorrência para capturar padrões semânticos e estatísticos globais no texto. Por meio de um modelo de regressão log-bilinear, o GloVe otimiza representações vetoriais que demonstram estrutura semântica clara, alcançando resultados superiores em tarefas como analogias de palavras, similaridade semântica e reconhecimento de entidades nomeadas em comparação a métodos anteriores baseados em janelas de contexto ou fatoração de matrizes (PENNINGTON; SOCHER; MANNING, 2014).

2.4.1 Modelos Baseados em *Transformer*

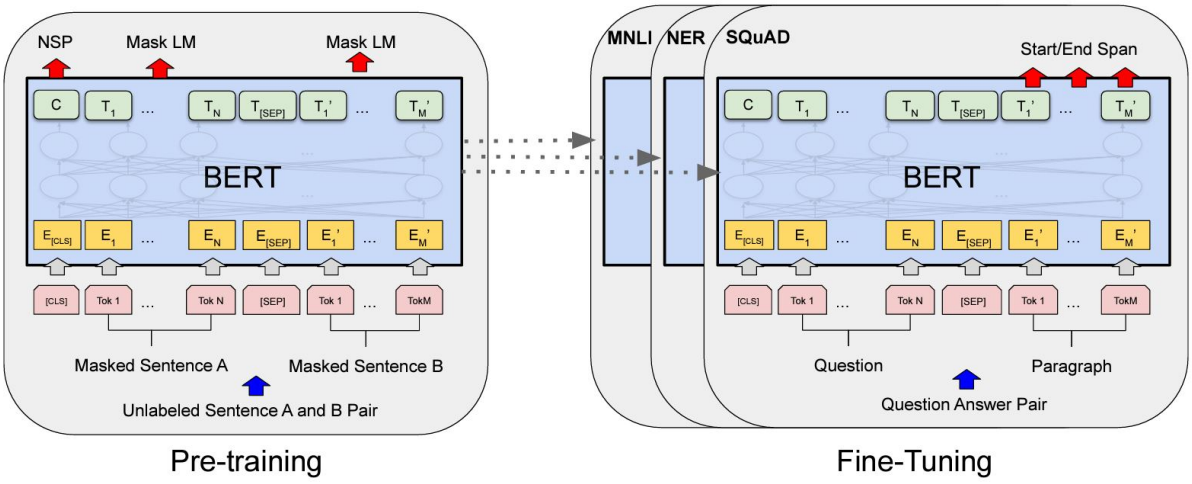
A estrutura *Transformer* é empregada com o objetivo de explorar a autocorrelação, ou autoatenção, dentro de uma extensa expressão linguística em relação a si mesma. Na aplicação da arquitetura *Transformer* à tradução entre expressões, a autoatenção ocorre em várias camadas tanto para as entradas quanto para as saídas, além da convencional correlação entre entrada e saída. Inovações no treinamento do modelo *Transformer* resultaram na desagregação da arquitetura, proporcionando um aumento significativo na precisão nas aplicações de PLN (FINGER, 2021).

2.4.1.1 *Bidirectional Encoder Representations from Transformers* (BERT)

BERT é o algoritmo de DL da *Google* para PLN, criado com o intuito de ajudar sistemas computacionais a entenderem a linguagem utilizada pelos seres humanos para se expressarem. BERT é considerado o primeiro modelo bem-sucedido de ajuste fino (*fine-tuning*), alcançando desempenho de última geração em um conjunto grande de tarefas, tanto em nível de frase quanto em nível de *token*, superando muitas arquiteturas específicas de tarefas (DEVLIN et al., 2019).

A Figura 8 apresenta a arquitetura geral do modelo BERT, destacando o fluxo de funcionamento desde o pré-treinamento até o ajuste fino (*fine-tuning*). O pré-treinamento envolve duas tarefas principais: Modelagem de Linguagem Mascarada ou *Masked Language Model* (MLM) e Previsão de Próxima Sentença ou *Next Sentence Prediction* (NSP). Essas tarefas permitem que o modelo compreenda tanto os contextos locais quanto globais nas sequências de texto (DEVLIN et al., 2019).

Figura 8 – Arquitetura do Modelo BERT

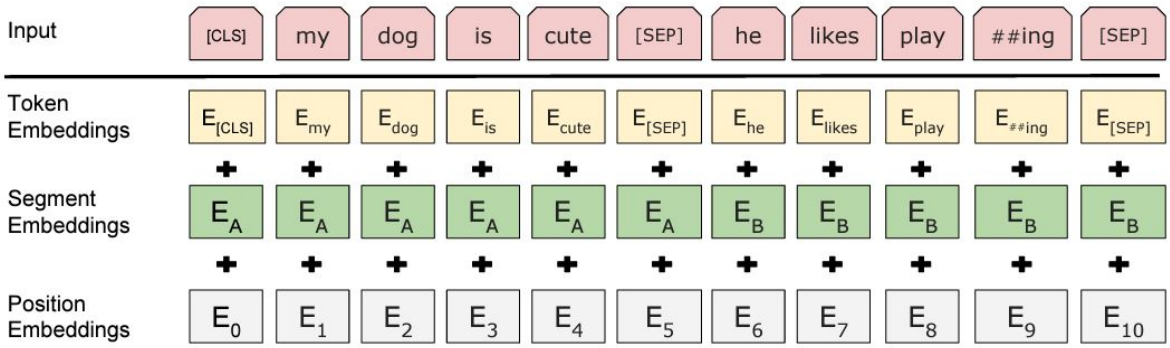


Fonte: Devlin et al. (2019)

Na arquitetura apresentada, o modelo utiliza *embeddings* de *tokens*, segmentos e posições para representar a entrada de texto. Esses *embeddings* são processados em camadas empilhadas de *Transformers*, que capturam o contexto bidirecional de cada *token* na sequência. Durante o *fine-tuning*, as camadas finais são ajustadas para tarefas específicas, permitindo que o modelo reutilize os parâmetros treinados no pré-treinamento.

A Figura 9 detalha como o BERT constrói os *embeddings* de entrada, essenciais para o processamento textual. Cada entrada é formada pela soma de três tipos de *embeddings*.

Figura 9 – Arquitetura *Embeddings* BERT



Fonte: Devlin et al. (2019)

i. ***Embeddings* de *Tokens***: Representam cada elemento textual na sequência de entrada. Palavras ou subpalavras são transformadas em vetores de dimensão fixa. O *token* especial "[CLS]" indica o início da sequência e é utilizado para gerar uma representação global em tarefas como classificação, enquanto o *token* "[SEP]" separa sentenças, sendo usado em tarefas como Previsão de Próxima Sentença.

ii. ***Embeddings* de Segmentos**: Diferenciam os *tokens* pertencentes às sentenças A e B em uma sequência. *Tokens* associados à primeira sentença recebem o *embedding* E_A, e os da segunda sentença recebem E_B. Esse mecanismo contribui diretamente para o desempenho de tarefas que envolvem múltiplas sentenças, como o NSP.

iii. ***Embeddings* de Posição**: Indicam a posição relativa de cada *token* na sequência de entrada. Como os *transformers* são invariantes à ordem dos *tokens*, esses *embeddings* introduzem uma noção de sequência, permitindo que o modelo reconheça a estrutura posicional dos dados.

Esses três tipos de *embeddings* são somados vetorialmente para cada *token*, formando uma única representação que é passada pelas camadas do *transformer*. Essa combinação permite que o BERT compreenda simultaneamente os significados dos *tokens*, a relação entre sentenças e a estrutura sequencial dos dados (DEVLIN et al., 2019).

2.4.1.2 *Robustly Optimized BERT Pretraining Approach* (RoBERTa)

O *Robustly Optimized BERT Approach* (RoBERTa), proposto por Liu et al. (2021b), introduz modificações no processo de pré-treinamento do modelo BERT. Uma das alterações realizadas é a remoção da tarefa de NSP, que no BERT era utilizada para treinar o modelo a identificar se uma sentença seguia logicamente outra, auxiliando na compreensão de relações entre sentenças. Outra modificação foi a ampliação do volume de dados utilizados no pré-treinamento, incorporando cinco grandes conjuntos de dados: o *BookCorpus* e a *English Wikipedia* (16 GB)[1]; o *CC-News* (76 GB)[2]; o *OpenWebText* (38 GB)[3]; e o *CC-Stories* (31 GB)[4]. Esses conjuntos de dados totalizam **mais de** 160 GB de texto (LIU et al., 2021b).

2.4.1.3 DistilBERT

O *Distilled Bidirectional Encoder Representations from Transformers* (DistilBERT), proposto por Sanh et al. (2019), foi desenvolvido como uma alternativa eficiente ao modelo BERT, buscando reduzir o custo computacional sem comprometer significativamente o desempenho. Esse modelo é baseado na técnica de distilação de conhecimento, onde um modelo menor e mais leve (*student model*) é treinado para imitar as previsões de um modelo maior e mais complexo (*teacher model*) (SANH et al., 2019).

O processo de distilação do DistilBERT combina três componentes principais na função de perda: a perda de distilação (L_{ce}), que força o *student model* a aprender as distribuições de probabilidade do *teacher model*; a perda de modelagem de linguagem mascarada (L_{mlm}),

[1] BookCorpus: <https://huggingface.co/datasets/bookcorpus>. English Wikipedia (dumps): <https://dumps.wikimedia.org/enwiki/latest/>.
[2] <https://commoncrawl.org/news-crawl/>
[3] <https://skylion007.github.io/OpenWebTextCorpus/>
[4] <https://huggingface.co/datasets/spacemanidol/cc-stories>.

que ajuda a preservar as capacidades de compreensão de linguagem do modelo; e a perda de distância cosseno (L_{cos}), que alinha as representações internas entre os dois modelos (SANH et al., 2019). Essas técnicas ajudam o DistilBERT capturar as propriedades mais importantes do BERT enquanto reduz sua complexidade computacional.

O DistilBERT apresenta 40% menos parâmetros do que o BERT, o que resulta em um modelo significativamente mais rápido, com redução de 60% no consumo de memória durante a inferência. Apesar dessas otimizações, ele preserva aproximadamente 97% da eficácia do BERT em *benchmarks* amplamente utilizados, como *General Language Understanding Evaluation* (GLUE) e *Stanford Question Answering Dataset* (SQuAD) (SANH et al., 2019).

2.4.1.4 BERTimbau

O *Bidirectional Encoder Representations from Transformers* Pretrained Model for Portuguese (BERTimbau), proposto por Souza, Nogueira e Lotufo (2019), representa o primeiro modelo de linguagem baseado na arquitetura BERT especificamente pré-treinado para a língua portuguesa. O modelo foi desenvolvido com o objetivo de preencher a lacuna de recursos linguísticos para tarefas de PLN em português, utilizando o corpus brWaC, composto por aproximadamente 2,68 bilhões de *tokens* provenientes de 3,53 milhões de documentos da *web*.

Foram disponibilizadas duas versões do modelo: O *BERT Pretrained Model for Portuguese base* (BERTimbau-base)[5], com 12 camadas de atenção e 110 milhões de parâmetros, e o *BERT Pretrained Model for Portuguese large* (BERTimbau-large)[6], com 24 camadas e 340 milhões de parâmetros. Ambas as versões foram treinadas com sequências de até 512 *tokens*. O modelo foi treinado preservando a distinção entre letras maiúsculas e minúsculas (modelo *cased*), ou seja, palavras como "Brasil" e "brasil" são tratadas como diferentes. Além disso, seu vocabulário é composto por 30.000 unidades linguísticas menores chamadas de *subwords*, que são fragmentos de palavras usados para representar termos raros ou complexos. Essa abordagem permite que o modelo compreenda melhor a estrutura das palavras em português, mesmo quando encontra termos desconhecidos.

2.4.1.5 XLNet

O XLNet, proposto por Yang et al. (2019), é uma extensão ao BERT que introduz um mecanismo de pré-treinamento autorregressivo permutado para superar limitações na modelagem bidirecional e na captura de dependências complexas no texto. Diferentemente do BERT, que utiliza modelagem bidirecional mascarada MLM, o XLNet explora todas as permutações possíveis da sequência de entrada, permitindo uma modelagem mais abrangente da distribuição conjunta dos *tokens*. Isso proporciona aprendizado tanto autorregressivo quanto bidirecional, combinando os benefícios de ambas as técnicas (YANG et al., 2019).

[5] BERTimbau base:<https://huggingface.co/neuralmind/bert-base-portuguese-cased>
[6] BERTimbau large:<https://huggingface.co/neuralmind/bert-large-portuguese-cased>.

O XLNet utiliza dois fluxos de atenção distintos, denominados *Two-Stream Attention*, que operam simultaneamente para capturar informações contextuais.

- ***Content Stream***: Responsável por processar o conteúdo contextual do texto, gerando representações enriquecidas para cada *token*.
- ***Query Stream***: Utilizado para prever *tokens* mascarados com base nas dependências contextuais aprendidas pelo modelo.

O XLNet preserva informações de longo alcance na sequência por meio do *segment recurrence mechanism* e do *relative position encoding*, ambos herdados do *Transformer*-XL.

2.4.1.6 *Text-to-Text Transfer Transformer* (T5)

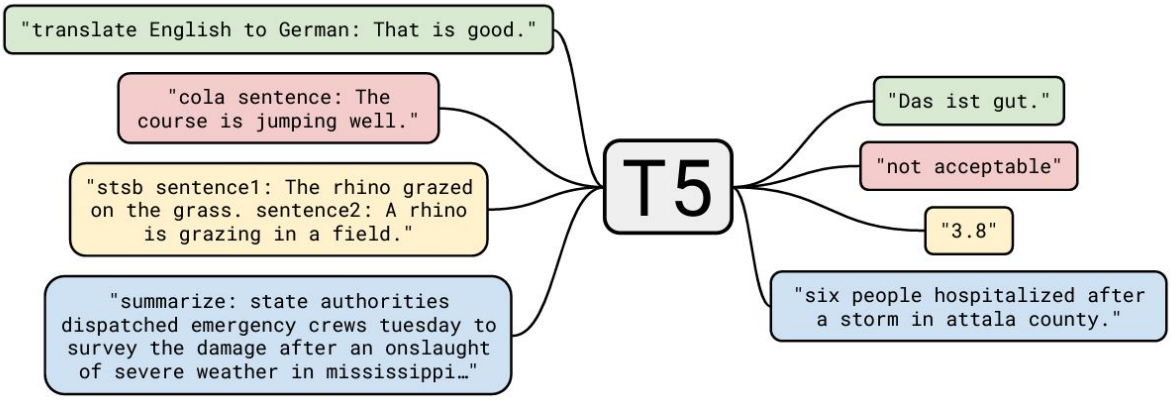
O modelo *Text-to-Text Transfer Transformer* (T5), proposto por Raffel et al. (2020), apresenta uma abordagem unificada para tarefas de PLN, onde todas as entradas e saídas são tratadas como problemas de transformação de texto. Essa estratégia permite o uso de um único modelo para diversas tarefas, como tradução, sumarização, classificação de texto e resposta a perguntas. No T5, a entrada é estruturada como um prefixo de tarefa seguido pelo texto de entrada, enquanto a saída corresponde ao resultado desejado da tarefa (RAFFEL et al., 2020).

O T5 foi pré-treinado utilizando o corpus *Colossal Clean Crawled Corpus* (C4) [7], uma base de dados limpa e de grande escala derivada do *Common Crawl*. O modelo foi treinado com o objetivo de reconstrução de texto mascarado (*span-corruption*), onde segmentos contínuos de *tokens* no texto de entrada são substituídos por marcadores especiais, e o modelo é treinado para prever esses segmentos (RAFFEL et al., 2020). A arquitetura do T5 é baseada no *Transformer*, seguindo o design *encoder-decoder*, mas adaptada para acomodar sua abordagem *text-to-text*, utilizando codificadores e decodificadores interligados por mecanismos de atenção.

A Figura 10 apresenta a abordagem do modelo T5 para o processamento de diferentes tarefas em PLN.

[7] <https://www.tensorflow.org/datasets/catalog/c4>

Figura 10 – Abordagem do Modelo T5



Fonte: Raffel et al. (2020)

Na Figura 10, as caixas à esquerda exemplificam as entradas textuais para diferentes tarefas:

- **Tradução**: *"translate English to German: That is good."* representa uma tarefa de tradução de idiomas.
- **Classificação de aceitabilidade**: *"cola sentence: The course is jumping well."* identifica a aceitação linguística de uma sentença.
- **Similaridade semântica**: *"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."* avalia o grau de similaridade entre duas sentenças.
- **Sumarização**: *"summarize: state authorities dispatched emergency crews. . . "* exemplifica uma tarefa de sumarização textual.

O núcleo central do modelo processa essas entradas por meio da arquitetura *Transformer*, utilizando representações aprendidas durante o pré-treinamento com o corpus C4. Durante o pré-treinamento, o T5 aprende a prever textos mascarados na tarefa de preenchimento de lacunas (*Span Corruption*), capturando nuances semânticas e contextuais.

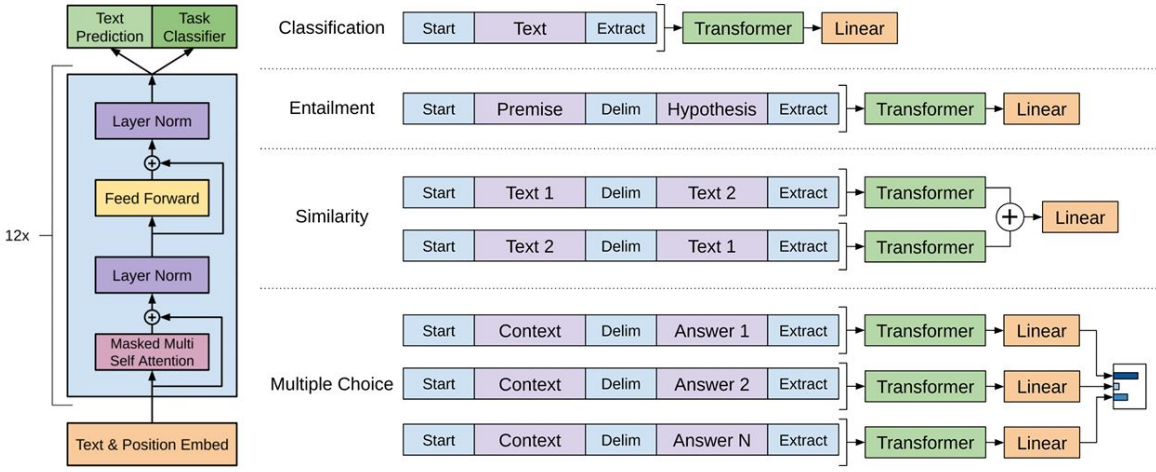
As caixas à direita mostram as saídas geradas pelo modelo para cada tarefa:

- **Tradução para o alemão**: *"Das ist gut."*
- **Classificação de aceitabilidade**: *"not acceptable."*
- **Similaridade semântica**: *"3.8"* (pontuação).
- **Sumarização**: *"six people hospitalized after a storm in attala county."*

2.4.1.7 *Generative Pre-trained Transformer* - GPT

O modelo *Generative Pre-trained Transformer* (GPT) foi introduzido pela empresa OpenAI no trabalho de Radford et al. (2018). O GPT baseia-se na arquitetura *Transformer Decoder*, utilizando camadas de autoatenção e mecanismos de aprendizado autoregressivo para modelagem de linguagem. A Figura 11 apresenta a arquitetura geral do GPT-1. O bloco à esquerda da Figura 11 representa a pilha de 12 camadas *Transformer* (*Decoder*), com destaque para o fluxo de *embeddings* de entrada, seguido pelas operações de autoatenção mascarada, redes *feedforward* e normalização.

Figura 11 – Arquitetura GPT



Fonte: Radford et al. (2018)

Na parte direita da figura são mostradas as diferentes estratégias utilizadas durante a fase de *fine-tuning* supervisionado para adaptação do modelo a tarefas específicas de PLN. Essas estratégias incluem:

- **Classificação de Sentenças:** A sequência de entrada é composta por um marcador inicial, o texto a ser classificado e um marcador de extração (*Extract*). A saída do *Transformer* é conectada a uma camada linear para classificação.
- **Inferência Textual (*Entailment*):** As sequências de premissa e hipótese são concatenadas com delimitadores, passando pelo *Transformer* antes da camada linear.
- **Similaridade Textual:** Duas sequências de texto são concatenadas com delimitadores e processadas, sendo suas representações finais combinadas antes da saída linear.
- **Escolha Múltipla (*Multiple Choice*):** O contexto é concatenado com cada uma das opções de resposta, e cada combinação passa separadamente pelo *Transformer*, seguido por uma camada linear. As saídas são então agregadas para gerar a decisão final.

O pré-treinamento do GPT-1 foi realizado com o corpus *BookCorpus*[8], contendo aproximadamente 7.000 livros de diversos gêneros, visando capturar dependências de longo alcance em linguagem natural. A configuração final incluiu 12 camadas *Transformer*, 768 unidades por estado oculto, 12 cabeças de atenção e um vocabulário de 40.000 *tokens*, utilizando codificação via *Byte Pair Encoding* (BPE).

Após a caracterização dos modelos empregados nesta pesquisa, torna-se necessário avaliar sua capacidade de realizar recomendações entre perfis de usuários que sejam mais similares possível. Para isso, adotam-se métricas que são utilizadas em tarefas de *matching*, em cenários de ranqueamento Top-k, nas quais se busca retornar, para cada entrada, os k itens mais relevantes segundo uma medida de similaridade. A seguir, são apresentadas as métricas utilizadas para mensurar o desempenho dos modelos.

2.5 Métricas de Validação

A avaliação do modelo proposto foi realizada com base em métricas comuns em tarefas de recomendação e ranqueamento Top-k. Para cada pesquisador, o sistema retorna uma lista ordenada de k *experts* com base na similaridade semântica entre os textos. A relevância das recomendações foi determinada por rótulos binários previamente definidos. As métricas empregadas incluem: *Precision*@k, que mede a proporção de itens relevantes recuperados, *Mean Reciprocal Rank* (MRR) e *Normalized Discounted Cumulative Gain* (nDCG), que consideram a posição dos itens relevantes no *ranking*, *Hit Ratio* (HR), que indica se ao menos um item relevante está entre os k primeiros e a Similaridade de Cosseno, utilizada como critério de ordenação das recomendações.

Essas métricas permitem mensurar diferentes aspectos do desempenho do sistema, incluindo cobertura, ordenação e relevância dos itens recomendados.

2.5.1 Métrica de (*Precision*@k)

A métrica de *Precision*@k avalia a proporção de itens relevantes entre aqueles que foram recomendados pelo sistema. Especificamente no contexto de sistemas de recomendação ou recuperação de informação, como o adotado nesta pesquisa, a *Precision*@k representa a fração de itens relevantes dentro dos k primeiros resultados retornados pelo modelo. A fórmula do *precision*@k é apresentada pela Equação 3 (LU et al., 2019):

$$Precision@k = \frac{\text{número de itens relevantes no Top-}k}{k} \tag{3}$$

Neste trabalho, considerou-se o valor $k = 3$, sendo avaliada a proporção de *experts* relevantes presentes entre os três primeiros *experts* recomendados para cada pesquisador. A *Precision*@k permite mensurar com que frequência o sistema retorna resultados relevantes nas

[8] <https://huggingface.co/datasets/bookcorpus>

primeiras posições do *ranking*. Quanto ao limiar de itens relevantes no Top-k, que pode ser interpretado como uma nota de corte para o desempenho aceitável, não há valor fixo, variando o número de itens relevantes de 0 a k, o que faz com que o valor da *Precision@k* varie de 0 a 1; para $k = 3$, isso corresponde a 0 (nenhum acerto), aproximadamente 0,3333 (um acerto), 0,6667 (dois acertos) e 1 (três acertos). O limiar ideal depende dos objetivos do estudo. Embora o valor de $k = 3$ tenha sido definido com base na necessidade de retornar os três primeiros itens, a inclusão específica de quais itens compõem o Top-k depende do algoritmo empregado, que seleciona e ordena os itens.

2.5.2 *Mean Reciprocal Rank* (MRR)

O MRR é uma métrica utilizada para avaliar sistemas de recuperação de informações e de perguntas e respostas ou *Question Answering* (QA). A métrica quantifica o desempenho de sistemas ranqueadores com foco na posição da primeira resposta correta encontrada para cada consulta (VOORHEES, 2001).

A principal característica do MRR é sua capacidade de avaliar o quão cedo um item relevante aparece no *ranking* retornado pelo sistema para cada consulta. A métrica calcula, para cada consulta, o inverso da posição da primeira ocorrência relevante e posteriormente, computa a média desses valores sobre todo o conjunto de consultas.

A fórmula do MRR é apresentada na Equação 4 (VOORHEES, 2001):

$$MRR = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{rank_i} \tag{4}$$

Onde:

- N representa o número total de consultas avaliadas;
- $rank_i$ corresponde à posição da primeira ocorrência de um item relevante no *ranking* gerado para a i-ésima consulta.

Para cada consulta i, o sistema de recuperação de informação gera uma lista ordenada de resultados. O valor de $rank_i$ é determinado pela posição, nessa lista, do primeiro item que atende aos critérios de relevância previamente definidos.

O valor do MRR varia de 0 a 1. Valores próximos de 1 indicam que o sistema consegue posicionar as respostas relevantes nas primeiras posições do *ranking* com alta frequência, enquanto valores próximos de 0 indicam que o sistema frequentemente posiciona as respostas corretas em posições mais baixas, ou falha em recuperá-las.

2.5.3 *Normalized Discounted Cumulative Gain* (nDCG)

O nDCG, proposto por Järvelin e Kekäläinen (2002), é uma métrica utilizada para avaliação da qualidade da ordenação de resultados em sistemas de recuperação da informação. O nDCG tem uma abordagem que considera a posição dos itens no *ranking*, e os diferentes níveis de relevância atribuídos a cada item, característica conhecida como *graded relevance*.

Diferentemente da métrica como o MRR, que consideram apenas a posição do primeiro item relevante, o nDCG avalia a ordenação global, atribuindo maior peso a documentos altamente relevantes posicionados nas primeiras posições e penalizando itens relevantes que aparecem em posições inferiores.

O cálculo inicia-se com a computação do nDCG, cuja fórmula é apresentada na Equação 5 (JÄRVELIN; KEKÄLÄINEN, 2002):

$$DCG_k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)} \tag{5}$$

Onde:

- k é o número de posições consideradas no *ranking* (Top-k);
- rel_i representa o grau de relevância do item na posição i, podendo assumir valores binários ou escalas graduais, conforme o critério de avaliação adotado. neste estudo adotamos relevância binária, obtida da coluna Resultado (*ground truth*) sem aplicação de limiar adicional.

Para tornar a métrica comparável entre diferentes consultas, o DCG obtido é normalizado pelo *Ideal DCG* (IDCG), que representa o ganho cumulativo máximo possível para aquela consulta, assumindo uma ordenação perfeita onde os documentos mais relevantes aparecem nas primeiras posições do *ranking*. Essa normalização tem como objetivo corrigir distorções causadas por variações no número de itens relevantes e no tamanho das listas de resultados entre diferentes consultas.

O cálculo do nDCG é formalizado na Equação 6 (JÄRVELIN; KEKÄLÄINEN, 2002):

$$nDCG_k = \frac{DCG_k}{IDCG_k} \tag{6}$$

Onde:

- DCG_k é o ganho cumulativo descontado obtido para a consulta, considerando os resultados efetivamente gerados pelo sistema até a posição k;

- $IDCG_k$ é o ganho cumulativo descontado ideal, calculado ordenando-se os documentos de forma que os itens mais relevantes apareçam nas primeiras posições até k, representando o melhor desempenho possível.

Essa divisão transforma o valor do DCG para uma escala normalizada entre 0 e 1, permitindo comparações diretas entre diferentes consultas, independentemente da quantidade ou da distribuição de documentos relevantes.

2.5.4 Hit Ratio@k (Taxa de Acerto no Top-k)

A HR é uma métrica utilizada na avaliação de sistemas de recomendação, especialmente em tarefas de recomendação do tipo Top-k. Esta métrica mensura a capacidade do sistema em apresentar ao menos um item relevante dentro das primeiras k posições do *ranking* gerado (CREMONESI; KOREN; TURRIN, 2010).

O HR está relacionado à efetividade da recomendação do ponto de vista do usuário final, focando na presença de itens relevantes nas posições superiores do *ranking*. O HR é particularmente relevante quando o objetivo é garantir que, para cada usuário ou consulta, ao menos uma recomendação efetivamente relevante esteja presente entre as primeiras colocações (KARYPIS, 2001).

O cálculo do HR é definido pela Equação 7 (CREMONESI; KOREN; TURRIN, 2010):

$$Hit@k = \frac{1}{N} \sum_{i=1}^{N} I_i \tag{7}$$

Onde:

- N representa o número total de consultas avaliadas;
- I_i é uma variável indicadora binária que assume o valor 1 se ao menos um item relevante estiver presente nas primeiras k posições do *ranking* da consulta i, e 0 caso contrário.

O valor do HR varia entre 0 e 1, sendo que valores mais próximos de 1 indicam maior capacidade do sistema em garantir pelo menos uma resposta relevante no Top-k. Essa métrica é de fácil interpretação e pode ser aplicada em análises comparativas de desempenho entre diferentes algoritmos de recomendação (CREMONESI; KOREN; TURRIN, 2010).

2.5.5 Similaridade por Cosseno

A similaridade por cosseno, ou *Cosine Similarity*, é uma métrica utilizada para medir a similaridade ou a proximidade entre dois vetores em um espaço vetorial. Essa métrica avalia

o valor do cosseno do ângulo formado entre os vetores, conforme apresentado na Equação 8 (PERREAULT-JENKINS, 2020):

$$\text{Cosine}(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} \tag{8}$$

Na equação, A e B representam dois vetores de características, por exemplo, vetores de *embeddings* extraídos de textos, documentos ou sentenças. Já θ representa o ângulo entre os vetores A e B no espaço vetorial.

Essa métrica é utilizada para comparar a similaridade entre documentos, independentemente de seu tamanho, pois considera apenas a orientação dos vetores, e não sua magnitude. Isso significa que dois documentos com conteúdos semelhantes, mesmo que com extensões diferentes, podem apresentar alta similaridade por cosseno.

Quando os vetores A e B apontam exatamente na mesma direção, a similaridade por cosseno assume valor 1, indicando alinhamento máximo entre os vetores. Quando os vetores são ortogonais (formam um ângulo de 90º), o valor da similaridade tende a 0, sugerindo ausência de correlação vetorial. Já quando apontam em direções opostas, o valor tende a -1, representando correlação negativa. No entanto, em aplicações de *matching* textual com *embeddings* positivos, os valores geralmente variam entre 0 e 1.

3

Revisão da Literatura

Esta revisão teve como objetivo mapear o estado da arte das técnicas de *matching* baseadas em PLN e Aprendizado de Máquina. Para auxiliar na condução e organização da revisão sistemática, utilizamos o *software Parsifal*[1]. Essa ferramenta foi empregada nas etapas de planejamento do protocolo, execução da busca e extração dos dados.

3.1 Perguntas de pesquisa

Esta revisão foi orientada por quatro perguntas de pesquisa, formuladas com o objetivo de consolidar o estado da arte das técnicas de *matching* baseadas em PLN, identificar lacunas metodológicas e discutir os principais desafios e oportunidades de pesquisa:

- **RQ1**: Quais técnicas e modelos de PLN têm sido utilizados para resolver problemas de *matching* ?
- **RQ2**: Quais modelos de linguagem pré-treinados são frequentemente adotados em tarefas de *matching*?
- **RQ3**: Quais métricas de desempenho são comumente utilizadas para avaliar algoritmos de *matching*?
- **RQ4**: Quais são os principais domínios de aplicação e os desafios na utilização de PLN para o *matching* ?

[1] <https://parsif.al/>

3.2 Estratégia de busca

A revisão da literatura foi realizada em bases de dados acadêmicas, incluindo *IEEE Xplore*, *Scopus*, *Web of Science*, *ScienceDirect* e *Google Scholar*. A escolha dessas bases foi motivada por sua ampla cobertura da literatura científica nas áreas de Ciência da Computação, IA e PLN.

A *string* de busca foi construída com base na metodologia *Population, Intervention, Comparison, Outcome e Context* (PICOC), incorporando sinônimos em inglês para maximizar a cobertura dos estudos relevantes, as consultas foram aplicadas aos campos Título, Resumo e *Abstract* e Texto. A Tabela 4 detalha os termos de busca utilizados para cada elemento do PICOC.

Tabela 4 – Termos de busca utilizados para cada elemento do modelo PICOC

Elemento do PICOC	**Termos de Busca**
População	
Intervenção	*"Matching algorithm"*, *"Matching Systems"*, *"Matching Models"*, *"Algorithmic Matching"*, *"Implicit Matching"*
Comparação	
Resultado	
Contexto	*"Natural Language Processing"*, *"NLP"*, *"Transformer models"*, *"Transformer technology"*, *"Embedding similarity"*, *"Word Embedding"*, *"Similarity Coefficient"*, *"Expertise Matching"*, *"Retrieval Systems"*, *"Performance Analysis"*
String principal	*("Matching algorithm" OR "Matching Systems" OR "Matching Models" OR "Algorithmic Matching" OR "Implicit Matching")* AND *("Natural Language Processing" OR "NLP" OR "Transformer models" OR "Embedding similarity" OR "Word Embedding" OR "Similarity Coefficient")* AND *("Expertise Matching" OR "Retrieval Systems" OR "Performance Analysis")*

* A *string* de busca foi adaptada para cada base de dados, de acordo com as suas especificidades sintáticas e operadores booleanos suportados.

3.2.1 Critérios de Seleção dos Estudos

A seleção dos estudos considerou os seguintes critérios: artigos revisados por pares, publicados entre 2018 e 2024, com o objetivo de garantir o foco nas abordagens mais recentes de *matching* baseado em PLN e ML. Também foram incluídos apenas estudos publicados em língua inglesa, devido à maior facilidade de acesso e compreensão por parte dos autores da revisão. Adicionalmente, foram selecionados apenas os trabalhos que descrevem e avaliam modelos de *matching* baseados em PLN e modelos pré-treinados que utilizem a arquitetura *Transformer*.

Foram excluídos estudos duplicados, artigos sem avaliação técnica ou que não apresentassem detalhes de implementação, além de trabalhos que descrevessem exclusivamente métodos

que seguem uma direção única, esquerda ou direita (unidirecionais) ou baseados apenas em regras.

3.2.2 Processo de Seleção dos Estudos e Extração dos Dados

O processo de seleção dos estudos ocorreu em duas etapas e foi conduzido por um único revisor. Após a aplicação dos critérios de seleção, foram analisados os títulos e resumos dos artigos. Os artigos que não atenderam aos critérios nessa fase inicial foram excluídos.

Na sequência, os textos completos dos artigos pré-selecionados foram analisados, sendo incluídos apenas os artigos que confirmaram a sua relevância para as perguntas de pesquisa e que apresentassem resultados para o campo do *matching* baseado em PLN.

A etapa de extração dos dados teve como objetivo categorizar as informações dos estudos selecionados. As informações extraídas foram organizadas de forma a responder às perguntas de pesquisa.

3.3 Consolidação dos achados

A Tabela 5 apresenta uma síntese dos artigos selecionados na revisão da literatura, destacando os respectivos autores, ano de publicação e os principais resultados obtidos em cada estudo analisado.

Tabela 5 – Síntese Resultados Revisão Literatura Algoritmos *Matching*

Autoria	**Principais Resultados**
Al-Faruk, Hussain e Shahriar (2018)	Propuseram o algoritmo *Back and Forth Matching* (BFM) para correspondência de padrões textuais, aplicando-o à análise de currículos universitários e requisitos de emprego. O método melhorou a eficiência da busca ao pré-processar índices e verificar padrões em posições otimizadas.
Hossain et al. (2018)	Desenvolveram um sistema para identificação de especialistas em pesquisa e formação de equipes multidisciplinares. A abordagem combinou PLN, aprendizado de máquina e visualização interativa para mapear áreas de pesquisa e conectar pesquisadores a chamadas de financiamento.
Khadilkar, Kulkarni e Bone (2018)	Desenvolveram um método para detecção de plágio baseado em grafos de conhecimento semântico, combinando *Named Entity Recognition* (NER) e *WordNet*. A abordagem superou técnicas tradicionais baseadas em *strings*, detectando plágio mesmo com substituição de sinônimos e mudanças na estrutura da frase.
Bernabé-Moreno et al. (2019)	Propuseram um método para padronização automática de habilidades profissionais, combinando GloVe, *Density-Based Spatial Clustering of Applications with Noise* (DBSCAN) e *t-distributed Stochastic Neighbor Embedding* (t-SNE) para agrupar competências. Nos testes com 17,5 mil descrições de vagas, consolidaram um conjunto padronizado de habilidades essenciais como *Python*, *java*, *Structured Query Language* (SQL) e R.

(Continua na próxima página)

Tabela 5 (Continuação da página anterior)

Autoria	Principais Resultados
Chakravarti et al. (2019)	Desenvolveram o *Computation Flow Orchestrator* (CFO), um *framework* baseado em microserviços para integração de PLN e recuperação de informações, otimizando a execução de modelos como BERT e *Elasticsearch* para ordenação de respostas.
Duan et al. (2019)	Desenvolveram um modelo para atribuição automática de revisores, utilizando BERT, CNN e *Bidirectional Long Short-Term Memory* (BiLSTM) para prever a similaridade entre revisores e manuscritos. O modelo melhorou a precisão da recomendação ao capturar relações semânticas entre títulos e resumos.
Ibrahimi et al. (2019)	Criaram um sistema interativo para exploração automática de vídeos jornalísticos, combinando *Residual Network* (ResNet), *Inflated 3D ConvNet* (I3D) e *Optical Character Recognition* (OCR) para reconhecimento multimodal e indexação semântica. O sistema eliminou a necessidade de anotação manual, permitindo buscas mais eficientes em vídeos.
Yilmaz (2019)	Implementaram um modelo para ranqueamento de relevância semântica em documentos longos, combinando *Best Matching* 25 (BM25) para recuperação inicial e BERT para reclassificação com base na relevância semântica de sentenças individuais, melhorando a precisão do ranqueamento.
Iyer et al. (2020)	Desenvolveram um sistema de aprendizado de máquina ponta a ponta para recomendação personalizada, utilizando LSTM para modelagem sequencial de preferências do usuário e incorporando *feedback* contínuo para ajustes em tempo real. A abordagem superou métodos tradicionais baseados apenas em filtragem colaborativa, alcançando ganhos significativos em métricas de engajamento e satisfação do usuário.
Jing et al. (2020)	Propuseram um sistema baseado em mapas de conhecimento ou (*knowledge mapping*) para gerenciamento e recuperação eficiente de informações em redes de distribuição elétrica. A abordagem utiliza grafos de conhecimento para representar semanticamente os dados, facilitando buscas complexas e melhorando em até 30% a eficiência na recuperação de dados operacionais críticos.
Perreault-Jenkins (2020)	Avaliaram diferentes abordagens de similaridade semântica, como similaridade de cosseno para melhorar recomendações em pequenos conjuntos de dados. O estudo mostrou que a combinação de TF-IDF com Similaridade Cosseno alcançou melhor desempenho geral na recomendação de candidatos em bases pequenas, com um aumento de precisão em torno de 12% comparado a métodos tradicionais.
Yang et al. (2020)	Desenvolveram um modelo híbrido que combina Entropia Diferencial para extração de características e redes neurais recorrentes bidirecionais BiLSTM para reconhecer emoções a partir de sinais cerebrais *Electroencephalography* (EEG). O modelo, aplicado ao conjunto *SJTU Emotion EEG Dataset IV* (SEED-IV), obteve precisão média superior a 90% na identificação de emoções como felicidade, tristeza, medo e neutralidade.
Akkasi e Moens (2021)	Analisaram técnicas para extração automática de relações causais em textos biomédicos usando redes neurais profundas (*Multiview* CNN, BiLSTM com atenção, *Graph* LSTM) e modelos pré-treinados (*Embeddings from Language Models* (ELMo), *Bidirectional Encoder Representations from Transformers for Biomedical Text Mining* (BioBERT), destacando benefícios para bases de conhecimento em medicina e biologia.

(Continua na próxima página)

Tabela 5 (Continuação da página anterior)

Autoria	Principais Resultados
Cui et al. (2021)	Propuseram o modelo híbrido *BiLSTM-Attention-CRF* (BAC) para reconhecimento automático de entidades nomeadas NER. O modelo estruturou textos utilizando *embeddings* contextuais do BERT, BiLSTM para captura bidirecional de contexto, atenção para dependências semânticas e BAC para etiquetagem das entidades nomeadas extraídas.
Duan, Weng e Gao (2021)	Propuseram o *Multi-Task Semantic Matching* (MTSM), um modelo multitarefa que combina RoBERTa e *Bidirectional Gated Recurrent Unit* (Bi-GRU) para *matching* semântico robusto em pequenos datasets ruidosos. O modelo usa similaridade de *Jaccard* para dividir pares textuais e Bi-GRU para capturar contexto, obtendo resultados superiores aos métodos baseados apenas em RoBERTa.
Elgammal et al. (2021)	Desenvolveram um sistema automatizado que combina PLN e aprendizado de máquina (vetorização e ranqueamento semântico) para correspondência entre currículos e vagas. O modelo atribuiu scores de compatibilidade e alcançou acurácia de 83% em testes com dados reais do Indeed.com.
Ferreira, Semedo e Magalhães (2021)	Propuseram os modelos *Convolutional Bidirectional Encoder Representations from Transformers* (ConvBERT) *Memory Network* (MemNet) para recuperação de informação baseada em contexto. Aplicaram o T5 para transformar consultas contextuais em independentes, melhorando significativamente o desempenho em recuperação semântica.
Liu et al. (2021a)	Propuseram o QuadrupletBERT, modelo baseado em BERT para recuperação de informação em larga escala. A abordagem introduziu uma arquitetura com quatro torres, considerando exemplos positivos, negativos fáceis e difíceis, otimizando separação semântica entre documentos relevantes e irrelevantes. Similaridade cosseno foi utilizada para medir distâncias entre *embeddings*, melhorando significativamente a recuperação da informação.
Mridha et al. (2021)	Apresentaram o L-Boost, modelo híbrido com BERT e LSTM para classificação de textos ofensivos em redes sociais nos idiomas bengali e Banglish. Utilizou AdaBoost para ajustar dinamicamente os pesos dos classificadores. Dados obtidos de *posts* do *Facebook* e *blogs*.
Santos e Lifschitz (2021)	Propuseram um modelo de busca semântica utilizando grafos de conhecimento hiper-relacionais para recuperar informações mais contextuais em bases de conhecimento. A abordagem transforma consultas em *embeddings* semânticos pré-calculados para recuperação eficiente.
Shan et al. (2021)	Propuseram o modelo *Global Weighted Self-Attention Network* (GLOW), que melhora a relevância da busca *web* combinando pesos globais do corpus, como BM25, com mecanismos de atenção dos *transformers*.
Su et al. (2021)	Desenvolveram um modelo de *Visual Storytelling* combinando BERT com LSTM, capaz de gerar narrativas coesas a partir de sequências de imagens. Avaliado no *Visual Storytelling Dataset* (VIST), superou modelos anteriores em métricas como *Bilingual Evaluation Understudy* (BLEU) e *Consensus-based Image Description Evaluation* (CIDEr).
Szarkowska et al. (2021)	Propuseram o *Knowledge Graph Bidirectional Encoder Representations from Transformers* (KG-BERT) para avaliação da qualidade de hierarquias em grafos de conhecimento, usando *embeddings* para representar relações semânticas e avaliando a precisão das relações hierárquicas por meio de aprendizado supervisionado.

(Continua na próxima página)

Tabela 5 (Continuação da página anterior)

Autoria	Principais Resultados
Tülümen et al. (2021)	Propuseram um modelo híbrido que combina aprendizado supervisionado e *embeddings* Word2Vec para realizar *matching* entre vagas de emprego e currículos. O modelo utiliza ponderação dinâmica das características dos candidatos, melhorando a correspondência baseada em experiência, habilidades e educação.
Zhang et al. (2021)	Desenvolveram o modelo *User-News Matching BERT* (UNBERT), que utiliza BERT para recomendar notícias personalizadas, resolvendo o problema de *cold-start* ao realizar *matching* em dois níveis: palavras (semântica entre notícias lidas e candidatas) e notícias (similaridade ampla).
Zhao (2021)	Apresentaram um *framework* baseado em modelos BERT e redes de atenção *Hierarchical Multi-Granularity Question-Aware Attention Network* (Himu-QAAN) e *BERT-based Question Answering Network* (BERT-QAnet) para seleção de respostas relevantes e identificação de perguntas duplicadas em plataformas de perguntas e respostas QA. Os resultados demonstraram maior precisão semântica nas tarefas realizadas.
Amalia et al. (2022)	Criaram o *Online Legal Consultation Bot* (OLCBot), *chatbot* integrado ao *Telegram*, para informar sobre a Lei de Criação de Empregos da Indonésia. Utilizou *Fuzzy Matching* e *Sastrawi Stemmer* para responder dúvidas dos usuários com base em documento oficial da lei.
García-Díaz e Valencia-García (2022)	Criaram o *Spanish SatiCorpus* 2021, *dataset* para identificar sátira em textos em espanhol, ajustando modelos como *model bert pre-trained on Spanish corpora* (BETO) e extraindo características linguísticas, com resultados expressivos na classificação entre textos satíricos e não satíricos.
Huang e Zhao (2022)	Desenvolveram modelo baseado em CNNs com *embeddings* TF-IDF, Word2Vec e ELMo para *matching* semântico multidimensional na descoberta de serviços *web*, aumentando a precisão na classificação dos serviços.
Jain, Miao e Kan (2022)	Desenvolveram modelo baseado em BERT para gerar *snippets* comparativos automáticos de opiniões sobre produtos, obtendo *Recall-Oriented Understudy for Gisting Evaluation – Longest Common Subsequence* (ROUGE-L) de 0,9876 em avaliações do *Amazon Reviews* com 3.269 produtos.
Khan et al. (2022)	Propuseram o DenseBert4Ret, modelo multimodal combinando *Densely Connected Convolutional Network* (DenseNet) e BERT via MLP para recuperação eficiente de imagens baseada em consultas textuais, usando *triplet loss* para otimizar similaridade entre imagens e textos.
Lei, Ji e Liu (2022)	Analisaram métodos de identificação de usuários em múltiplas redes sociais, empregando cinco abordagens baseadas em atributos de usuário, conteúdo, comportamento, topologia da rede e combinação desses atributos. Foram utilizados métodos como distância de *Levenshtein*, coeficiente de *Dice* e redes convolucionais para correspondência de perfis.
Martenot et al. (2022)	Desenvolveram o sistema *Literature Search Application* (LiSA), que combina BERT e NER, para identificar automaticamente eventos adversos relacionados a medicamentos e classificar sua gravidade de acordo com critérios da *European Medicines Agency* (EMA).
Meenakshi e Shanavas (2022)	Desenvolveram um modelo para identificação automática de perguntas duplicadas na plataforma Quora, combinando *embeddings* do BERT (128 dimensões), similaridade por cosseno e árvores de decisão em *bagging*. Alcançou 92,5% de acurácia, precisão de 88,2%, recall de 93,7% e F1-*score* de 90,9%, superando o modelo MaLSTM.

(Continua na próxima página)

Tabela 5 (Continuação da página anterior)

Autoria	Principais Resultados
Nair et al. (2022)	Propuseram o modelo *Contextualized Late Interaction over BERT with eXpansion* (ColBERT-X), uma extensão do *Contextualized Late Interaction over BERT* (ColBERT) utilizando *Cross-lingual Language Model - Robustly Optimized BERT Approach* (XLM-RoBERTa) e aprendizado por transferência para recuperação multilíngue de informações. O modelo superou métodos tradicionais como BM25, obtendo melhorias expressivas em *Mean Average Precision* (MAP) nas coleções multilíngues *Headlines Corpus 4* (HC4) (Chinês e Persa) e *Cross-Language Evaluation Forum* (CLEF) (Francês, Alemão, Italiano, Russo e Espanhol).
Sadri (2022)	Propôs um *framework* padronizado para avaliação de modelos de recuperação densa baseados em *transformers*, gerando *embeddings* e calculando similaridade por produto escalar, testado com métricas MRR@100 no *dataset Microsoft Machine Reading Comprehension* (MSMARCO).
Sridevi e Suganthi (2022)	Desenvolveram um sistema de *matching* automático entre currículos e vagas utilizando agrupamento de palavras em *clusters* (habilidades primárias, secundárias, adjetivos e advérbios), com similaridade calculada pelo coeficiente de *Jaccard*. O sistema foi validado em 14.906 currículos e 8 descrições de vagas do *Kaggle* e *LinkedIn*, classificando candidatos conforme relevância.
Wang (2022)	Desenvolveu um sistema de perguntas e respostas sobre *Coronavirus Disease 2019* (COVID-19) usando o BioBERT, melhorando significativamente a compreensão automática de textos biomédicos relacionados à pandemia.
Wang et al. (2022a)	Propuseram o CNN-BERT, um modelo híbrido que combina CNN e BERT para reconhecimento de alvos em imagens de radar *High-Resolution Range Profile* (HRRP), capturando características espaciais e temporais com eficiência superior aos métodos anteriores.
Wang et al. (2022b)	Propuseram o *Element-Focused Sentence-BERT* (EF-SBERT), modelo baseado em *Sentence-BERT* (SBERT) combinado com *FrameNet* para cálculo de similaridade semântica entre sentenças. O método usa um *ensemble* ajustado por hiperparâmetro, focando em sujeito, predicado e objeto das sentenças, aplicado em recuperação de informações, sistemas QA e mineração textual.
Agrawal e Shukla (2023)	Desenvolveram um método automatizado para geração de perguntas subjetivas e objetivas com o modelo T5 e biblioteca *Fast Text-to-Text Transfer Transformer* (FastT5), utilizando técnicas como desambiguação de sentidos *(Word Sense Disambiguation)*, enriquecimento com *ConceptNet* e extração contextual de palavras-chave através de grafos multipartidos.
Alruqi e Alzahrani (2023)	Desenvolveram um *chatbot* em árabe com *question-answering* extrativo, combinando modelos *transformer* pré-treinados (*Arabic Bidirectional Encoder Representations from Transformers* (AraBERT), *Contextualized Arabic Model Embedding Language BERT* (CAMeLBERT), *AraElectra fine-tuned on the Stanford Question Answering Dataset* (AraElectra-SQuAD) e *Arabic Efficiently Learning an Encoder that Classifies Token Replacements Accurately* (AraElectra)), ajustados em perguntas da Wikipédia árabe, obtendo respostas contextualizadas com desempenho competitivo frente a abordagens tradicionais.
Brisset et al. (2023)	Desenvolveram o *Similarity-based Flexible Tree Matching* (SFTM), algoritmo que combina TF-IDF e propagação entre nós para correspondência de árvores *web*, alcançando precisão de 89% e reduzindo o tempo de execução para 182 ms, desempenho superior ao *Tree Edit Distance* (TED).

(Continua na próxima página)

Tabela 5 (Continuação da página anterior)

Autoria	Principais Resultados
Chen et al. (2023)	Propuseram um modelo de codificação automática para diagnósticos médicos *International Classification of Diseases*, 10th Revision (ICD-10), combinando BERT e BiLSTM para capturar relações semânticas e contextuais, alcançando precisão de 0,979 e F1-*score* de 0,981, superando métodos tradicionais como *Deep Structured Semantic Model* (DSSM), *Convolutional Neural Network* (ConvNet) e *Enhanced Sequential Inference Model* (ESIM).
Dong (2023)	Apresentou o modelo *Word Hashing and Convolutional Representation* (WHCR) para *matching* de textos curtos em chinês, integrando *Word-order-preserving BERT* (WoBERT), *OpenHowNet* para similaridade semântica e *Regularized Dropout* (R-Drop) para robustez do modelo, abordando desafios como múltiplos significados e granularidade semântica.
Fujishiro, Otaki e Kawachi (2023)	Desenvolveram sistema semântico de recuperação de casos de negligência médica no Japão usando SBERT ajustado com modelos locais, *User–Topic–Hashtag BERT* (UTH-BERT) e *National Institute of Information and Communications Technology BERT* (NICT-BERT), obtendo melhor precisão comparado ao método tradicional Okapi BM25.
Hu et al. (2023)	Apresentaram uma abordagem *zero-shot* baseada em *Natural Language Inference* (NLI) combinada com *Generative Pre-trained Transformer 3.5* (GPT-3.5) e *Generative Pre-trained Transformer 4* (GPT-4) para classificar relações políticas em textos jornalísticos, sem necessidade de grandes quantidades de dados anotados. O modelo proposto *Zero-Shot Prediction* (ZSP) apresentou desempenho competitivo comparado a métodos supervisionados.
Hu, Zhang e Sun (2023)	Propuseram o modelo *Fine-Tuning BERT-Attention-BiLSTM* (FBAB), que combina *embeddings* BERT, mecanismo de atenção e BiLSTM, melhorando a precisão na correspondência de textos médicos curtos em chinês, destacando palavras-chave importantes e capturando contextos bidirecionais.
Jayasudha, Deepa e Devi (2023)	Desenvolveram um modelo híbrido para prever doenças a partir de sintomas, combinando *Biomedical Text Segmentation* (BiMM), conexão de termos e *embeddings* do BERT, obtendo precisão de 91,29% em um conjunto de 172 doenças e sintomas.
Kumar e Pati (2023)	Desenvolveram um modelo CNN-LSTM para reconhecimento automático de escrita manuscrita *offline Handwriting Recognition* (HWR), aplicando técnicas de filtragem de ruído e binarização adaptativa para melhorar a legibilidade dos textos digitalizados.
Li e He (2023)	Desenvolveram modelo baseado em redes neurais Siamese com Bi-GRU, mecanismo de atenção e CNN para correspondência de perguntas médicas online, utilizando Word2Vec e distância de *Manhattan*, obtendo acurácia de 97,24% e F1-*score* de 97,98%, superando *Attention-Based Convolutional Neural Network* (ABCNN) e ESIM em bases sobre medicina étnica e COVID-19.
Milošević e Thielemann (2023)	Realizaram comparação entre abordagens para extração de relações biomédicas, usando modelos baseados em regras, aprendizado de máquina e *transformers* DistilBERT, *Bidirectional Encoder Representations from Transformers pre-trained on PubMed abstracts and full-text articles* (PubMedBERT), T5 e *Scientific Text-to-Text Transfer Transformer based on T5 for biomedical and scientific tasks* (SciFive), com o objetivo de construir gráficos de conhecimento úteis na descoberta de medicamentos e expansão de indicações terapêuticas.

(Continua na próxima página)

Tabela 5 (Continuação da página anterior)

Autoria	Principais Resultados
Moravvej et al. (2022)	Desenvolveram um modelo para detecção de plágio combinando *embeddings* BERT, redes BiLSTM e algoritmo de evolução diferencial, utilizando focal *loss* para tratar desequilíbrio entre classes e mecanismo de atenção para calcular similaridades textuais.
Patil e Jadon (2023)	Propuseram um modelo que utiliza pré-processamento, técnicas de vetorização (TF-IDF, *One Hot Encoding*) e o modelo BERT para melhorar a similaridade semântica na detecção automática de relatórios duplicados de *bugs*, alcançando 70% de precisão em testes com dados dos projetos *Firefox*, *Eclipse*.
Su et al. (2025)	Propuseram o *Caseformer*, modelo pré-treinado baseado em *transformers* para recuperação de casos jurídicos, superando métodos tradicionais como TF-IDF e BM25 ao utilizar técnicas não supervisionadas para lidar com escassez de dados anotados, integrando codificadores duplos e *cross-encoder* para recuperar e reclassificar casos relevantes.
Tanberk et al. (2023)	Propuseram um *framework* utilizando OCR, BERT e RoBERTa para classificar e extrair informações de currículos, calculando similaridade por cosseno para ranquear candidatos. BERT obteve um F1-*score* de 93,98% na classificação de seções, e RoBERTa alcançou 89,63% em reconhecimento de entidades.
Uhlig et al. (2023)	Desenvolveram o *Deep Learning Approximate Matching* (DLAM), combinando *fuzzy hashing* com modelos *transformers* para detecção de *malware* em arquivos *JavaScript* e segredos corporativos em PDFs e documentos de escritório.
Vanetik e Kogan (2023)	Propuseram o *Vector Matching*, método que combina TF-IDF, n-grams, *embeddings* do BERT e técnicas de resumo automático para ranquear candidatos a vagas de TI. O método demonstrou melhor correlação de ranqueamento *(Krippendorff's alpha e Spearman)* comparado ao OKAPI BM25 e BERT-*rank*.
Wadud et al. (2023)	Desenvolveram o modelo *Deep Bidirectional Encoder Representations from Transformers* (Deep-BERT), que combina redes BERT com redes neurais convolucionais CNNs para classificar textos ofensivos em mídias sociais, alcançando precisão de 93,11% em inglês, 92,45% em bengali e 91,83% em contexto multilíngue, superando Word2Vec, TF-IDF, *Support Vector Machine* (SVM) e LSTM.
Askari et al. (2024)	Criaram um modelo híbrido *Cross-lingual Embedding-based BM25 Context-Aware Transformer* (CEBM25CAT), combinando BM25 com BERT por meio da inserção do *score* BM25 como *token* textual, resultando em melhoria de até 10% nas métricas de MAP e nDCG@10 nos MSMARCO e *Text REtrieval Conference Deep Learning 2019* (TREC DL'19) e *Text REtrieval Conference Deep Learning 2020* (TREC DL'20).
Li, Jiang e Zhang (2024)	Propuseram um modelo *Transformer-based Model for Tabular Data* (Tab-Transformer) integrado com técnicas de justiça algorítmica para prever cinco fatores de risco após transplante hepático (diabetes, rejeição, infecção, malignidade e complicações cardiovasculares), reduzindo disparidades entre grupos demográficos com alta precisão *Area Under the Receiver Operating Characteristic Curve* (AUROC).
Wu et al. (2024)	Desenvolveram o modelo *Bidirectional Gated Recurrent Unit with Semantic Fusion* (BiGRU-SF), combinando Bi-GRU e *embeddings* Word2Vec para reconhecimento automático de entidades nomeadas ner reveclacionadas a genes e fenótipos do arroz, obtendo um F1-*score* de 85%.

(Continua na próxima página)

Tabela 5 (Continuação da página anterior)

Autoria	Principais Resultados
Wang et al. (2024)	Propõem o *lightweight intrusion detection model* (BT-TPF), um modelo leve para detecção de intrusão em redes, que combina redes Siamese para redução de dimensionalidade e destilação de conhecimento com um *Vision Transformer* (ViT) como professor e um *PoolFormer* como aluno, reduzindo a complexidade em 90% sem perda de precisão.
Xiong et al. (2024)	Propuseram um método híbrido combinando redes *transformers* BERT, otimização por colônia de abelhas artificiais e aprendizado por reforço para detecção automática de plágio em textos, atingindo precisão de 94,5%, superando modelos anteriores como Siamese e *Context-Aware Recurrent Neural Network* (CA-RNN).
Zhang et al. (2024)	Propuseram o modelo *Tool for Evaluating Mobile Droid-based applications* (TEMdroid) que usa *embeddings* do BERT e redes Siamese para alinhamento semântico em migração automática de testes *Graphical User Interface* (GUI) entre aplicativos, melhorando em 17% sobre métodos anteriores como Android Test Matching (ATM) e *Context-Aware Test Case Recommendation Framework for Android* (Craftdroid), atingindo 76% de precisão geral.

3.4 Discussão da revisão da literatura

A busca nas bases de dados resultou na recuperação de 2.157 artigos, dos quais 134 foram excluídos por serem duplicados. Os 2.023 artigos foram para etapa de triagem, que foi conduzida por meio da leitura dos títulos e resumos, com o objetivo de excluir estudos que não abordavam diretamente o tema proposto. Como resultado dessa triagem inicial, 152 estudos foram selecionados para leitura completa. Após a leitura integral, 65 estudos foram considerados relevantes e incluídos na análise final. A Tabela 6 apresenta a distribuição dos estudos por base de dados.

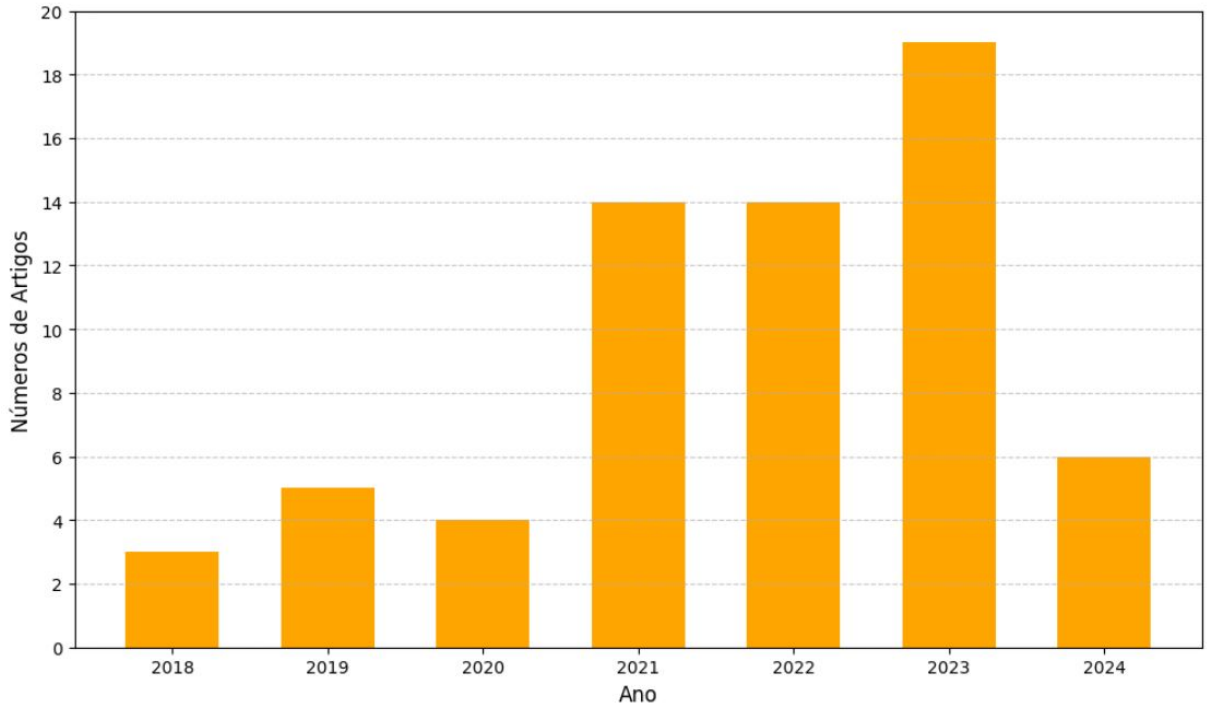
Tabela 6 – Artigos Selecionados e Aceitos por Base de Dados

base dados	Artigos Selecionados	Artigos Aceitos
Scopus	295	9
Google Scholar	965	27
ScienceDirect	666	16
IEEE Xplore	62	6
Web of Science	169	7
Total	**2157**	**65**

A análise temporal, ilustrada na Figura 12, evidencia uma tendência crescente na publicação de estudos voltados ao *matching* computacional a partir de 2020, com um pico expressivo em 2023. Esse comportamento acompanha o aumento da popularidade e da aplicação de modelos de linguagem baseados em arquiteturas *Transformer*, como BERT, RoBERTa e T5, que impulsionaram avanços significativos na área. Observa-se uma redução no número de artigos

indexados em 2024. No entanto, essa queda pode ser atribuída ao fato de que a coleta dos dados desta revisão foi realizada no segundo semestre de 2023, o que possivelmente limitou a inclusão de estudos que estavam em andamento, em fase final de submissão, publicação ou ainda em processo de indexação nas bases consultadas.

Figura 12 – Distribuição Final de Artigos por Ano



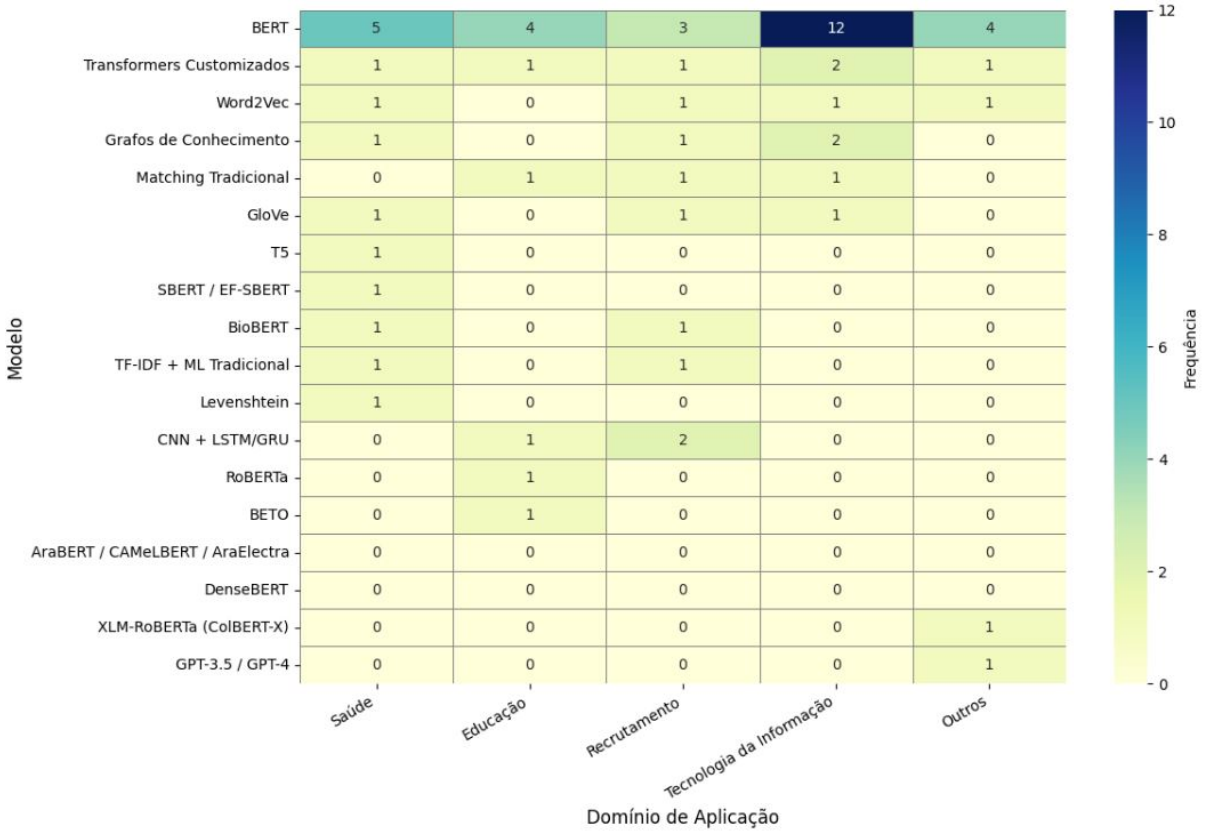
Fonte: Autoria própria (2025)

Para caracterizar o estado da arte, os estudos foram classificados com base em quatro critérios principais: domínios de aplicação, métodos utilizados, tipos de modelos pré-treinados e métricas de avaliação. A distribuição por domínio de aplicação, apresentada na Tabela 7, mostra uma maior concentração no setor de Tecnologia da Informação, seguida por Saúde e Recrutamento. Setores como Educação e Jurídico apresentam menor volume de estudos e maior diversidade metodológica, como evidenciado na Figura 13.

Tabela 7 – Distribuição dos Domínios de Aplicação entre os Estudos Selecionados

Domínio de Aplicação	**Numeros de Artigos**	**Porcentagem**
Saude	14	21%
Educação	10	15%
Recrutamento e Seleção	11	18%
Tecnologia da Informação	24	36%
Outros	6	9%
Total	65	100%

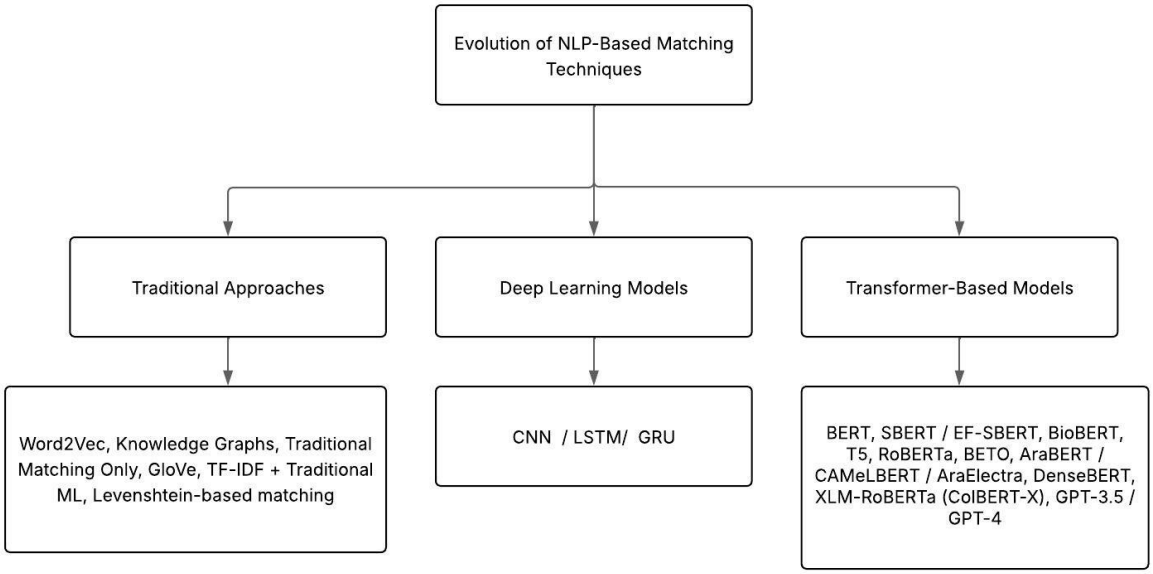
Figura 13 – Mapa de calor de modelos aplicados em domínios de aplicação



Fonte: Autoria própria (2025)

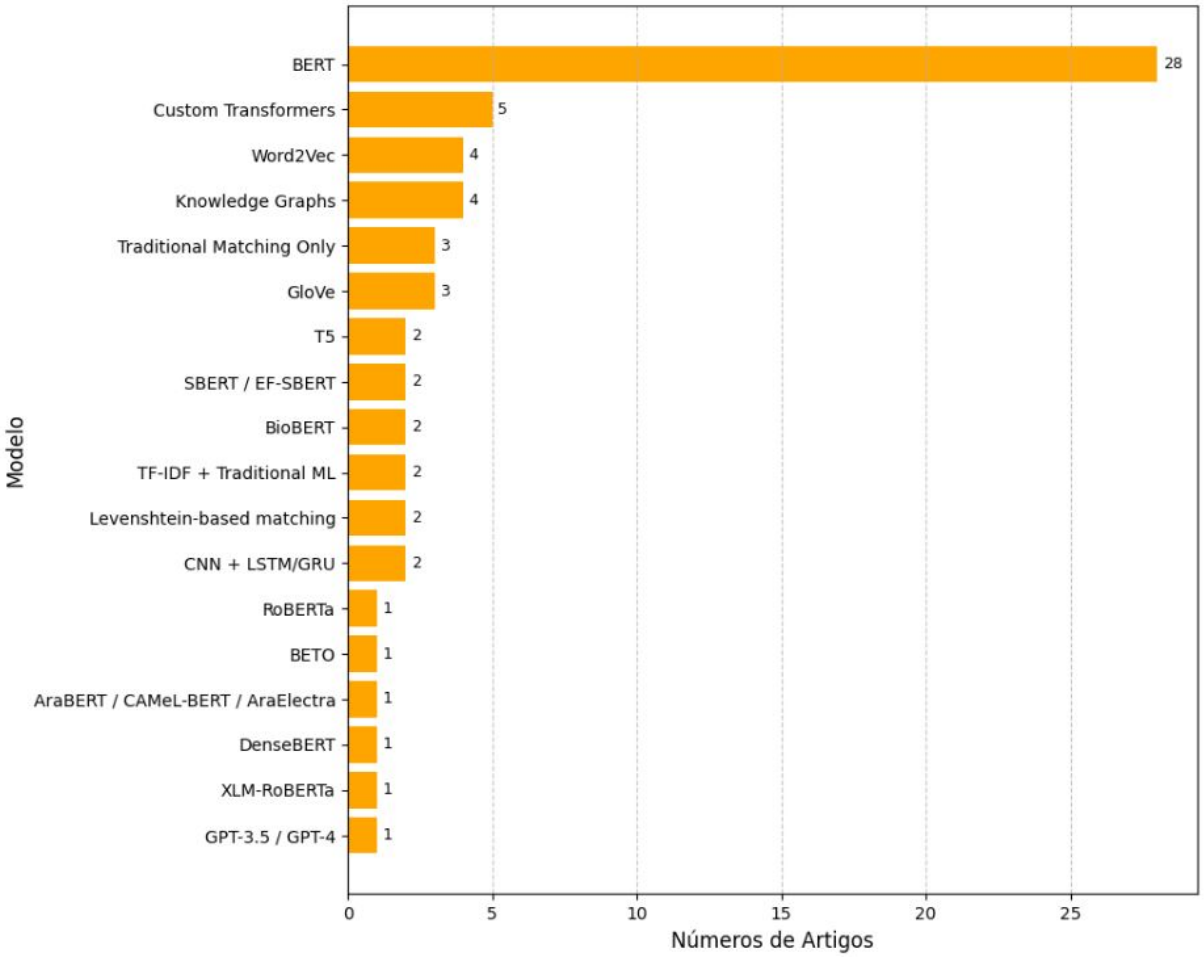
A Figura 14 mostra a evolução das técnicas utilizadas ao longo do tempo. Houve uma transição de métodos tradicionais, como TF-IDF e Word2Vec, para técnicas híbridas e, predominantemente, para modelos baseados em *Transformers*. O BERT foi utilizado em 28 estudos, sendo o modelo mais presente, conforme detalhado na Figura 15. Essa mudança aponta para uma preferência por arquiteturas que oferecem maior compreensão contextual, embora exijam maior consumo de recursos computacionais e apresentem desafios de interpretabilidade.

Figura 14 – Evolução das técnicas de matching baseadas em PLN



Fonte: Autoria própria (2025)

Figura 15 – Distribuição de modelos usados em estudos selecionados



Fonte: Autoria própria (2025)

A classificação das técnicas, apresentada na Tabela 8, reforça essa tendência. Embora técnicas tradicionais, como TF-IDF e Word2Vec, sejam empregadas em estudos mais antigos ou específicos, a maior parte das pesquisas recentes utiliza modelos baseados em *Transformers*, consolidando sua centralidade nas soluções de *matching* em PLN.

Tabela 8 – Modelos Utilizados nos Artigos Selecionados

Categoria de Modelo	**Número de Artigos**	**Artigos**
BERT	28	(XIONG et al., 2024), (ASKARI et al., 2024), (ZHANG et al., 2024), (WADUD et al., 2023), (PATIL; JADON, 2023), (VANETIK; KOGAN, 2023), (TANBERK et al., 2023), (JAYASUDHA; DEEPA; DEVI, 2023), (HU; ZHANG; SUN, 2023), (CHEN et al., 2023), (MORAVVEJ et al., 2022), (MEENAKSHI; SHANAVAS, 2022), (JAIN; MIAO; KAN, 2022), (SADRI, 2022), (MARTENOT et al., 2022), (WANG et al., 2022a), (SHAN et al., 2021), (ZHANG et al., 2021), (MRIDHA et al., 2021), (LIU et al., 2021a), (SU et al., 2021), (ZHAO, 2021), (CUI et al., 2021), (ELGAMMAL et al., 2021), (FERREIRA; SEMEDO; MAGALHãES, 2021), (DUAN et al., 2019), (CHAKRAVARTI et al., 2019), (YILMAZ, 2019)
Custom Transformers	5	(LI; JIANG; ZHANG, 2024), (WANG et al., 2024), (UHLIG et al., 2023), (DONG, 2023), (SU et al., 2025)
Word2Vec	4	(WU et al., 2024), (LI; HE, 2023), (HUANG; ZHAO, 2022), (TÜLÜMEN et al., 2021)
Knowledge Graphs	4	(SANTOS; LIFSCHITZ, 2021), (SZARKOWSKA et al., 2021), (JING et al., 2020), (KHADILKAR; KULKARNI; BONE, 2018)
Traditional Matching Only	3	(IYER et al., 2020), (AL-FARUK; HUSSAIN; SHAHRIAR, 2018), (HOSSAIN et al., 2018)
GloVe	3	(BRISSET et al., 2023), (PERREAULT-JENKINS, 2020), (BERNABÉ-MORENO et al., 2019)
T5	2	(MILOŠEVIĆ; THIELEMANN, 2023), (AGRAWAL; SHUKLA, 2023)
SBERT / EF-SBERT	2	(FUJISHIRO; OTAKI; KAWACHI, 2023), (WANG et al., 2022b)
BioBERT	2	(WANG, 2022), (AKKASI; MOENS, 2021)
TF-IDF + Traditional ML	2	(SRIDEVI; SUGANTHI, 2022), (IBRAHIMI et al., 2019)
Levenshtein-based matching	2	(LEI; JI; LIU, 2022), (AMALIA et al., 2022)
CNN + LSTM/GRU	2	(KUMAR; PATI, 2023), (YANG et al., 2020)
RoBERTa	1	(DUAN; WENG; GAO, 2021)
BETO	1	(GARCÍA-DÍAZ; VALENCIA-GARCÍA, 2022)
AraBERT / CAMeL-BERT / AraElectra	1	(ALRUQI; ALZAHRANI, 2023)
DenseBERT	1	(KHAN et al., 2022)
XLM-RoBERTa (ColBERT-X)	1	(NAIR et al., 2022)
GPT-3.5 / GPT-4	1	(HU et al., 2023)

3.4.1 Técnicas de PLN e Modelos Utilizados em *Matching*

Dos 65 artigos incluídos nesta revisão sistemática, 43 estudos apresentaram contribuições relevantes para a discussão sobre as técnicas e modelos de PLN utilizados no *matching*. A análise desses trabalhos evidenciou uma evolução metodológica ao longo dos anos, refletindo avanços tecnológicos e mudanças nas demandas das aplicações. Essa evolução ocorreu no sentido de uma transição gradual de abordagens baseadas em regras para métodos estatísticos e de aprendizado de máquina, incorporando probabilidades e vetores de palavras como Word2Vec e, mais recentemente, para técnicas de aprendizado profundo com redes neurais, como RNNs, LSTMs e modelos com arquitetura *transformer* permitindo *matching* semântico contextual e bidirecional. As demandas das aplicações, por sua vez, mudaram para maior ênfase em escalabilidade para grandes volumes de dados e ambiguidades linguísticas, suporte multilíngue e integração com tarefas multimodais, impulsionadas pelo crescimento de dados não estruturados e aplicações em tempo real.

Inicialmente, os estudos concentraram-se em técnicas baseadas em representações lexicais, como TF-IDF e *n-grams*, combinadas com medidas de similaridade tradicionais, como distância de cosseno, distância de Manhattan e coeficiente de *Jaccard* (PATIL; JADON, 2023), (VANETIK; KOGAN, 2023), (BRISSET et al., 2023), (HUANG; ZHAO, 2022), (TÜLÜMEN et al., 2021), (PERREAULT-JENKINS, 2020). Embora essas técnicas apresentem baixo custo computacional, sua limitação na captura de relações semânticas profundas e de contexto reduziu sua eficácia em aplicações mais complexas. Com o avanço do PLN, a utilização de *word embeddings* estáticos, como Word2Vec, GloVe e *FastText*, passou a ser explorada para melhorar a representação semântica (WU et al., 2024; LI; HE, 2023; HUANG; ZHAO, 2022; TÜLÜMEN et al., 2021; ZHAO, 2021; BERNABÉ-MORENO et al., 2019; IBRAHIMI et al., 2019), ainda que tais modelos também não considerem as variações contextuais de significado.

A partir de 2020, observou-se uma predominância crescente de métodos baseados em arquiteturas *Transformer*, com destaque para o BERT e suas variantes, como RoBERTa, DistilBERT, *Scientific Bidirectional Encoder Representations from Transformers* (SciBERT), BioBERT, AraBERT e KG-BERT (DEVLIN et al., 2019; WADUD et al., 2023; ASKARI et al., 2024; TANBERK et al., 2023; HU; ZHANG; SUN, 2023; FUJISHIRO; OTAKI; KAWACHI, 2023; ALRUQI; ALZAHRANI, 2023; CHEN et al., 2023; MILOŠEVIĆ; THIELEMANN, 2023; MORAVVEJ et al., 2022; SU et al., 2025; NAIR et al., 2022; JAIN; MIAO; KAN, 2022; MARTENOT et al., 2022; WANG, 2022; SHAN et al., 2021; ZHANG et al., 2021; DUAN; WENG; GAO, 2021; LIU et al., 2021a; ZHAO, 2021; FERREIRA; SEMEDO; MAGALHãES, 2021; SZARKOWSKA et al., 2021; DUAN et al., 2019; YILMAZ, 2019). Esses modelos possibilitaram a geração de *embeddings* contextuais, elevando a capacidade dos sistemas em capturar nuances semânticas e relações complexas entre consultas de usuários e perfis de *experts*.

Também foi observada uma tendência ao desenvolvimento de arquiteturas híbridas, integrando *Transformers* com outras redes, como CNNs, BiLSTM, GRU e redes Siamesas, com

o objetivo de melhorar a representatividade semântica e a eficiência computacional (WADUD et al., 2023; ZHANG et al., 2024; HU; ZHANG; SUN, 2023; CHEN et al., 2023; LI; HE, 2023; MORAVVEJ et al., 2022; WANG et al., 2022a; DUAN; WENG; GAO, 2021; CUI et al., 2021; DUAN et al., 2019). Exemplos incluem a combinação de BERT com CNNs (WADUD et al., 2023), com BiLSTM (HU; ZHANG; SUN, 2023; CHEN et al., 2023; MORAVVEJ et al., 2022; DUAN; WENG; GAO, 2021; DUAN et al., 2019) e com redes Siamesas (ZHANG et al., 2024; LI; HE, 2023; MORAVVEJ et al., 2022).

Técnicas multitarefa e mecanismos de atenção também foram empregados para fortalecer a capacidade de modelagem contextual (HU; ZHANG; SUN, 2023; DUAN; WENG; GAO, 2021; DUAN et al., 2019). Alguns estudos incorporaram estratégias de pré-processamento avançado, incluindo reconhecimento de entidades nomeadas NER, desambiguação de sentido de palavras, expansão semântica com bases de conhecimento e uso de grafos semânticos (KHADILKAR; KULKARNI; BONE, 2018; TANBERK et al., 2023; AGRAWAL; SHUKLA, 2023; BERNABÉ-MORENO et al., 2019). Além disso, técnicas de otimização, como aprendizado por reforço (XIONG et al., 2024), evolução diferencial (MORAVVEJ et al., 2022) e distilação de conhecimento que transfere o conhecimento de um modelo professor para um aluno mais leve (WANG et al., 2024), foram exploradas para reduzir custos computacionais e melhorar a eficiência.

Apesar dos avanços, os estudos analisados destacam limitações recorrentes, como o elevado custo computacional associado aos modelos baseados em *Transformers* (WANG et al., 2024; ASKARI et al., 2024; MORAVVEJ et al., 2022; NAIR et al., 2022), a baixa interpretabilidade dos modelos mais complexos (MORAVVEJ et al., 2022; SHAN et al., 2021; DUAN et al., 2019) a escassez de dados rotulados em domínios específicos, tais como saúde, jurídico e educação (FUJISHIRO; OTAKI; KAWACHI, 2023; CHEN et al., 2023; MILOŠEVIĆ; THIELEMANN, 2023). Além disso, foram identificadas lacunas relacionadas à generalização dos modelos entre diferentes domínios e à adaptação a cenários com restrição de recursos (HU et al., 2023; NAIR et al., 2022; ZHANG et al., 2021). Tais limitações reforçam a necessidade de pesquisas futuras voltadas ao desenvolvimento de modelos mais eficientes, interpretáveis e adaptáveis, capazes de operar de forma robusta mesmo em contextos com limitações de dados e infraestrutura computacional.

3.4.2 Modelos de Linguagem Pré-Treinados Frequentemente Adotados em Tarefas de *Matching*

Dos 65 artigos incluídos nesta revisão sistemática, 36 estudos apresentaram dados específicos sobre o uso de *pre-trained language model* (PLM) aplicados ao *matching*. A análise desses trabalhos revela que o BERT e suas variantes dominam o cenário, configurando-se como a principal escolha de arquitetura base para tarefas de representação semântica e cálculo de similaridade textual.

O BERT aparece como o modelo mais recorrente, sendo empregado tanto na sua forma original quanto em versões adaptadas a domínios específicos ou com otimizações arquiteturais. Os estudos (DEVLIN et al., 2019; PATIL; JADON, 2023; WADUD et al., 2023; ASKARI et al., 2024; ZHANG et al., 2024; TANBERK et al., 2023; JAYASUDHA; DEEPA; DEVI, 2023; HU; ZHANG; SUN, 2023; FUJISHIRO; OTAKI; KAWACHI, 2023; CHEN et al., 2023; MILOŠEVIĆ; THIELEMANN, 2023; MORAVVEJ et al., 2022; SU et al., 2025; NAIR et al., 2022; MEENAKSHI; SHANAVAS, 2022; JAIN; MIAO; KAN, 2022; MARTENOT et al., 2022; WANG, 2022; SHAN et al., 2021; ZHANG et al., 2021; DUAN; WENG; GAO, 2021; LIU et al., 2021a; ZHAO, 2021; FERREIRA; SEMEDO; MAGALHãES, 2021; SZARKOWSKA et al., 2021; IYER et al., 2020; YILMAZ, 2019) apresentam trabalhos que utilizaram o BERT para geração de *embeddings* contextuais, classificação, recuperação de informações ou construção de sistemas de *matching* especializados. Algumas dessas implementações utilizaram o BERT puro, enquanto outras realizaram *fine-tuning* supervisionado, adaptando o modelo aos respectivos domínios de aplicação.

Além do BERT, outras variantes especializadas foram identificadas. Modelos como RoBERTa (TANBERK et al., 2023; MILOŠEVIĆ; THIELEMANN, 2023; DUAN; WENG; GAO, 2021), DistilBERT (MILOŠEVIĆ; THIELEMANN, 2023), BioBERT (WANG, 2022; AKKASI; MOENS, 2021), SciBERT (MILOŠEVIĆ; THIELEMANN, 2023; AKKASI; MOENS, 2021), AraBERT e CAMeLBERT (ALRUQI; ALZAHRANI, 2023), XLM-RoBERTa (NAIR et al., 2022), SBERT (FUJISHIRO; OTAKI; KAWACHI, 2023), KG-BERT (SZARKOWSKA et al., 2021) e T5 (MILOŠEVIĆ; THIELEMANN, 2023; AGRAWAL; SHUKLA, 2023; FERREIRA; SEMEDO; MAGALHãES, 2021) também foram empregados para enfrentar desafios específicos de domínio, idioma ou tamanho de conjunto de dados.

O uso de variantes como BioBERT e SciBERT, por exemplo, destacou-se em tarefas do domínio biomédico, onde o vocabulário técnico demanda representações linguísticas especializadas (MILOŠEVIĆ; THIELEMANN, 2023; WANG, 2022; AKKASI; MOENS, 2021). Já o AraBERT e o CAMeLBERT mostraram eficácia em aplicações em língua árabe, como chatbots e sistemas de QA (ALRUQI; ALZAHRANI, 2023). Modelos como o XLM-RoBERTa e o ColBERT-X (NAIR et al., 2022) demonstraram bom desempenho em cenários multilíngues, particularmente em tarefas de recuperação densa de informações.

Outra tendência observada foi a adoção de modelos multitarefa, como o MTSM, que utiliza o RoBERTa como codificador compartilhado (DUAN; WENG; GAO, 2021), e o uso de SBERT ajustado com variantes locais, como o UTH-BERT e o NICT-BERT, para busca semântica em contexto médico (FUJISHIRO; OTAKI; KAWACHI, 2023).

Além dos *Transformers*, alguns poucos estudos continuaram explorando *word embeddings* estáticos, como Word2Vec (WU et al., 2024; LI; HE, 2023; HUANG; ZHAO, 2022; TÜLÜMEN et al., 2021), GloVe (ZHAO, 2021; DUAN et al., 2019) e ELMo (HUANG; ZHAO, 2022), geralmente em arquiteturas híbridas ou como *baseline* para comparação de desempenho. No

entanto, o desempenho inferior desses modelos frente aos *embeddings* contextuais tem sido consistentemente reportado.

Quanto às limitações, destaca-se o elevado custo computacional dos modelos *Transformer*, que restringe sua aplicação em contextos com infraestrutura limitada (WANG et al., 2024; ASKARI et al., 2024; MORAVVEJ et al., 2022; NAIR et al., 2022). Além disso, a baixa interpretabilidade das decisões geradas por esses modelos permanece como uma barreira para adoção em domínios sensíveis, como saúde e jurídico (MORAVVEJ et al., 2022; SHAN et al., 2021; DUAN et al., 2019). Outro aspecto crítico é a necessidade de grande volume de dados rotulados para o *fine-tuning*, o que representa uma dificuldade adicional em áreas com escassez de bases de dados anotadas (FUJISHIRO; OTAKI; KAWACHI, 2023; CHEN et al., 2023; MILOŠEVIĆ; THIELEMANN, 2023).

Como *gaps* a serem explorados em pesquisas futuras, destaca-se a demanda por modelos de linguagem especializados para domínios com vocabulário técnico específico, o desenvolvimento de estratégias e compressão de modelos para ambientes com restrições de *hardware* e a criação de *frameworks* explicáveis que permitam maior transparência nas recomendações de *matching*.

Em síntese, a adoção de modelos de linguagem pré-treinados baseados em *Transformers*, com forte predominância do BERT e suas variantes, tem-se consolidado como a principal tendência para o desenvolvimento de sistemas de *matching*, com benefícios claros em termos de compreensão semântica, mas também com desafios técnicos e operacionais ainda não totalmente superados.

3.4.3 Métricas de Desempenho Comumente Utilizadas para Avaliar Algoritmos de *Matching*

Dos 65 artigos incluídos nesta revisão sistemática, 18 estudos apresentaram informações específicas e relevantes sobre as métricas de desempenho utilizadas na avaliação de algoritmos de *matching*. A análise desses trabalhos evidencia uma forte predominância de métricas tradicionalmente empregadas nas áreas de Recuperação da Informação e Aprendizado de Máquina, refletindo as diferentes naturezas das tarefas de *matching* abordadas.

As métricas mais utilizadas foram Precisão, *Recall* e F1-*Score*, aplicadas principalmente em tarefas de classificação binária ou multi-classe, onde o objetivo era discriminar entre pares relevantes e não relevantes. Essas métricas foram reportadas nos estudos apresentados nos artigos (PATIL; JADON, 2023; WADUD et al., 2023; TANBERK et al., 2023; JAYASUDHA; DEEPA; DEVI, 2023; HU; ZHANG; SUN, 2023; CHEN et al., 2023; LI; HE, 2023; MORAVVEJ et al., 2022; MEENAKSHI; SHANAVAS, 2022; JAIN; MIAO; KAN, 2022), sendo o F1-*Score* particularmente valorizado por oferecer um equilíbrio entre os erros de omissão e de comissão, especialmente em cenários com classes desbalanceadas.

Nas tarefas de ranqueamento e recuperação de informações, as métricas mais utilizadas

foram nDCG@k, MAP e MRR@k, que avaliam a qualidade da ordenação dos resultados apresentados aos usuários. Estas métricas foram empregadas nos artigos (ASKARI et al., 2024; VANETIK; KOGAN, 2023; NAIR et al., 2022; SADRI, 2022; YILMAZ, 2019), principalmente em cenários de recuperação de documentos, recomendação de especialistas ou ranqueamento de candidatos para vagas.

Outras métricas de concordância entre *rankings*, como o Alfa de *Krippendorff* e o Coeficiente de *Spearman*, foram relatadas no artigo (VANETIK; KOGAN, 2023), sendo utilizadas para medir a correlação entre os *rankings* gerados pelos modelos e os *rankings* fornecidos por avaliadores humanos. Em termos de avaliação da similaridade semântica, medidas como distância de cosseno e distância de *Manhattan* foram mencionadas como componentes intermediários do processo de *matching*, mas não como métricas finais de avaliação. As similaridades semânticas são usadas como funções internas de pontuação entre os *embeddings* isto é, calculam o *score*, que induz a ordenação no processo de *matching* (ZHANG et al., 2024; LI; HE, 2023; HUANG; ZHAO, 2022).

Um aspecto relevante identificado foi a diversidade e a falta de padronização nas escolhas métricas entre os estudos. Enquanto alguns trabalhos relataram exclusivamente a acurácia (PATIL; JADON, 2023; WADUD et al., 2023; JAYASUDHA; DEEPA; DEVI, 2023; LI; HE, 2023), outros apresentaram um conjunto mais robusto de métricas, incluindo medidas específicas de ranqueamento (ASKARI et al., 2024; VANETIK; KOGAN, 2023; NAIR et al., 2022; SADRI, 2022). Em certos casos, os autores combinaram métricas de classificação com métricas de ranqueamento para fornecer uma avaliação mais ampla, como observado nos artigos (YILMAZ, 2019).

As principais limitações encontradas na literatura incluem a ausência de reportes de intervalos de confiança ou testes de significância estatística, o que compromete a comparação entre modelos (PATIL; JADON, 2023; WADUD et al., 2023; VANETIK; KOGAN, 2023; LI; HE, 2023; MORAVVEJ et al., 2022). Além disso, observou-se que poucos estudos realizaram avaliações de desempenho em cenários reais de aplicação, restringindo-se a testes em *datasets* artificiais ou publicamente disponíveis, o que pode não refletir adequadamente o comportamento do sistema em ambientes produtivos (ASKARI et al., 2024; NAIR et al., 2022; SADRI, 2022).

Como *gaps* de pesquisa, destaca-se a carência de métricas específicas para o problema de *matching*, capazes de capturar dimensões como a adequação contextual, a satisfação do usuário final e a qualidade semântica do pareamento. Também se identificou a necessidade de maior uniformidade na escolha e no reporte de métricas, de forma a permitir comparações mais justas e replicáveis entre os diferentes métodos propostos.

Em síntese, apesar da ampla adoção de métricas clássicas como F1-*Score*, MAP, MRR@k e nDCG@k, a literatura carece de um padrão metodológico consolidado para a avaliação de algoritmos de *matching*, além de carecer de métricas mais alinhadas aos desafios específicos desse tipo de tarefa.

3.4.4 Principais Domínios de Aplicação e Desafios na Utilização de PLN para o *Matching*

Dos 65 artigos incluídos nesta revisão sistemática, 26 apresentaram informações relevantes sobre os domínios de aplicação e os desafios associados à utilização de PLN para o *matching*. A análise dos dados evidencia uma concentração significativa de pesquisas em três principais áreas: Tecnologia da Informação, Saúde e Recrutamento de Recursos Humanos, seguidas por contribuições mais pontuais nos domínios educacional e jurídico.

No domínio da Tecnologia da Informação, ênfase foi dada ao desenvolvimento de sistemas de recomendação e recuperação de informações, especialmente no contexto de repositórios de *software* e triagem de *bugs* (PATIL; JADON, 2023; ASKARI et al., 2024; VANETIK; KOGAN, 2023; NAIR et al., 2022; SADRI, 2022; SHAN et al., 2021). As tarefas mais comuns envolvem o pareamento de documentos técnicos, relatórios de erros e componentes de *software*, utilizando modelos baseados em BERT e suas variantes, além de técnicas híbridas de ranqueamento.

Na área da Saúde, os estudos focaram principalmente no *matching* de sintomas com diagnósticos, perguntas médicas com respostas relevantes e na recuperação de casos clínicos para suporte à decisão médica (JAYASUDHA; DEEPA; DEVI, 2023; HU; ZHANG; SUN, 2023; FUJISHIRO; OTAKI; KAWACHI, 2023; CHEN et al., 2023; LI; HE, 2023; WANG, 2022; AKKASI; MOENS, 2021). Os modelos aplicados variaram desde arquiteturas baseadas em *Transformers* até redes siamesas e BiLSTM. No entanto, os estudos destacaram como principal desafio a escassez de dados rotulados, além da alta complexidade terminológica e da sensibilidade ética envolvida na manipulação de dados clínicos.

O setor de Recursos Humanos representou outro domínio com expressiva quantidade de estudos (VANETIK; KOGAN, 2023; TANBERK et al., 2023; SRIDEVI; SUGANTHI, 2022; TÜLÜMEN et al., 2021; BERNABÉ-MORENO et al., 2019). As aplicações incluem o *matching* entre currículos e descrições de vagas, classificação de perfis profissionais e recomendação de candidatos. O desafio central identificado neste domínio é a padronização das descrições de habilidades e a variação linguística nas formas como candidatos descrevem suas experiências.

No campo educacional, embora com menos representatividade, alguns métodos buscaram realizar o *matching* entre alunos e materiais de estudo, ou entre alunos e tutores, como observado nos artigos (AGRAWAL; SHUKLA, 2023; MEENAKSHI; SHANAVAS, 2022; DUAN et al., 2019). Aqui, a limitação principal está na carência de *datasets* especializados e na definição de critérios de relevância personalizados para cada usuário.

No domínio jurídico, o número de estudos foi mais restrito, mas com avanços na recuperação de casos semelhantes e na atribuição automática de revisores para periódicos acadêmicos, como demonstrado nos artigos (SU et al., 2025; DUAN et al., 2019; HOSSAIN et al., 2018). O desafio predominante nessa área envolve a complexidade semântica dos textos jurídicos e a necessidade de alto grau de interpretabilidade nos modelos, devido ao potencial

impacto das decisões.

Além das dificuldades específicas de cada domínio, desafios transversais foram identificados. O custo computacional elevado para o treinamento e inferência com modelos baseados em *Transformers* foi apontado em múltiplos estudos (WANG et al., 2024; ASKARI et al., 2024; MORAVVEJ et al., 2022; NAIR et al., 2022). Outro problema recorrente é a baixa interpretabilidade dos modelos, dificultando a adoção em domínios sensíveis como saúde e jurídico (MORAVVEJ et al., 2022; SHAN et al., 2021; DUAN et al., 2019). A escassez de *datasets* públicos e rotulados também foi uma barreira mencionada, limitando a capacidade de generalização e a realização de *benchmarks* comparáveis (FUJISHIRO; OTAKI; KAWACHI, 2023; CHEN et al., 2023; MILOŠEVIĆ; THIELEMANN, 2023; SU et al., 2025).

Como principais *gaps*, destaca-se a necessidade de mais estudos voltados para domínios com menor representatividade, como educação e jurídico, além da urgência em desenvolver métodos de explicabilidade e redução de custo computacional.

Em síntese, embora os avanços tecnológicos tenham ampliado as possibilidades de aplicação do PLN para o *matching*, os desafios técnicos, computacionais e de disponibilidade de dados ainda representam barreiras que demandam atenção nas pesquisas futuras.

4

Materiais e Métodos

Neste capítulo são descritos os materiais utilizados e a metodologia adotada para o desenvolvimento e avaliação do modelo de *matching* proposto. São apresentados os conjuntos de dados, as ferramentas computacionais, as bibliotecas empregadas, bem como as etapas de preparação dos dados, geração de *embeddings*, cálculo de similaridade e os métodos de validação utilizados.

4.1 Materiais

Esta seção apresenta as ferramentas de *software*, bibliotecas e *datasets* utilizados no desenvolvimento do modelo de *matching* proposto.

4.1.1 *Datasets* Utilizados

A construção dos conjuntos de dados utilizados no desenvolvimento e validação do modelo de *matching* baseou-se em duas fontes principais: (i) dados não estruturados, documentos acadêmicos elaborados por pesquisadores, incluindo dissertações e teses, para simular a realidade hoje da SciBees, o que seria a entrada de novos clientes de diversas áreas de conhecimento e (ii) dados estruturados fornecidos por *experts*, extraídos da base da plataforma SciBees, para testar os modelos de acordo com a realidade da SciBees em questões de abrangências de áreas de conhecimento e expertise, simulando uma situação real e as limitações de fornecer *matching* assertivo de acordo de como é realizado hoje de forma manual pela equipe técnica da SciBees.

4.1.1.1 *Dataset* dos Pesquisadores

O *dataset* dos pesquisadores foi composto a partir da extração de 513 documentos acadêmicos em formato PDF, obtidos diretamente do repositório institucional da Universidade

Estadual do Oeste do Paraná (UNIOESTE) [1]. Os documentos abrangem dissertações e teses defendidas nos campus de Cascavel, Foz do Iguaçu, Francisco Beltrão, Marechal Cândido Rondon e Toledo, referentes ao ano de 2024, abrangendo grandes áreas do conhecimento como Ciências Humanas, Ciências Sociais Aplicadas, Ciências Agrárias, Ciências da Saúde, Ciências Biológicas, Ciências Exatas e da Terra, Engenharias, Linguística, Letras e Artes, Interdisciplinar e Multidisciplinar.

Os documentos foram organizados em um diretório no *Google Drive* e convertidos para o formato texto puro, utilizando um *script* desenvolvido em *Python*, com apoio da biblioteca *pdfplumber*. O *script* percorre todos os arquivos do diretório de entrada, extrai o conteúdo textual de cada página e concatena os trechos em um único arquivo de saída, sem aplicar filtragens, marcações estruturais ou reconhecimento óptico de caracteres. A Figura 16 apresenta uma página do PDF original (entrada).

[1] <https://tede.unioeste.br/>

Figura 16 – Amostra do documento de entrada (PDF) antes da conversão com *pdfplumber*

RESUMO

A pesquisa e o desenvolvimento de novas tecnologias na área de materiais e de procedimentos químicos busca por novas inovações em variados campos da tecnologia, tais como área da medicina tem sido foco de muitos trabalhos científicos que buscam constantemente por inovações. A utilização de materiais poliméricos para aplicação na área biomédica está em constante desenvolvimento, e possuem importantes aplicações como sistema controlado de liberação de drogas, regeneração de tecidos muscular, encapsulamento de células e blendas para uso tópico. Nessa vertente destaca-se a aplicação de polímeros de origem natural como a Sericina, uma proteína extraída do casulo do bicho da seda *(Bombyx mori)*. A aplicação da Sericina em filmes poliméricos requer a utilização de agentes plastificantes e/ou reticulantes, a fim de, produzir blendas com melhores propriedades ao fim destinado. Recentemente observou-se efeito semelhante gerado pela introdução de cátions Al^{3+} em sistemas poliméricos. A pesquisa em questão, teve como foco avaliar os efeitos de íons metálicos nas propriedades de filmes poliméricos da blenda Sericina/PVA obtidos pela técnica de casting, dessa forma foi preparados filmes poliméricos com diferentes íons metálicos como: Na^{+}, Mg^{2+}, Al^{3+} entre outros íons convenientes. Os filmes foram avaliados quanto suas propriedades mecânicas, térmicas e espectroscópicas, o que possibilitou observar a solubilidade dos íons Al^{3+} induzidos a reticulação, tornando os filmes insolúveis e com características de hidrogel devido à sua baixa solubilidade. Enquanto os filmes com os íons Na^{+} e Mg^{2+} permaneceram altamente solúveis. Já em termos mecânicos, os filmes com Al^{3+} apresentaram maior rigidez devido à reticulação formada. Já os filmes sem íons metálicos, ficaram rígidos sem nenhuma maleabilidade, enquanto os filmes com Na^{+} e Mg^{2+} mostraram-se mais maleáveis. Desta forma, a introdução dos íons metálicos afeta a conformação das estruturas poliméricas dos filmes, aumentando ou diminuindo as interações intermoleculares, este fato pode estar relacionado ao raio iônico e da valência destes íons. Ainda seria possível otimizar uma condição para produção de um material com propriedades mecânicas e perfil desejado, como por exemplo, condições mais adequadas para a liberação de fármacos.

Palavras-chave: Sericina; Íons Metálicos; Blendas Poliméricas.

Fonte: Almeida et al. (2024)

A Figura 17 apresenta o trecho correspondente do arquivo já em formato `.txt` gerado após a conversão do documento.

Figura 17 – Amostra do arquivo de saída (TXT) após a conversão com *pdfplumber*

```
RESUMO
A pesquisa e o desenvolvimento de novas tecnologias na área de materiais e de
procedimentos químicos busca por novas inovações em variados campos da tecnologia,
tais como área da medicina tem sido foco de muitos trabalhos científicos que buscam
constantemente por inovações. A utilização de materiais poliméricos para aplicação na
área biomédica está em constante desenvolvimento, e possuem importantes aplicações
como sistema controlado de liberação de drogas, regeneração de tecidos muscular,
encapsulamento de células e blendas para uso tópico. Nessa vertente destaca-se a
aplicação de polímeros de origem natural como a Sericina, uma proteína extraída do
casulo do bicho da seda (Bombyx mori). A aplicação da Sericina em filmes poliméricos
requer a utilização de agentes plastificantes e/ou reticulantes, a fim de, produzir blendas
com melhores propriedades ao fim destinado. Recentemente observou-se efeito
semelhante gerado pela introdução de cátions Al3+ em sistemas poliméricos. A pesquisa
em questão, teve como foco avaliar os efeitos de íons metálicos nas propriedades de
filmes poliméricos da blenda Sericina/PVA obtidos pela técnica de casting, dessa forma
foi preparados filmes poliméricos com diferentes íons metálicos como: Na+, Mg2+, Al3+
entre outros íons convenientes. Os filmes foram avaliados quanto suas propriedades
mecânicas, térmicas e espectroscópicas, o que possibilitou observar a solubilidade dos
íons Al3+ induzidos a reticulação, tornando os filmes insolúveis e com características de
hidrogel devido à sua baixa solubilidade. Enquanto os filmes com os íons Na+ e Mg2+
permaneceram altamente solúveis. Já em termos mecânicos, os filmes com Al3+
apresentaram maior rigidez devido à reticulação formada. Já os filmes sem íons metálicos,
ficaram rígidos sem nenhuma maleabilidade, enquanto os filmes com Na+ e Mg2+
mostraram-se mais maleáveis. Desta forma, a introdução dos íons metálicos afeta a
conformação das estruturas poliméricas dos filmes, aumentando ou diminuindo as
interações intermoleculares, este fato pode estar relacionado ao raio iônico e da valência
destes íons. Ainda seria possível otimizar uma condição para produção de um material
com propriedades mecânicas e perfil desejado, como por exemplo, condições mais
adequadas para a liberação de fármacos.
Palavras-chave: Sericina; Íons Metálicos; Blendas Poliméricas.
```

Fonte: Almeida et al. (2024)

A conversão dos documentos em formato PDF para o formato TXT com *pdfplumber* apresentou boa fidelidade lexical para o corpus em PT-BR, houve apenas perdas de formatação e *layout* (negrito/itálico, figuras, tabelas e quebras de página), o que não impacta as etapas de pré-processamento. A Tabela 9 apresenta a distribuição dos documentos, categorizados segundo as respectivas grandes áreas e subáreas do conhecimento, conforme a classificação da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

Tabela 9 – Distribuição dos documentos por área do conhecimento

Grande Área	**Subárea**	**Nº de Documentos**
Ciências Sociais Aplicadas	Administração	28
	Contabilidade	9
	Economia	1
	Serviço Social	6
Ciências Agrárias	Engenharia Agrícola	28
	Engenharia de Energia na Agricultura	25
	Agronomia	14
	Zootecnia	5
	Bioenergia	1
	Recursos Pesqueiros e Engenharia de Pesca	15
Ciências da Saúde	Biociências e Saúde	21
	Ciências Farmacêuticas	6
	Odontologia	6
	Saúde Pública em Região de Fronteira	14
	Ciências Aplicadas à Saúde	16
	Educação Física	6
Ciências Exatas e da Terra	Ciência da Computação	4
	Matemática	1
	Química	6
Ciências Humanas	Educação	61
	Educação em Ciências e Educação Matemática	18
	Ensino	14
	Geografia	28
	História	20
	Filosofia	20
Ciências Biológicas	Conservação e Manejo de Recursos Naturais	6
Engenharias	Engenharia Elétrica e Computação	7
Interdisciplinar	Sociedade, Cultura e Fronteiras	22
	Tecnologias, Gestão e Sustentabilidade	7
	Desenvolvimento Rural Sustentável	31
	Desenvolvimento Regional e Agronegócio	10
Linguística, Letras e Artes	Letras	45
Multidisciplinar	Ciências Ambientais	12
	Total	**513**

4.1.1.2 Documentos selecionados para o *matching*

Para a etapa de validação do modelo de *matching*, foram selecionados 10 documentos. A escolha decorreu do custo de anotação humana: a equipe técnica da SciBees leu integralmente cada trabalho e analisou cada perfil dos *experts* e registrou os perfis de *experts* mais adequados para cada documento selecionado no experimento, sendo uma tarefa que exige tempo e disponibilidade para ser executada, com isso, foi constituído o *ground truth* de referência. A amostra foi intencionalmente heterogênea quanto às grandes áreas para simular a entrada operacional de dez novas demandas de *matching* na scibees.

Essa amostra é adequada para este estudo cujo objetivo é comparar o modelo proposto ao

processo manual sob condições reais de uso. Ampliar o número de casos sem ampliar a capacidade de anotação confiável tem ganho marginal, pois a validade das métricas depende sobretudo da qualidade do *ground truth*. A base total de pesquisadores foi extraída da UNIOESTE. A seleção foi realizada com o objetivo de compor um conjunto amostral heterogêneo, contemplando diferentes grandes áreas do conhecimento, conforme a classificação da CAPES.

Esses documentos foram utilizados em duas frentes distintas e complementares:

- ***Matching* manual**: os perfis foram analisados e pareados manualmente pela equipe técnica da plataforma SciBees, com base na leitura e interpretação humana dos textos e perfis de *experts*;
- ***Matching* automatizado**: os mesmos documentos foram utilizados como entrada nos modelos computacionais desenvolvidos neste trabalho, que utilizou como base modelos pré-treinados BERTimbau-base, BERTimbau-large, *OpenAI Embedding Model text-embedding-3-small* (OpenAI-small) e *OpenAI Embedding Model text-embedding-3-large* (OpenAI-large));

A Tabela 10 apresenta os títulos das dissertações selecionadas, os respectivos autores e as grandes áreas CAPES às quais pertencem.

Tabela 10 – Documentos selecionados para teste do modelo de *matching*

Título	**Pesquisador**	**Área CAPES**
Estratégias competitivas para pequenos mercados de Ramilândia - PR	Pesquisador 1	Ciências Sociais Aplicadas
Propriedades antioxidantes, fotoprotetoras e antimicrobianas de formulações cosméticas com apitoxina	Pesquisador 2	Ciências da Saúde
Era uma vez... Figurações do arquétipo da princesa e do pater familias em contos de Marina Colasanti	Pesquisador 3	Linguística, Letras e Artes
Estratégias de aprendizagem para o ensino médio: construção de um recurso instrucional	Pesquisador 4	Ciências Humanas
Modelagem de corredores ecológicos no estado do Paraná e análise comparativa com áreas estratégicas de conservação	Pesquisador 5	Ciências Biológicas
Substituição parcial da farinha de peixe por hidrolisados proteicos na dieta de Penaeus vannamei	Pesquisador 6	Ciências Agrárias
Abordagens para o problema do desbalanceamento em detecção de intrusão - um estudo de caso com CIC-IDS2018	Pesquisador 7	Engenharias
Esquiva e fuga: diálogos entre Heidegger e o Buda acerca do modo de ser na cotidianidade mediana	Pesquisador 8	Ciências Humanas
Uso de aminoácidos, nanopartículas de prata e micropartículas de zinco no tratamento de sementes em soja	Pesquisador 9	Ciências Agrárias
Efeito de cátions metálicos nas propriedades de filmes poliméricos à base de sericina e álcool polivinílico	Pesquisador 10	Ciências Exatas e da Terra

4.1.1.3 *Dataset* dos *Experts*

O *dataset* dos *experts* utilizado neste trabalho foi fornecido pela equipe da plataforma SciBees, composta por profissionais com sólida formação acadêmica e experiência em pesquisa científica. Foram selecionados todos os 10 perfis de *experts* [2] que compõem o quadro de *expert* da scibees, previamente avaliados pela equipe da SciBees, com o objetivo de compor a base de dados para os testes de *matching* automatizado e a criação do *ground truth* manual de referência.

Cada perfil contém informações extraídas diretamente da base de cadastro da Scibees

[2] <https://drive.google.com/drive/folders/1Yd9UEqWpEgwYeI3Kgt_qlfBTijLjmfTe?usp=drive_link/>

[3] dos *experts*, disponibilizado o arquivo em formato .xlsx (Excel) que abrange informações organizadas de forma estruturada, separadas por colunas:

- Pseudo Nome;
- Grande área de conhecimento (ex.: Ciências Biológicas, Ciências Exatas, Linguística, Ciências Sociais Aplicadas);
- Temas de dissertação e tese;
- Experiência em diferentes tipos de pesquisa (qualitativa, quantitativa, quali-quanti, revisão sistemática, metanálise, entre outras);
- Habilidades metodológicas e técnicas (análise de conteúdo, análise estatística, análise documental, modelagem, análise do discurso etc.);
- Proficiência em idiomas;
- Resumo *Experts* apresenta uma síntese técnica elaborada pela equipe da SciBees, baseada no conhecimento tácito acumulado sobre a trajetoria de cada *expert* junto a Scibees. Contendo, informações estratégicas e qualitativas, destacando pontos fortes, áreas de maior expertise e aspectos relevantes para o processo de *matching*;

A Tabela 11 apresenta um resumo do *dataset* dos *experts*, destacando suas grandes áreas, tipos de pesquisa e habilidades técnicas.

[3] <https://scibees.com.br>

Tabela 11 – Resumo dos perfis dos *experts* selecionados

Nome	**Grande Área**	**Experiência**	**Habilidades Técnicas**	**Idioma**	**Resumo do Expert**
Expert 1	Ciências Exatas e da Terra	Quantitativa, IA, Metanálise	Análise Quantitativa, Metanálise	Inglês	Coordenador de IA da Fundação CERTI. Experiência em inteligência artificial, metanálise e revisão sistemática. Atua com orientação e escrita científica em português e inglês.
Expert 2	Ciências Biológicas	Quali e Quantitativa	Bardin, Análise Quantitativa	Inglês	Bacharela e Licenciada em Biologia pela USP. Atua com ecologia de comunidades, atributos funcionais e educação científica. Experiência em análise de conteúdo de Bardin e métodos mistos.
Expert 3	Ciências Sociais Aplicadas	Qualitativa, Revisão Sistemática	Análise Qualitativa, Bibliométrica	Inglês	Doutora em Ciência da Informação pela UFMG. Pesquisa gestão da informação, imaginários sociais e tomada de decisão. Experiência com metodologias qualitativas e bibliométricas.
Expert 4	Ciências Biológicas	Quantitativa, Revisão Sistemática	Modelagem Ecológica	Inglês	Pós-doutorando no Instituto de Biociências da USP. Pesquisa efeitos ecológicos do fogo sobre comunidades de plantas e polinizadores. Experiência em modelagem e ecologia aplicada.
Expert 5	Ciências da Saúde	Quantitativa, CEP, CEUA	Análise Estatística, Revisão	Inglês	Mestre em Bioquímica pela UFRGS, doutoranda em Biologia Celular e Molecular. Pesquisa bioinformática, metanálise e integração multi-ômica.
Expert 6	Ciências Biológicas	Quantitativa, Bioinformática	Quali-Quantitativa, Bibliométrica	Inglês	Pesquisadora do Instituto SENAI de Inovação. Atua em bioinformática, aprendizado de máquina e revisões sistemáticas.
Expert 7	Ciências Sociais Aplicadas	Qualitativa	Discurso, Conteúdo, Documental	Inglês	Doutora em Sociologia pela UFS. Experiência em pesquisa qualitativa, análise de discurso e exclusão social. Pós-doutora pela Universidade Tiradentes.
Expert 8	Ciências Biológicas	Quali-Quantitativa	CEP, CEUA, Conteúdo	Inglês	Doutora em Biologia pela UFRGS. Atua com genética de populações, expressão gênica e docência no ensino superior.
Expert 9	Ciências Biológicas	Quantitativa, Modelagem	R, Estatística, Geoprocessamento	Inglês	Doutor em Ecologia. Experiência em modelagem ecológica, geoprocessamento e estatística aplicada à biodiversidade.
Expert 10	Letras e Artes	Qualitativa	Discurso, Ecocrítica	Inglês	Doutora em Letras. Pesquisa discurso, literatura e meio ambiente. Atua com ecocrítica e análise cultural.

Os perfis dos *experts* apresentados refletem o cenário real de atuação da SciBees, em que todos os dez *experts* atendem a diferentes áreas do conhecimento e empregam variadas abordagens metodológicas. Essa diversidade permite avaliar a capacidade dos modelos propostos em identificar compatibilidades interdisciplinares. É importante destacar que os dados dos *experts*

não são desbalanceados, pois representam de forma fidedigna o *dataset* operacional da empresa.

Na SciBees, o processo de *matching* manual não se limita à grande área de formação acadêmica dos *experts*. Conforme ilustrado na Figura 18, o processo envolve múltiplos critérios que consideram, além da área de conhecimento, a experiência prévia e as habilidades técnicas do profissional. Dessa forma, ainda que parte dos *experts* possua formação principal em áreas como Biologia, esses profissionais atuam de maneira transversal em projetos de outras grandes áreas, desde que apresentem domínio metodológico e técnico compatível com as demandas dos pesquisadores.

O *dataset* dos *experts* foi utilizado como referência para os testes de validação com os modelos BERTimbau-base, BERTimbau-large, OpenAI-small e OpenAI-large, permitindo a comparação entre os resultados obtidos automaticamente e os pareamentos realizados manualmente pela equipe avaliadora.

4.1.2 Ambiente Computacional e Ferramentas Utilizadas

Todo o processo de desenvolvimento, teste e execução dos experimentos deste trabalho foi realizado na plataforma *Google Colab*, utilizando a versão gratuita (sem suporte a *Graphics Processing Unit* (GPU)). Essa decisão foi adotada para garantir a reprodutibilidade dos resultados, assegurando que o desempenho observado dos modelos não fosse influenciado por aceleração de *hardware*.

O uso do *Google Drive* como sistema de armazenamento auxiliar permitiu a organização eficiente de todos os diretórios de entrada e saída, incluindo os *datasets*, os arquivos intermediários de vetorização e os resultados finais das métricas.

4.1.3 Bibliotecas Utilizadas

O desenvolvimento do modelo de *matching* fez uso de diversas bibliotecas e recursos computacionais que viabilizaram tanto o pré-processamento textual quanto a geração de representações semânticas e a comparação entre vetores. A seguir, são descritas as principais bibliotecas e ferramentas utilizadas, acompanhadas de suas respectivas fontes de acesso:

- **pdfplumber**[4]: utilizada para extração de texto bruto a partir de arquivos PDF, permitindo controle preciso de páginas e segmentação estruturada dos conteúdos das dissertações e teses.
- **spaCy**[5]: biblioteca de PLN empregada nas etapas de pré-processamento linguístico, realizando operações como tokenização, remoção de *stopwords* e normalização de textos.

[4] <https://github.com/jsvine/pdfplumber>
[5] <https://spacy.io>

- **transformers (Hugging Face)**[6]: utilizada para o carregamento e aplicação de modelos de linguagem baseados em *Transformers*. Neste projeto, foram empregados os modelos **BERTimbau Base** e **BERTimbau Large**, ambos treinados especificamente para o português:
  - `neuralmind/bert-base-portuguese-cased`[7]
  - `neuralmind/bert-large-portuguese-cased`[8]
- **sentence-transformers**[9]: usada para facilitar a aplicação de modelos do tipo BERT em tarefas de *embedding* de sentenças e documentos, com suporte para processamento em lote e uso de GPU. A seleção dos modelos BERTimbau-base e BERTimbau-large decorre de evidência empírica da literatura e de resultados superiores sobre modelo multilíngue BERT em tarefas semânticas (SOUZA; NOGUEIRA; LOTUFO, 2019). Essa vantagem é relevante para o problema de *matching* desta pesquisa, que depende de representação fina de relações semânticas na língua portuguesa.
- **torch (PyTorch)**[10]: biblioteca de aprendizado profundo utilizada para a manipulação de tensores e integração com os modelos BERT e seus *embeddings* vetoriais.
- **OpenAI API**[11]: utilizada para gerar representações vetoriais dos textos a partir dos modelos `text-embedding-3-small` e `text-embedding-3-large`, disponíveis por meio da API da OpenAI. Optou-se por avaliar esses modelos por serem soluções comerciais estáveis e escalável. Essa comparação quantifica a qualidade dos modelos em relação a qualidade dos *matching* além do custo e latência que são relevantes ao cenário aplicado deste trabalho.
- **scikit-learn**[12]: empregada para o cálculo da similaridade do cosseno, métrica essencial para medir a compatibilidade entre os vetores representando pesquisadores e *expert*.
- **pandas**[13] e **NumPy**[14]: utilizadas para leitura, manipulação e organização dos dados estruturados, como arquivos `.csv` e `.xlsx`, além de operações vetoriais e estatísticas.
- **matplotlib**[15]: empregada para geração de gráficos, incluindo visualizações como distribuição de artigos por ano.
- **openpyxl**[16]: utilizada para leitura e escrita de arquivos Excel no formato `.xlsx`, em conjunto com o pandas.

6 <https://huggingface.co/transformers>
7 <https://huggingface.co/neuralmind/bert-base-portuguese-cased>
8 <https://huggingface.co/neuralmind/bert-large-portuguese-cased>
9 <https://www.sbert.net>
10 <https://pytorch.org>
11 <https://platform.openai.com/docs/guides/embeddings>
12 <https://scikit-learn.org>
13 <https://pandas.pydata.org>
14 <https://numpy.org>
15 <https://matplotlib.org>
16 <https://openpyxl.readthedocs.io>

4.2 Metodologia

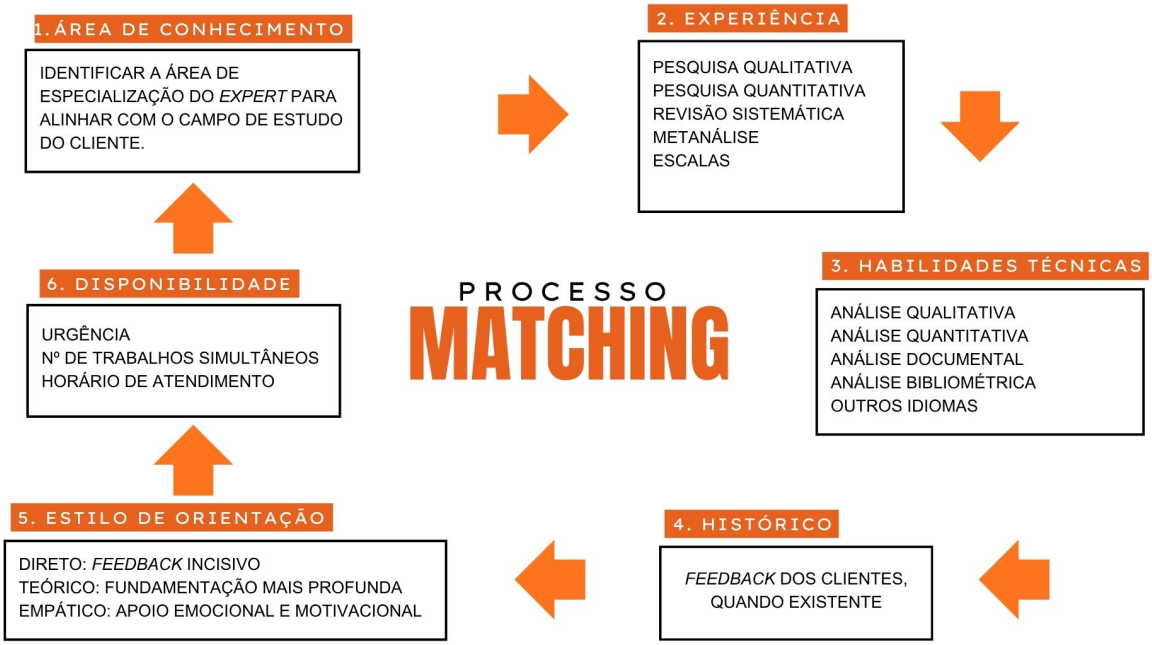
4.2.1 Processo de *Matching*

Esta seção apresenta as metodologias aplicadas no processo manual de *matching* da SciBees e a metodologia do modelo desenvolvido, detalhando cada etapa da metodologia desenvolvida.

4.2.1.1 Processo de *Matching* Manual na SciBees

O processo de *matching* entre pesquisadores e *experts* atualmente praticado na plataforma SciBees é realizado de maneira manual e envolve múltiplas etapas operacionais. Esse fluxo é caracterizado por decisões baseadas em critérios qualitativos e conhecimento tácito da equipe de gestão. A Figura 18 apresenta de forma esquemática a metodologia atualmente adotada para condução do processo.

Figura 18 – Metodologia do processo de *matching* manual da SciBees



Fonte: Autoria própria (2025)

O fluxo inicia-se com a captação da demanda pelos setores de *Marketing* e Comercial, os quais realizam o levantamento inicial das necessidades do cliente. Após o fechamento do contrato, as informações são encaminhadas para a equipe de Gestão Operacional, que realiza uma análise detalhada da solicitação. Nessa análise, são considerados aspectos como a área de conhecimento solicitada, a complexidade do projeto, a experiência requerida e as habilidades técnicas demandadas.

Com base nesses critérios, a equipe de gestão realiza a escolha manual do *expert* que melhor atende às características da demanda. O *expert* selecionado realiza o primeiro contato com o cliente e conduz as etapas subsequentes de mentoria e revisão, até a entrega final do serviço contratado.

A metodologia manual atual, apesar de estruturada, apresenta limitações em termos de escalabilidade e tempo de resposta. Nesse contexto, o modelo automatizado proposto nesta dissertação busca reproduzir e automatizar as etapas decisórias atualmente realizadas de forma humana, utilizando técnicas de PLN e aprendizado de máquina para aumentar a eficiência e reduzir o esforço operacional necessário.

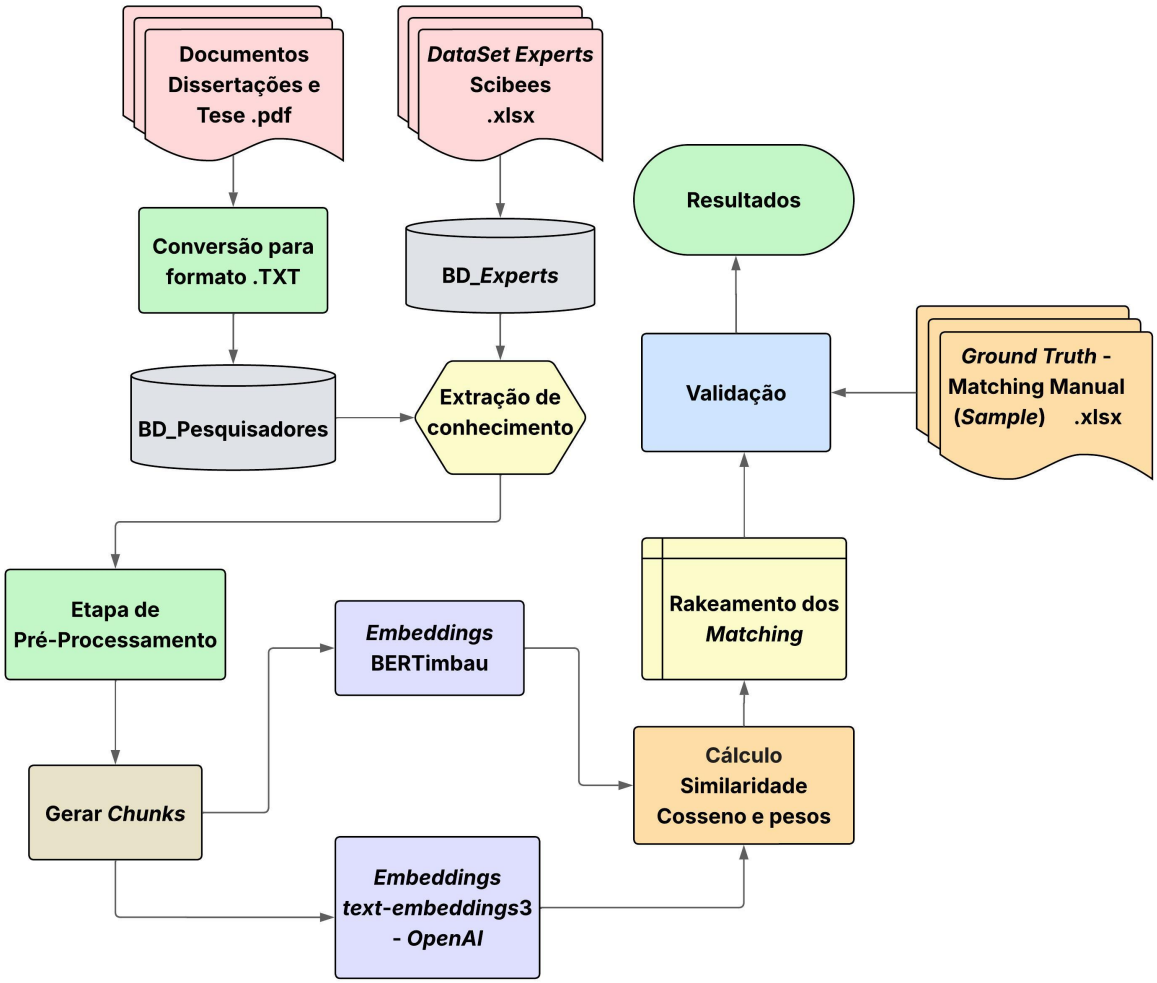
Com o objetivo de estabelecer uma base de referência *ground truth* para utilizar na validação do modelo proposto e conferir maior rigor metodológico ao experimento, foi conduzido um procedimento sistemático de *matching* manual. Este processo foi executado pela gestora operacional da SciBees, profissional responsável pela atual metodologia de alocação e detentora de profundo conhecimento sobre o quadro de *experts* da empresa. O corpus de análise foi composto por um conjunto de 10 documentos, representando demandas reais de clientes. A gestora realizou a leitura integral de cada documento para extrair as características centrais da pesquisa, como seus objetivos e o tipo de estudo, focando a análise em seções de maior densidade informacional, como o Resumo, a Introdução e a Metodologia. Com base nesse levantamento, a equipe técnica para esse momento aplicou os critérios definidos nas três primeiras etapas do processo de *matching* manual Área de Conhecimento, Experiência e Habilidades Técnicas. Como resultado, para cada um dos 10 documentos, foi gerada uma lista ordenada dos três *experts* considerados mais qualificados para atender à demanda. Este conjunto de dados associando cada documento a seus três melhores *experts* serviu como o gabarito de referência para a avaliação do modelo de automação desenvolvido neste trabalho.

Diante dessas limitações, o presente trabalho propõe a automação parcial do processo de *matching*, com foco nas primeiras etapas decisórias descritas na metodologia manual, especificamente as etapas de análise da demanda, identificação da área de conhecimento, experiência e habilidades técnicas dos *experts* (Etapas 1, 2 e 3 da Figura 18). Essas etapas foram mapeadas e incorporadas ao pipeline computacional desenvolvido, cujo fluxo detalhado será apresentado na subseção a seguir.

4.2.1.2 Automação do processo de *matching*

A Figura 19 apresenta o fluxograma da arquitetura metodológica desenvolvida para o modelo de *matching* entre pesquisadores e *experts*. O diagrama representa, de forma sequencial e modular, todas as etapas envolvidas no processo, desde a coleta dos dados brutos até a geração e validação dos resultados.

Figura 19 – Fluxograma da arquitetura do modelo de *matching*



Fonte: Autoria própria (2025)

O processo inicia com a obtenção de dados a partir de duas fontes principais:

- Dissertações e teses em PDF pesquisadores, extraídas do repositório institucional da UNIOESTE contendo dados não estruturados;
- Planilhas estruturadas em Excel contendo os dados dos *experts* fornecidos pela plataforma SciBees.

Os documentos dos pesquisadores são convertidos para o formato texto por meio de um *script* em *Python*, utilizando a biblioteca `pdfplumber`. Essa conversão é essencial para permitir o pré-processamento e a vetorização dos textos. Em paralelo, os dados dos *experts* são estruturados no banco de dados `BD_Experts` a partir dos arquivos tabulares.

Com os dois bancos de dados previamente estruturados — *BD_Pesquisadores* e *BD_Experts* — inicia-se a etapa de extração de conhecimento, fundamental para alimentar

o processo de *matching*. Cada banco de dados é tratado de forma distinta, considerando a natureza e o formato das informações disponíveis.

No caso dos pesquisadores, os dados brutos consistem em arquivos textuais contendo as dissertações e teses previamente convertidas para o formato texto. A extração de atributos relevantes é realizada por meio de um *script* desenvolvido em *Python*, que utiliza técnicas baseadas em expressões regulares (*regex*) para localizar e segmentar seções-chave de cada documento.

O *script* percorre todos os arquivos da pasta de entrada, buscando padrões específicos que identificam as seções de Resumo, Introdução e Metodologia. Cada uma dessas seções é isolada e salva em arquivos de saída, organizados individualmente por pesquisador. A estruturação dos dados extraídos permite posterior análise semântica, categorização temática e identificação de informações metodológicas relevantes.

No caso do banco de dados dos *experts* (*BD_Experts*), o processo de extração de conhecimento segue uma abordagem diferenciada, baseada na leitura de um arquivo estruturado no formato nativo *MS-Excel* (`xlsx`). Inicialmente, todas as colunas da planilha são carregadas para a memória utilizando a biblioteca `pandas` do *Python*. Contudo, apenas um subconjunto específico de atributos é selecionado para compor o perfil textual de cada *expert*.

As colunas utilizadas incluem: Área de atuação, Experiência, Habilidades técnicas e o campo Resumo *Experts*. Esses atributos foram escolhidos por representarem de forma integrada os principais aspectos que caracterizam o perfil acadêmico e profissional de cada *expert*.

Para construir o perfil textual final de cada *expert*, os conteúdos dessas colunas são concatenados com um sistema de pesos ponderados, de modo a dar maior ou menor importância relativa a cada dimensão de informação durante o processo de geração de *embeddings*. Os pesos atribuídos às características de entrada foram definidos de forma alinhada ao processo interno de *matching* manual realizado pela equipe técnica da SciBees. A ponderação estabelecida foi de 20% para a área de atuação, 30% para a experiência, 30% para as habilidades técnicas e 20% para o resumo técnico. Essa distribuição reflete o entendimento operacional de que a experiência prática e as competências metodológicas do *expert* exercem maior influência na escolha da melhor correspondência, uma vez que determinam sua capacidade de auxiliar o pesquisador de forma efetiva.

Por outro lado, embora a área de atuação e o resumo técnico sejam fatores importantes para assegurar o alinhamento inicial com o tema de pesquisa, sua relevância isolada é limitada. Em outras palavras, somente compartilhar uma mesma grande área do conhecimento não garante, por si só, a compatibilidade entre pesquisador e *expert*. Dessa forma, os pesos definidos buscam reproduzir com fidelidade os critérios adotados no processo de *matching* manual da SciBees, preservando a priorização de atributos que contribuem diretamente para a qualidade da recomendação.

Após a concatenação e ponderação textual, cada perfil é submetido ao processo de *embedding* utilizando os modelos da OpenAI e BERTimbau. Na Figura 20 apresenta uma amostra da saída gerada dos *embeddigns* para os 10 *experts* e 10 pesquisadores, para o modelo BERTimbau-base, que possui 768 dimensões. O *preview*5" mostra os cinco primeiros valores do vetor. Esses valores são componentes do vetor denso no espaço semântico. Em casos de perfis muito extensos, o texto é automaticamente segmentado em pedaços menores (*chunks*) de até 2500 *tokens* e 512 *tokens* para os modelos BERTimbau. Os *embeddings* gerados para cada pedaço são, então, agregados por média vetorial, formando uma representação densa única para cada *expert*.

Figura 20 – Amostra *Embeddings*

```
=== EMBEDDINGS | RESUMO ===
Modelo: neuralmind/bert-base-portuguese-cased | Dimensão: 768
Experts: 10 | Pesquisadores: 10
[Expert 1] | preview5=[-0.11190000176429749, -0.20880000293254852, 0.7134000062942505, 0.016100000590085983, 0.35589998960494995]
[Expert 2] | preview5=[-0.09139999747276306, -0.16850000619888306, 0.9973000288009644, 0.061400000005960464, 0.15230000019073486]
[Expert 3] | preview5=[-0.15000000596046448, -0.10840000212192535, 0.8562999963760376, 0.09950000047683716, 0.25440001487731934]
[Expert 4] | preview5=[-0.1624000072479248, -0.1388999968767166, 0.9045000076293945, -0.05220000073313713, 0.19599999487400055]
[Expert 5] | preview5=[-0.09120000153779984, -0.1535000056028366, 0.9010999798774719, -0.015599999576807022, 0.3508000075817108]
[Expert 6] | preview5=[-0.19200000166893005, -0.1876000001927719, 0.8438000082969666, 0.09749999642372131, 0.358599990606308]
[Expert 7] | preview5=[-0.13590000569820404, -0.16099999845027924, 0.7472000122070312, -0.016200000420212746, 0.2231999933719635]
[Expert 8] | preview5=[-0.2353000044822693, -0.14499999582767487, 0.7910000085830688, 0.06599999964237213, 0.22869999706745148]
[Expert 9] | preview5=[-0.1331000030040741, -0.026599999517202377, 0.9218000173568726, -0.027300000190734863, 0.14959999918937683]
[Expert 10] | preview5=[-0.2535000145435333, -0.1200999990105629, 0.7300999760627747, -0.013199999928474426, 0.3540000021457672]
[Pesquisador 1] | preview5=[-0.029400000348687172, -0.08649999648332596, 0.7263000011444092, 0.03020000085234642, 0.24809999763965607]
[Pesquisador 2] | preview5=[-0.08980000019073486, -0.024399999529123306, 0.6050000190734863, 0.013000000268220901, 0.17419999837875366]
[Pesquisador 3] | preview5=[-0.08860000222921371, -0.13099999725818634, 0.675000011920929, -0.006300000008195639, 0.26080000400543213]
[Pesquisador 4] | preview5=[-0.12970000505447388, -0.1234000027179718, 0.7488999962806702, 0.029600000008940697, 0.3431999981403351]
[Pesquisador 5] | preview5=[-0.17800000309944153, -0.04320000112056732, 0.8514000177383423, 0.006099999882280827, 0.18629999458789825]
[Pesquisador 6] | preview5=[-0.1983000040054321, -0.01769999973475933, 0.724399983882904, -0.009200000204145908, 0.18199999630451202]
[Pesquisador 7] | preview5=[-0.225600004196167, -0.07900000363588333, 0.828000009059906, 0.04060000018119812, 0.19009999930858612]
[Pesquisador 8] | preview5=[-0.1923999935388565, -0.06080000102519989, 0.8648999929428101, 0.10279999673366547, 0.15800000727176666]
[Pesquisador 9] | preview5=[-0.11289999634027481, -0.14679999664952469, 0.7947999835014343, 0.031099999323487282, 0.2897000014781952]
[Pesquisador 10] | preview5=[-0.27149999141693115, -0.13770000636577606, 0.7443000078201294, 0.05260000005364418, 0.17730000615119934]
```

Fonte: Autoria própria (2025)

Essa estruturação vetorial dos perfis permite, na etapa seguinte, o cálculo das similaridades entre pesquisadores e *experts* por meio da métrica de similaridade do cosseno e a geração de *rankings* para o *matching*.

Os textos segmentados são então submetidos à geração de representações vetoriais (*embeddings*) por meio de dois caminhos distintos:

- *Embeddings* com BERTimbau: gerados a partir de modelos pré-treinados em português, disponibilizados pela biblioteca SentenceTransformers;
- *Embeddings* gerados por meio dos modelos OpenAI-small e OpenAI-large, acessados via *Application Programming Interface* (API) da OpenAI.

Com os vetores gerados, realiza-se o cálculo da similaridade, utilizando a métrica do cosseno, ponderada por pesos atribuídos a diferentes atributos (ex: experiências, habilidades, área de conhecimento, resumo). A soma ponderada das similaridades parciais gera uma pontuação combinada para cada par pesquisador e *experts*. Em seguida, ocorre o ranqueamento dos *experts*, ordenando-os para cada pesquisador com base na pontuação final. O resultado do processo é exportado em planilhas e submetido à etapa de validação.

A validação é realizada comparando-se os resultados obtidos com um conjunto *ground truth* previamente definido pela equipe da sciBees. Esse conjunto de referência é composto por 10 documentos com *matching* manual, e as métricas de avaliação computadas incluem: Precision@3, MRR@3, nDCG@3, HR@3 e similaridade de cosseno. Por fim, os resultados são consolidados e analisados no Capítulo 5, fornecendo uma base empírica para comparar o desempenho dos modelos testados. Embora a validação tenha sido documentada com uma única iteração utilizando 10 documentos, diferentes execuções e abordagens foram realizadas durante o desenvolvimento do método, assegurando consistência nos resultados observados. Além disso, os documentos selecionados representam situações reais enfrentadas pela SciBees, cobrindo diferentes áreas do conhecimento e níveis de complexidade. Dessa forma, considera-se que os resultados obtidos não sofrem influências significativas de viés amostral.

5

Resultados e Discussão

Este capítulo apresenta os resultados experimentais obtidos com a implementação e avaliação de diferentes modelos de *matching* baseados em PLN e aprendizado profundo.

5.1 Resultados e discussão da pesquisa experimental

O objetivo central desta etapa da pesquisa foi desenvolver e testar um algoritmo capaz de identificar as correspondências relevantes entre pesquisadores e *experts*, considerando o uso de dados não estruturados e estruturados. Para isso, foram utilizados modelos pré-treinados de linguagem que capturam relações semânticas latentes, permitindo uma associação mais contextualizada entre os perfis analisados.

A avaliação do desempenho foi conduzida com dados reais, abrangendo diferentes cenários de complexidade. As métricas adotadas para análise incluem *Precision*, além de métricas de ranqueamento como MRR@k, nDCG@k e HR@K, com o objetivo de medir tanto a capacidade de cobertura quanto a qualidade da ordenação das recomendações.

Os resultados obtidos são apresentados nas seções a seguir, considerando uma comparação detalhada entre quatro modelos implementados, com análise estatística, visualizações gráficas e discussão crítica sobre o desempenho observado.

5.1.1 Modelos avaliados

Nesta seção, são descritos os modelos de linguagem utilizados na tarefa de *matching* entre pesquisadores e *experts*, com foco em arquiteturas modernas de PLN aplicadas ao contexto da língua portuguesa ou multilíngue, incluindo alternativas de código aberto e comerciais de alto desempenho.

- BERTimbau-base: modelo BERTimbau pré-treinado especificamente para o idioma portu-

guês, baseado na arquitetura BERTimbau-base, com 12 camadas e aproximadamente 110 milhões de parâmetros.

- BERTimbau-large: versão expandida do BERTimbau, com 24 camadas e aproximadamente 335 milhões de parâmetros, oferecendo maior capacidade de modelagem e abstração semântica.
- OpenAI-small: modelo proprietário fornecido via API pela OpenAI, otimizado para rapidez e baixo custo computacional, utilizado como alternativa leve para tarefas de similaridade textual.
- OpenAI-large: modelo de última geração da OpenAI com alta capacidade de generalização semântica, empregado para avaliar o potencial de performance na tarefa de *matching*.

Cada modelo foi utilizado para gerar representações vetoriais (*embeddings*) dos textos descritivos de pesquisadores e *experts*. Em seguida, aplicou-se a métrica de similaridade do cosseno para identificar os pares mais compatíveis. A avaliação quantitativa foi conduzida com base em métricas clássicas de ranqueamento e recuperação de informação, conforme descrito nas próximas seções.

A Tabela 12 apresenta um resumo técnico das principais características de cada modelo.

Tabela 12 – Resumo técnico dos modelos avaliados

Modelo	**Idioma**	**Tipo de Arquitetura**	**Categoria**
BERTimbau-Base	Português	BERT-base (12 camadas)	Aberto
BERTimbau-Large	Português	BERT-large (24 camadas)	Aberto
OpenAI-Small	Multilíngue	Proprietário (API)	Comercial
OpenAI-Large	Multilíngue	Proprietário (API)	Comercial

5.1.1.1 Resultados quantitativos

Nesta seção, são apresentados os resultados obtidos a partir da aplicação dos modelos na tarefa de *matching* entre pesquisadores e *experts*. A avaliação foi realizada com base em métricas de ranqueamento e desempenho classificatório, considerando os três principais *experts* retornados para cada pesquisador (Top-3).

A métrica *Precision@3* quantifica a proporção de recomendações corretas entre os três primeiros *experts* retornados. O MRR@3 considera a posição da primeira ocorrência relevante no *ranking* gerado. Já o nDCG@3 pondera os resultados com base em uma escala binária de relevância, valorizando posições mais altas. Por fim, a métrica HR@3 verifica se pelo menos um *expert* relevante está presente no Top-3. Todos os valores foram calculados individualmente para cada pesquisador e, em seguida, agregados por média para análise comparativa entre os modelos.

5.1.1.2 Análise individual: Bertimbau-base

O modelo BERTimbau-base foi o primeiro a ser avaliado na tarefa de *matching* entre pesquisadores e *experts*. Por ser uma arquitetura pré-treinada voltada ao idioma português, esperava-se que o modelo apresentasse um bom desempenho na identificação de relações semânticas presentes nos textos analisados. A Tabela 13 apresenta os resultados obtidos pelo modelo BERTimbau-base.

Tabela 13 – Resultado Modelo BERTimbau-base

Pesquisadores	**Precision@3**	**HR@3**	**MRR@3**	**nDCG@3**
Pesquisador 1	0,3333	1	0,5	0,2961
Pesquisador 2	0,3333	1	0,3333	0,2346
Pesquisador 3	0,0000	0	0,0	0,0000
Pesquisador 4	0,3333	1	0,5	0,2961
Pesquisador 5	0,6667	1	1,0	0,7654
Pesquisador 6	0,6667	1	0,5	0,5307
Pesquisador 7	0,3333	1	0,5	0,2961
Pesquisador 8	0,3333	1	0,3333	0,2346
Pesquisador 9	0,6667	1	0,5	0,5307
Pesquisador 10	0,6667	1	0,5	0,5307

Os resultados obtidos revelaram um desempenho moderado, com variações significativas entre os casos. A métrica Precision@3 apresenta valores entre 0,0 e 0,6667. Em quatro dos dez testes realizados, essas três métricas atingiram 0,6667, o que indica que o modelo foi capaz de retornar dois *experts* relevantes entre os três primeiros resultados. Em outros cinco casos, os valores registrados foram de 0,3333 para a métrica Precision@3, sinalizando que apenas um *expert* correto foi incluído entre os três retornos.

No entanto, em um dos dez cenários avaliados, o modelo não conseguiu recuperar nenhum *expert* relevante, resultando em Precision@3 igual a 0,0. Esse caso demonstra a limitação do modelo em manter consistência nos resultados, especialmente quando há maior variação semântica entre os documentos analisados.

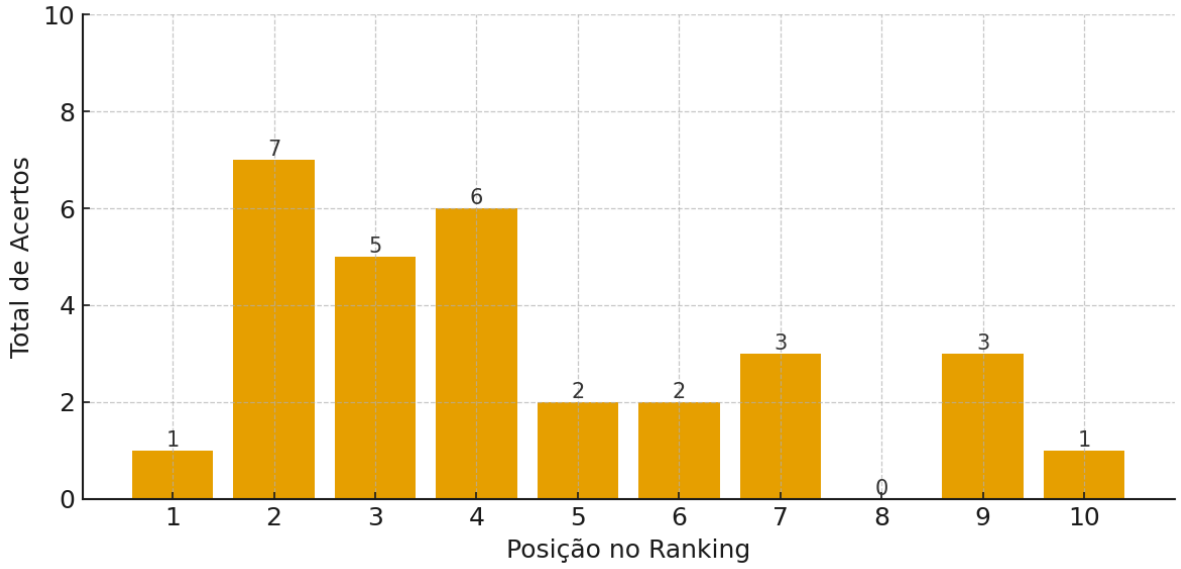
A HR@3, que mede a proporção de vezes em que ao menos um *expert* correto foi incluído no *ranking* top-3 de recomendações, atingiu 90%. Isso significa que, em 9 dos 10 casos analisados, houve pelo menos um acerto entre os três retornos.

Em relação às métricas de ordenação, o MRR@3 variou entre 0,0 e 1,0 ao longo dos experimentos. O valor máximo, 1,0, indica que em pelo menos um dos testes o *expert* mais relevante foi corretamente posicionado na primeira posição do *ranking* gerado pelo modelo. Por outro lado, o valor mínimo de 0,0 ocorreu em uma vez dos dez casos avaliados, refletindo situações em que nenhum dos *experts* relevantes apareceu entre os três primeiros retornos. Nos demais casos, o MRR@3 apresentou valor igual a 0,5 em 6 dos 10 casos analisados, indicando

que o *expert* relevante apareceu na segunda posição do *ranking* gerado. Em 2 casos, o valor foi de 0,3333, o que significa que o *expert* relevante foi identificado apenas na terceira posição.

A métrica nDCG@3, que avalia a presença, assim como a posição relativa dos *experts* relevantes no *ranking*, apresentou valores que oscilaram entre 0,0 e 0,7654. Em 3 casos, o valor foi de 0,5307, indicando que o *expert* relevante apareceu na primeira posição do *ranking*. Em 3 casos, o valor foi de 0,2961, sinalizando que o *expert* relevante apareceu na segunda posição. Em 2 casos, a métrica foi de 0,2346, o que indica que o *expert* relevante estava na terceira posição. Em 1 caso, o valor foi 0,7654, demonstrando que dois *experts* relevantes estavam nas posições 1 e 2. Em 1 caso, o valor foi 0,0, confirmando que nenhum *expert* relevante foi encontrado no Top-3. A Figura 21 apresenta o total de acertos nas dez posições do *ranking*.

Figura 21 – Acerto por posição Modelo Bertimbau-base



Fonte: Autoria própria (2025)

O modelo BERTimbau-base acertou uma recomendação correta apenas na primeira posição, é obteve maior quantidade de acertos na segunda posição, seguido de cinco acertos na terceira posição, obtendo aproximadamente 43,33% de acerto nas três primeiras posições. O modelo ainda acertou seis recomendações corretas na posição quatro, próximo aos top-3. Mas na posição cinco e seis acertou apenas duas recomendações, para a posição sete obteve três acertos, e na posição oito nenhum acerto, na posição nove teve três acertos e na última posição do *ranking* teve um acerto. Aproximadamente 70% dos acertos concentram-se nas cinco primeiras posições do *ranking*.

O BERTimbau-base demonstrou ser funcional e adaptado ao idioma, mas com limitações perceptíveis em termos de consistência e ordenação. Seu uso pode ser justificado em cenários com restrições computacionais ou quando se deseja uma solução de código aberto adaptada ao contexto linguístico brasileiro.

5.1.1.3 Análise Individual: Bertimbau-large

O modelo BERTimbau-large representa a versão expandida do BERTimbau, com maior número de camadas e parâmetros, proporcionando maior capacidade de abstração semântica. Por ser treinado especificamente para o português, era esperado que esse modelo tivesse desempenho superior ao BERTimbau-base na tarefa de *matching* entre pesquisadores e *experts*. A Tabela 14 apresenta os resultados obtidos pelo modelo BERTimbau-large.

Tabela 14 – Resultado Modelo BERTimbau-large

Pesquisadores	**Precision@3**	**HR@3**	**MRR@3**	**nDCG@3**
Pesquisador 1	0,0000	0	0,0	0,0000
Pesquisador 2	0,3333	1	1,0	0,4693
Pesquisador 3	0,0000	0	0,0	0,0000
Pesquisador 4	0,0000	0	0,0	0,0000
Pesquisador 5	1,0000	1	1,0	1,0000
Pesquisador 6	0,6667	1	1,0	0,7654
Pesquisador 7	0,6667	1	1,0	0,7039
Pesquisador 8	0,3333	1	0,3333	0,2346
Pesquisador 9	0,3333	1	0,5	0,2961
Pesquisador 10	0,3333	1	0,3333	0,2346

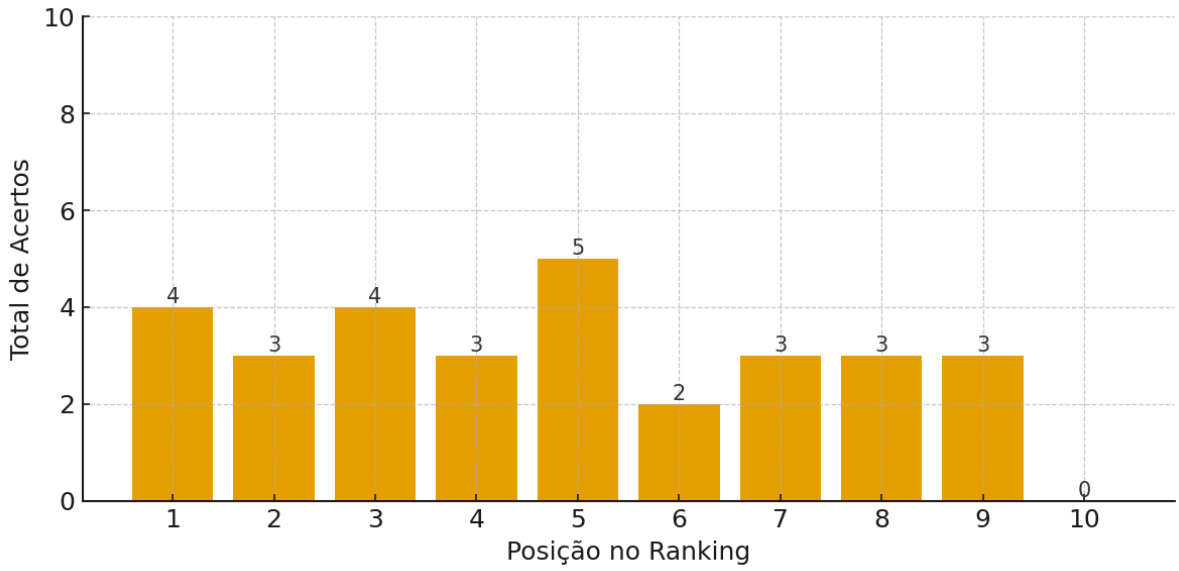
A análise dos resultados revela um desempenho intermediário, com variação expressiva entre os testes. Em apenas um dos dez casos avaliados, o modelo obteve Precision@3, com valor igual a 1,0, indicando que todos os *experts* recomendados estavam corretos. Em outros dois casos, obteve o valor de 0,6667, representando acerto de dois *experts* entre os três primeiros retornos.

Além disso, foram observados quatro casos com valores de 0,3333 para a métrica Precision@3, indicando que apenas um *expert* relevante foi incluído no *ranking* top-3. Nos três casos restantes, o modelo apresentou valores iguais a 0,0, evidenciando falha completa na recomendação. O HR@3 foi de 70%, ou seja, em sete dos dez testes o modelo conseguiu incluir ao menos um *expert* correto entre os três principais retornos. Embora esse valor seja inferior à taxa registrada pelo BERTimbau-base (90%).

Em relação às métricas de ordenação, o MRR@3 variou entre 0,0 e 1,0 ao longo dos experimentos. O valor máximo, 1,0, foi observado em 4 dos 10 casos avaliados, indicando que o *expert* mais relevante foi corretamente posicionado na primeira posição do *ranking* gerado pelo modelo, o cenário ideal para essa métrica. Por outro lado, o valor mínimo de 0,0 ocorreu em 3 dos 10 casos, refletindo situações em que nenhum dos *experts* relevantes apareceu entre os três primeiros retornos. Nos demais casos, o MRR@3 apresentou valor igual a 0,5 em um dos testes, indicando que o *expert* relevante foi posicionado na segunda posição. Em dois casos, o valor foi de 0,3333, o que significa que o *expert* relevante foi identificado apenas na terceira posição da lista de recomendações.

A métrica nDCG@3, apresentou valores que oscilaram entre 0,0 e 1,0. Em um caso, o valor foi de 1,0, indicando que todos os *experts* relevantes foram corretamente posicionados nas primeiras posições o cenário ideal de recomendação. Em outro caso, a métrica alcançou 0,7654, sinalizando que dois *experts* relevantes estavam nas posições 1º e 2º. Em um terceiro caso, o valor foi de 0,7039, o que também reflete a presença de dois *experts* relevantes, porém nas posições 1º e 3º resultando em uma pontuação ligeiramente menor devido à penalização da terceira posição. Um dos testes apresentou valor de 0,4693, revelando que o único *expert* relevante foi ranqueado na primeira posição. Em um caso, o valor foi de 0,2961, indicando acerto na 2º posição. Em outros dois casos, o valor registrado foi de 0,2346, sinalizando que o *expert* relevante foi posicionado na 3º posição. Por fim, em 3 dos 10 testes, a métrica resultou em 0,0, indicando que nenhum *expert* relevante foi recuperado no Top-3. A Figura 21 apresenta o total de acertos nas dez posições do *ranking*.

Figura 22 – Acerto por posição - Bertimbau-large



Fonte: Autoria própria (2025)

O modelo BERTimbau-large acertou quatro recomendações corretas na primeira posição e outras quatro na terceira, além de três acertos na segunda posição, obtendo aproximadamente 36,67% de acerto nas três primeiras posições. O modelo ainda acertou três recomendações na posição quatro, porém, sua maior quantidade de acertos ocorreu na posição cinco, com cinco recomendações. Na posição seis acertou apenas duas recomendações, seguido por três acertos em cada uma das posições sete, oito e nove, e na última posição do *ranking* não teve acerto. Aproximadamente 63,33% dos acertos concentram-se nas cinco primeiras posições do *ranking*.

Os resultados mostram que o BERTimbau-large apresenta melhor desempenho na ordenação dos *experts* relevantes quando acerta, superando o BERTimbau-base em métricas como nDCG@3 e MRR@3. No entanto, o BERTimbau-base demonstrou maior regularidade,

com desempenho superior em *Precision*@3 e HR@3, indicando maior frequência de acertos entre os três primeiros retornos.

5.1.1.4 Análise Individual: OpenAI-Small

O modelo OpenAI-small foi incluído nos experimentos como uma alternativa leve e de menor custo computacional, disponibilizada via API comercial da OpenAI. A Tabela 15 apresenta os resultados obtidos pelo modelo OpenAI-small.

Tabela 15 – Resultado Modelo OpenAI-Small

Pesquisadores	**Precision@3**	**HR@3**	**MRR@3**	**nDCG@3**
Pesquisador 1	0,3333	1	1,0	0,4693
Pesquisador 2	0,6667	1	1,0	0,7039
Pesquisador 3	0,3333	1	1,0	0,4693
Pesquisador 4	0,6667	1	1,0	0,7654
Pesquisador 5	1,0000	1	1,0	1,0000
Pesquisador 6	0,3333	1	1,0	0,4693
Pesquisador 7	0,3333	1	0,3333	0,2346
Pesquisador 8	0,3333	1	0,5	0,2961
Pesquisador 9	0,6667	1	0,5	0,5307
Pesquisador 10	0,0000	0	0,0	0,0000

Na métrica Precision@3, o modelo obteve desempenho variável, oscilando entre 0,0 e 1,0. Em um dos dez casos avaliados, o valor da precision@3 atingiu 1,0, o que indica que todos os três *experts* retornados foram relevantes. Em outros três casos, os valores de Precision@3 obtiveram 0,6667, o que corresponde à recuperação de dois *experts* relevantes entre os três recomendados. Já em cinco casos, precision@3 apresenta valor de 0,3333, representando um acerto entre os três retornos. Por fim, em um dos casos o modelo falhou, resultando em 0,0, refletindo a ausência de *experts* relevantes no top-3.

A HR@3 do modelo foi de 90%, o que indica que, em nove dos dez casos, ao menos um *expert* relevante foi incluído entre os três primeiros resultados. Essa taxa é equivalente à do modelo BERTimbau-base 90%, e ligeiramente superior à do BERTimbau-large 70%, o que evidencia uma boa capacidade de cobertura do modelo da OpenAI mesmo com menor complexidade computacional.

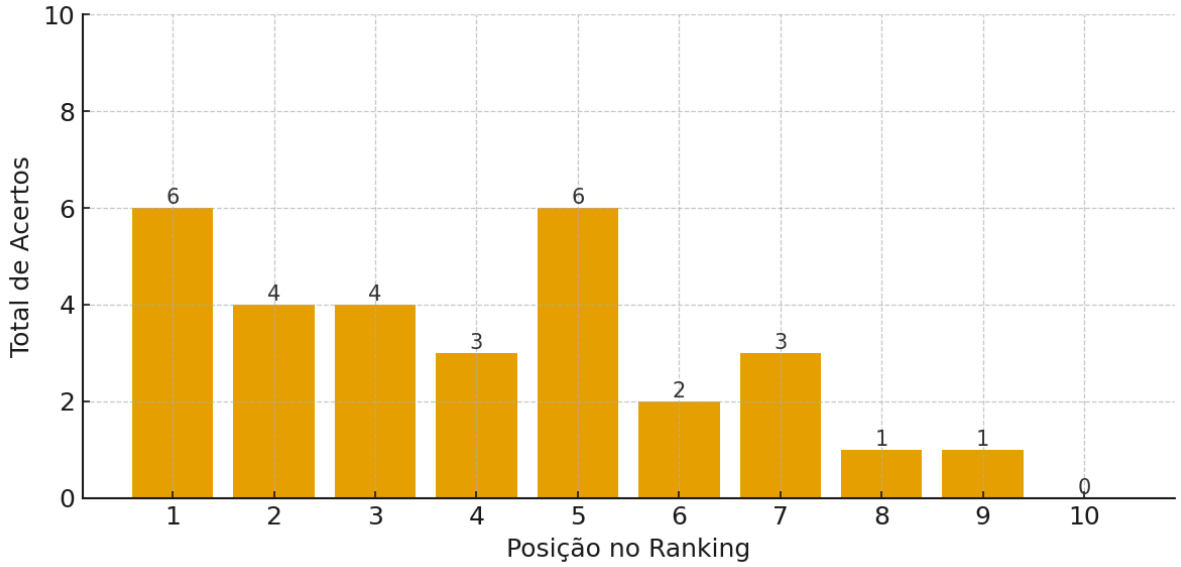
A métrica MRR@3, que avalia a posição do primeiro *expert* relevante no *ranking*, apresentou resultados consistentes na maioria dos testes. Em seis dos dez casos 60%, o valor registrado foi 1,0, indicando que o *expert* mais relevante foi corretamente posicionado na primeira posição do *ranking*, o melhor cenário possível para essa métrica. Esses resultados demonstram elevada precisão na ordenação por parte do modelo.

Em dois casos, o valor foi de 0,5, o que revela que o primeiro *expert* relevante apareceu

na segunda posição do *ranking*, ainda representando uma ordenação satisfatória. Já em um dos testes, o valor obtido foi 0,3333, indicando que o *expert* relevante foi localizado na terceira posição entre os retornos, o limite inferior aceitável para a métrica MRR@3. Por fim, em apenas um dos dez casos, a métrica apresentou valor igual a 0,0, evidenciando que nenhum *expert* relevante foi encontrado entre os três primeiros resultados, refletindo uma falha de recomendação neste cenário específico.

A métrica nDCG@3, apresentou valores que variaram entre 0,0 e 1,0 ao longo dos testes. Em um dos casos, o valor foi 1,0, indicando que todos os *experts* relevantes foram corretamente posicionados nas primeiras posições. Em um dos casos, o valor registrado foi 0,7654, o que demonstra que dois *experts* relevantes foram recuperados nas posições 1º e 2º. Em um dos casos, obteve o valor 0,7039, refletindo também a presença de dois *experts* relevantes, mas nas posições 1º e 3º, o que impacta levemente a pontuação devido à penalização da terceira posição. Além disso, em três dos dez casos, a métrica assumiu o valor 0,4693, revelando que o único *expert* relevante foi ranqueado na primeira posição. Em um caso, o valor foi 0,5307, indicando a presença de dois *experts* relevantes, indicando presença de *expert* nas 2º e 3º posição do *ranking*. Também foram observados valores de 0,2961, indicando apenas um *expert* na 2º posição do *ranking*. Em um dos casos obteve o valor de 0,2346, indicando um *expert* na 3º posição. Por fim, em apenas um dos casos a métrica foi 0,0, evidenciando a ausência total de *experts* relevantes entre os top-3. A Figura 23 apresenta o total de acertos nas dez posições do *ranking*.

Figura 23 – Acerto por posição - OpenAI-Small



Fonte: Autoria própria (2025)

O modelo OpenAI-small acertou seis recomendações corretas na primeira posição, seguido de quatro acertos na segunda e mais quatro na terceira posição, obtendo aproximadamente 46,67% de acerto nas três primeiras posições. O modelo ainda acertou três recomendações

corretas na posição quatro, mas apresentou um segundo pico de acertos na posição cinco, com seis recomendações. Na posição seis acertou duas recomendações, na posição sete obteve três acertos, e nas posições oito e nove teve um acerto em cada. Na última posição do *ranking* não houve acerto. Aproximadamente 76,67% dos acertos concentram-se nas cinco primeiras posições do *ranking*.

O OpenAI-small demonstrou desempenho superior aos modelos baseados em BERTimbau, tanto em cobertura quanto em ordenação.

5.1.1.5 Análise Individual: OpenAI-Large

O modelo OpenAI-large representa a versão mais robusta entre os avaliados, com maior número de parâmetros, maior capacidade de generalização e fornecido via API comercial. A Tabela 16 apresenta os resultados obtidos pelo modelo OpenAI-large.

Tabela 16 – Resultado Modelo OpenAI-Large

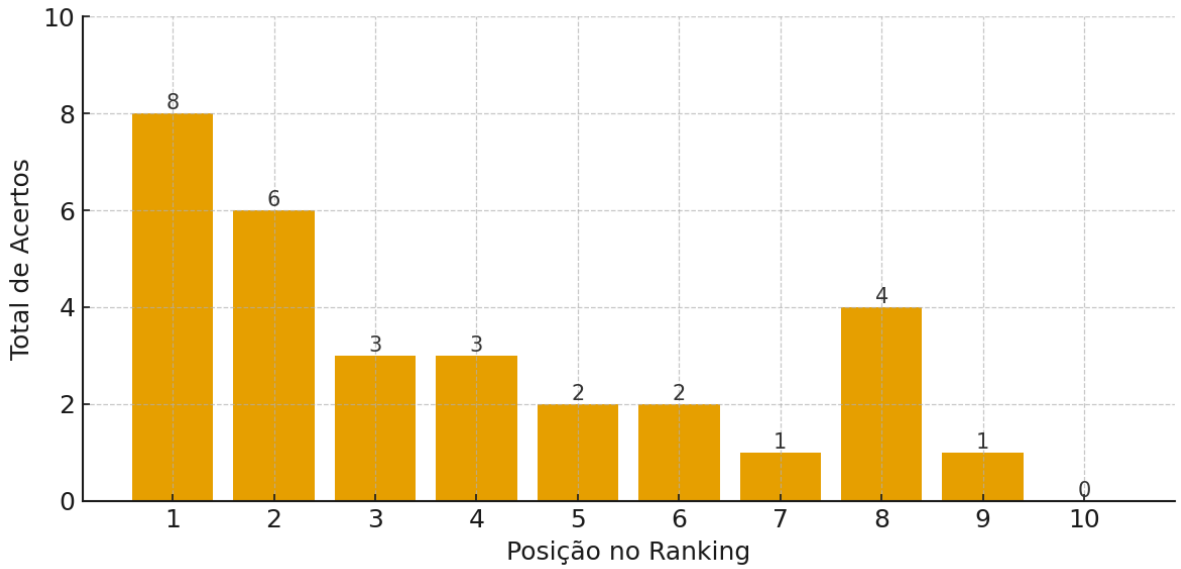
Pesquisadores	**Precision@3**	**HR@3**	**MRR@3**	**nDCG@3**
Pesquisador 1	0,6667	1	1,0	0,7654
Pesquisador 2	0,6667	1	1,0	0,7654
Pesquisador 3	0,3333	1	1,0	0,4693
Pesquisador 4	0,6667	1	1,0	0,7654
Pesquisador 5	1,0000	1	1,0	1,0000
Pesquisador 6	0,3333	1	1,0	0,4693
Pesquisador 7	0,6667	1	1,0	0,7039
Pesquisador 8	0,3333	1	0,5	0,2961
Pesquisador 9	0,6667	1	1,0	0,7654
Pesquisador 10	0,3333	1	0,3333	0,2346

A métrica *precision*@3, que avalia a proporção de *experts* relevantes entre os três retornados, apresentou variação entre 0,3333 e 1,0. Em um dos casos, o valor atingiu 1,0, indicando que todos os *experts* recomendados foram corretos. Em cinco dos dez casos, a precisão foi de 0,6667, demonstrando que dois entre os três retornos eram relevantes, o que representa um desempenho bastante satisfatório. Já em quatro dos casos, a métrica foi igual a 0,3333, mostrando que apenas um dos três *experts* retornados era relevante.

A HR@3 foi de 100%, confirmando que, em todos os casos, pelo menos um *expert* relevante foi incluído entre os três principais retornos. A métrica MRR@3, que mede a posição do primeiro *expert* relevante no *ranking*, apresentou desempenho elevado em quase todos os testes. Em oito dos dez casos, o valor foi 1,0, indicando que o *expert* mais relevante apareceu na primeira posição, o melhor cenário possível para essa métrica. Em um caso, o valor foi 0,5, refletindo que o *expert* correto apareceu na segunda posição. Por fim, em apenas um dos caso o valor foi 0,3333, o que indica que o *expert* relevante foi identificado apenas na terceira posição entre os retornos.

A métrica nDCG@3, apresentou valores entre 0,2346 e 1,0. Em quatro dos dez casos, os valores foram iguais a 0,7654, indicando que dois *experts* relevantes foram corretamente ranqueados nas posições 1º e 2º. Em um dos casos, o valor foi de 1,0, no qual os três *experts* mais relevantes foram posicionados nas três primeiras posições do *ranking*. Em um caso, o valor foi de 0,7039, o que sugere a presença de dois *experts* relevantes nas posições 1º e 3º. Em dois casos, o valor registrado foi de 0,4693, correspondendo ao acerto de um *expert* na 1º posição. Já os desempenhos mais baixos foram observados em um caso com valor 0,2961, indicando acerto na 2º posição, e em um caso com 0,2346, que reflete acerto na 3º posição do *ranking*. A Figura 24 presenta o total de acertos nas dez posições do *ranking* do modelo OpenAI-large.

Figura 24 – Acerto por posição - OpenAI-Large



Fonte: Autoria própria (2025)

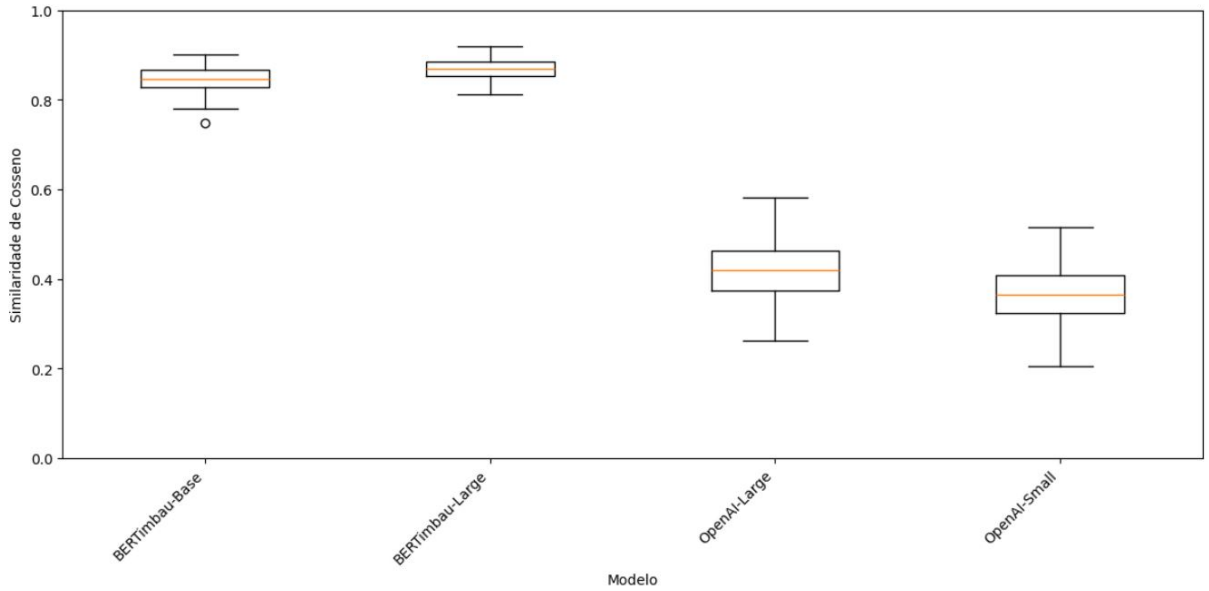
O modelo OpenAI-large obteve sua maior quantidade de acertos na primeira posição, com oito recomendações corretas, seguido de seis acertos na segunda e três na terceira posição, obtendo aproximadamente 56,67% de acerto nas três primeiras posições. O modelo ainda acertou três recomendações na posição quatro e duas nas posições cinco e seis. Na posição sete houve um acerto, mas o modelo apresentou uma recuperação na oitava posição, com quatro acertos, e teve mais um acerto na nona posição. Na última posição do *ranking* não houve acerto. Aproximadamente 73,33% dos acertos concentram-se nas cinco primeiras posições do *ranking*.

O modelo OpenAI-large apresentou os melhores resultados entre os modelos avaliados, com maiores valores de Precision@3, MRR, nDCG e HR. Esses indicadores sugerem maior efetividade na identificação em cobertura e ordenação dos *experts* mais relevantes.

5.1.2 Análise das Similaridades de Cosseno

A análise de similaridade de cosseno teve como objetivo medir a proximidade semântica entre os documentos dos pesquisadores e os perfis dos *experts*. Para cada um dos 10 pesquisadores selecionados, o sistema gerou um *ranking* contendo os 10 *experts* ordenados de acordo com seus respectivos valores de similaridade. Isso resultou em um total de 100 comparações por modelo avaliado. A Figura 25 apresenta a distribuição dos valores de similaridade de cosseno para os modelos analisados.

Figura 25 – Distribuição dos valores de similaridade de cosseno entre pesquisadores e *experts*, por modelo



Fonte: Autoria própria (2025)

Observa-se que os modelos baseados em BERTimbau apresentaram valores de similaridade mais elevados e menos dispersos, concentrando-se na faixa entre 0,80 e 0,90, com poucos *outliers*. Isso indica maior consistência e proximidade semântica entre os pares recomendados por esses modelos. Os modelos da OpenAI apresentaram distribuições mais amplas e medianas mais baixas. O modelo OpenAI-large exibiu uma concentração de valores de similaridade entre aproximadamente 0,35 e 0,55, indicando uma maior variabilidade na qualidade das correspondências semânticas geradas.

5.1.2.1 Modelo BERTimbau-Base

O modelo BERTimbau-base apresentou desempenho moderado, com valores de similaridade de cosseno variando entre 0,7475 e 0,9018. A média geral de similaridade entre todos os 100 pares (10 pesquisadores × 10 *experts*) foi aproximadamente 0,842, indicando que o modelo foi capaz de identificar relações semânticas razoáveis, embora com menor precisão em comparação a modelos mais avançados.

A distribuição dos valores mostra que a maior parte dos scores está concentrada entre 0,80 e 0,88, com poucos pares ultrapassando a marca de 0,89. O valor máximo de similaridade foi observado no par entre a pesquisadora 4 e o *expert* 4, com um *score* de 0,9018, representando um dos poucos casos com alinhamento semântico muito forte. Em contraste, o menor valor foi registrado no par Pesquisador 7 com *Expert* 10, com 0,7475, refletindo uma correspondência de baixa relevância temática.

5.1.2.2 Modelo BERTimbau-Large

No experimento conduzido com o modelo BERTimbau-large, cada um dos 10 pesquisadores teve suas respectivas similaridades calculadas com os 10 *experts*, totalizando 100 combinações. A análise dos valores de similaridade de cosseno revelou um desempenho superior ao da versão Base, com *scores* mais concentrados em faixas elevadas e maior estabilidade na ordenação dos candidatos mais relevantes.

A média geral de similaridade entre os pares foi de aproximadamente 0,8721, evidenciando que o modelo capturou relações semânticas com maior refinamento em comparação à versão BERTimbau-base. O valor mais alto observado foi 0,9196, correspondente ao par entre Pesquisador 4 e *Expert* 2, refletindo forte alinhamento temático entre o conteúdo textual analisado.

Entre os 100 pares, mais da metade apresentou similaridades superiores a 0,87, e cerca de 30% ultrapassaram a marca de 0,89. Essa concentração elevada indica que o modelo conseguiu diferenciar os conteúdos com maior granularidade sem comprometer sua consistência. Além disso, a ordenação dos *experts* mais bem ranqueados manteve-se estável para a maioria dos pesquisadores, o que reforça a robustez do modelo.

No entanto, apesar da melhora nos *scores* em relação à versão BERTimbau-base, o BERTimbau-large ainda apresentou algumas limitações em contextos mais específicos, com certos pares atingindo valores abaixo de 0,83, como o caso do pesquisador 7 que teve a menor similaridade registrada em 0,8187 com o *expert* 8.

5.1.2.3 Modelo OpenAI-Small

O modelo OpenAI-small apresentou resultados satisfatórios na ordenação dos *experts* com base na similaridade de cosseno. A hierarquia entre os *experts* foi bem estabelecida, com os primeiros colocados geralmente apresentando maiores valores de similaridade em relação aos últimos, o que indica a capacidade do modelo em capturar relações semânticas relevantes entre os pares pesquisador–*expert*.

Entre os 10 pesquisadores avaliados, os maiores valores de similaridade foram geralmente associados aos *experts* mais relevantes, embora com uma margem menor em relação aos demais pares. O melhor resultado foi obtido para a pesquisadora 5, com valor de similaridade de aproximadamente 0,52 com a *expert* 9. Por outro lado, o pior caso de *matching* foi observado

para a pesquisadora 6, cuja correspondência com *expert* 10 apresentou uma similaridade de apenas 0,21.

Contudo, ao comparar os valores absolutos de similaridade com aqueles obtidos pelos modelos baseados no BERTimbau, observa-se que o OpenAI-small obteve, em média, pontuações mais baixas. As similaridades variaram entre aproximadamente 0,22 e 0,51, sendo a maioria dos valores concentrada na faixa de 0,30 a 0,40. Em contraste, os modelos BERTimbau-base e BERTimbau-large apresentaram, em geral, valores de similaridade mais elevados para os *experts* mais relevantes.

5.1.2.4 Modelo OpenAI-Large

O modelo OpenAI-large apresentou resultados consistentes na tarefa de *matching*, com valores de similaridade de cosseno distribuídos de maneira homogênea ao longo dos *rankings*. Embora não tenha atingido os maiores valores absolutos entre os modelos avaliados, sua performance foi marcada por uma boa coerência entre os *experts* mais relevantes e os primeiros colocados nas listas de recomendação.

Como no caso do pesquisador 1, que obteve uma similaridade de 0,5824 com o *expert* 3, ocupando a primeira posição do *ranking* um dos maiores valores registrados pelo modelo. A menor similaridade com valor observado foi identificada na correspondência entre o pesquisador 7 e o *expert* 10, com uma similaridade de apenas 0,2611, indicando uma fraca relação semântica neste caso específico.

De modo geral, os valores de similaridade do modelo OpenAI-large variaram entre 0,26 e 0,58, com a maioria das pontuações concentradas na faixa de 0,40 a 0,50. A ordenação interna dos *rankings* mostrou-se estável, com uma queda progressiva das similaridades a partir das posições intermediárias. Ainda que o modelo OpenAI-large não tenha superado os modelos baseados no BERTimbau em termos de valores absolutos, sua consistência e equilíbrio na distribuição das similaridades o tornam uma alternativa viável e eficiente para tarefas de *matching* semântico com custo computacional moderado.

5.1.3 Comparação Entre os Modelos

A análise comparativa permite agrupá-los em três categorias distintas com base em seu comportamento e performance:

- Grupo 1 — Modelos (BERTimbau-base e BERTimbau-large): os modelos baseados na arquitetura BERTimbau apresentaram desempenho inferior quando comparados aos modelos OpenAI-small e OpenAI-large. O BERTimbau-base obteve médias de *Precision*@3 de 0,4333, superando ligeiramente o BERTimbau-large, que obteve 0,3667 nessas mesmas métricas. Apesar disso, o modelo BERTimbau-large apresentou desempenho levemente

superior nas métricas de ordenação, com MRR@3 de 0,5167 contra 0,4667 do BERTimbau-base. No entanto, ambos apresentaram nDCG@3 próximos (0,3704 no BERTimbau-large e 0,3715 no BERTimbau-base). A HR foi de 90% para o Base e 70% para o BERTimbau-large , indicando maior consistência do modelo menor, contrariando a expectativa inicial de que o modelo mais complexo teria melhor desempenho.

- Grupo 2 — OpenAI-small: este modelo apresentou médias de *Precision*@3 de 0,4667, equivalentes às obtidas pelo BERTimbau-base. No entanto, destacou-se nas métricas de ranqueamento, com MRR@3 de 0,7333 e nDCG@3 de 0,4939, indicando que foi mais eficaz ao ordenar os *experts* relevantes nas primeiras posições do *ranking*. A HR de 90% reforça a confiabilidade do modelo, mesmo sendo uma alternativa computacionalmente mais leve.

- Grupo 3 — OpenAI-large: este modelo apresentou desempenho significativamente superior em todas as métricas avaliadas. Com Precision@3 de 0,5667, MRR@3 de 0,8833 e nDCG@3 de 0,6235, foi o único a atingir 100% de acerto no Top-3. Isso evidencia sua elevada capacidade de identificar e ranquear corretamente os *experts* mais relevantes, o que o torna a opção mais robusta para aplicações de alta exigência em precisão e qualidade de recomendação.

A análise sugere que o aumento da capacidade dos modelos, aliado ao uso de arquiteturas mais modernas e pré-treinadas em conjuntos de dados amplos e diversificados, como é o caso dos modelos da OpenAI, resulta em ganhos significativos de desempenho. A discrepância observada entre os modelos BERTimbau e os modelos da OpenAI pode ser atribuída à maior generalização semântica, maior escala de treinamento e suporte multilíngue presentes nos modelos comerciais.

Em síntese, os resultados comparativos indicam que o modelo OpenAI-large representa a escolha mais indicada quando se busca excelência em cobertura e ranqueamento. O OpenAI-small surge como uma solução de compromisso eficiente, oferecendo desempenho superior aos modelos BERTimbau com menor custo computacional. Já os modelos BERTimbau, embora úteis, apresentaram limitações em termos de cobertura e ordenação dos *experts* mais relevantes, especialmente em contextos com maior complexidade semântica.

5.1.4 Análise estatística na análise dos resultados

Foram avaliadas as métricas *Precision*@3, MRR@3, nDCG@3 e HR@3, obtidas nos quatro modelos de *matching* BERTimbau-base, BERTimbau-large, OpenAI-small e OpenAI-large. O objetivo da análise estatística foi verificar se as diferenças de desempenho entre os modelos são estatisticamente significativas e identificar quais modelos apresentaram desempenho superior.

Inicialmente, para cada métrica, foram realizados testes de normalidade *Shapiro-Wilk* e homogeneidade de variâncias *Brown-Forsythe*.

Quando os dados atenderam aos pressupostos de normalidade e homogeneidade, aplicou-se a Análise de Variância de Medidas Repetidas *ANOVA One-Way Repeated Measures*.

Quando os pressupostos não foram atendidos no caso do HR@3, utilizou-se o teste não paramétrico de *Friedman*.

Nos casos em que o teste principal indicou diferença significativa ($p < 0{,}05$), foi aplicado um teste pós-hoc de comparações múltiplas pelo método *Holm–Šidák*, adequado por apresentar maior poder estatístico para detectar diferenças entre grupos.

Todos os testes estatísticos foram executados com o auxílio do *software SigmaPlot* versão 13[1].

5.1.5 Análise estatística aplicada à *Precision@3*

A Tabela 17 apresenta a verificação dos pressupostos para a aplicação da ANOVA de medidas repetidas. O teste de normalidade de *Shapiro-Wilk* foi aprovado ($P = 0{,}872$) e o teste de homogeneidade de variâncias de *Brown-Forsythe* também foi aprovado ($P = 0{,}517$). Esses resultados justificam o uso do modelo paramétrico com medidas repetidas. Em todas as condições experimentais, utilizou-se a mesma amostra de dez pesquisadores (N=10), e cada um deles foi avaliado em todos os quatro modelos.

Tabela 17 – Verificação de pressupostos para *Precision@3*.

Teste	***P*-valor**
Normality Test (Shapiro–Wilk)	$P = 0{,}872$
Equal Variance Test (Brown–Forsythe)	$P = 0{,}517$

A Tabela 18 resume as estatísticas descritivas por tratamento. Observa-se que as médias de *Precision*@3 variaram de 0,367 BERTimbau-large a 0,567 OpenAI-large. Os desvios-padrão situaram-se entre 0,225 e 0,331, e os erros-padrão da média (*SEM*) entre 0,0712 e 0,105, indicando variação moderada no desempenho entre os sujeitos avaliados para cada modelo.

Tabela 18 – Estatísticas descritivas reportadas para *Precision@3*.

Modelo	**N**	**Média**	**Desvio-padrão**	**SEM**
BERTimbau-base	10	0,433	0,225	0,0712
BERTimbau-large	10	0,367	0,331	0,105
OpenAI-small	10	0,467	0,281	0,0889
OpenAI-large	10	0,567	0,225	0,0712

A Tabela 19 apresenta os resultados da *ANOVA* para a métrica *Precision*@3. O efeito de *Between Treatments* não foi estatisticamente significativo, com $F(3, 27) = 1{,}626$ e $P = 0{,}206$.

[1] <https://systatsoftware.com/products/sigmaplot/>

Assim, as diferenças observadas entre as médias dos quatro modelos não foram suficientes para rejeitar a hipótese nula de igualdade entre os tratamentos. Dado o resultado global não significativo, não se faz necessária a aplicação de procedimentos pós-hoc para esta métrica.

Tabela 19 – *Resultado ANOVA* aplicada à *Precision*@3.

Fonte de variação	**DF**	**SS**	**MS**	**F**	**P**
Between Subjects	9	1,458	0,162		
Between Treatments	3	0,208	0,0695	1,626	0,206
Residual	27	1,153	0,0427		
Total	39	2,820			

As diferenças entre as médias dos grupos de tratamento não foram grandes o suficiente para excluir a possibilidade de que sejam devidas à variabilidade amostral aleatória; portanto, não há diferença estatisticamente significativa (P = 0,206). O poder do teste realizado com α = 0,050 foi de 0,163, valor abaixo do poder desejado de 0,800. Um poder estatístico reduzido indica menor probabilidade de detectar uma diferença real, caso ela exista, o que sugere cautela na interpretação de resultados negativos. Dessa forma, a análise indica que os modelos avaliados apresentaram desempenho semelhante em termos de *Precision*@3 dentro da amostra considerada.

5.1.6 Análise estatística aplicada à *HR@3*

O teste de normalidade de *Shapiro-Wilk* indicou violação do pressuposto de normalidade (P < 0,050), impossibilitando a aplicação da ANOVA de medidas repetidas. Diante disso, utilizou-se o teste não paramétrico de *Friedman*, adequado para comparações entre medidas repetidas quando os dados não seguem distribuição normal.

A Tabela 20 apresenta as estatísticas descritivas em termos de mediana e quartis (25% e 75%) para cada modelo. Observa-se saturação de desempenho, uma vez que para três configurações a mediana foi igual a 1,000, com primeiro e terceiro quartis também em 1,000. Apenas o modelo BERTimbau-large apresentou maior dispersão, com mediana em 1,000 e primeiro quartil em 0,000, indicando leve variação entre os sujeitos.

Tabela 20 – Mediana e quartis por grupo para *HR@3*.

Grupo	**N**	**Mediana**	**25%**	**75%**
HR@3 BERTimbau-base	10	1,000	1,000	1,000
HR@3 BERTimbau-large	10	1,000	0,000	1,000
HR@3 OpenAI-small	10	1,000	1,000	1,000
HR@3 OpenAI-large	10	1,000	1,000	1,000

A Tabela 21 apresenta o resultado do teste de *Friedman*. O teste indicou ausência de diferenças estatisticamente significativas entre os modelos para HR@3, com χ^2 = 4,385,gl

= 3 e $P = 0{,}223$. As diferenças observadas entre as medianas dos grupos de tratamento não são suficientemente grandes para excluir a possibilidade de que sejam devidas à variabilidade amostral aleatória. Dessa forma, conclui-se que não há diferença estatisticamente significativa entre os modelos quanto à métrica HR@3.

Tabela 21 – Resultado do teste de *Friedman* para *HR@3*.

Teste	**Estatística**	**gl**	***P*-valor**
Friedman	$chi^2 = 4{,}385$	3	0,223

Os resultados sugerem que todos os modelos atingiram desempenho semelhante em termos de HR@3 nos três primeiros resultados retornados, caracterizando uma saturação do desempenho, possivelmente associada à elevada taxa de acerto geral dos modelos nesta métrica.

5.1.7 Análise estatística aplicada à *MRR@3*

Com a mesma amostra sendo dez pesquisadores que foram avaliados nos quatro modelos de *matching*, os resultados de desempenho foram comparados utilizando a Análise de Variância (ANOVA) de medidas repetidas. Os pressupostos para a abordagem paramétrica foram atendidos normalidade de *Shapiro-Wilk*, $P = 0{,}224$; homogeneidade de variâncias de *Brown-Forsythe*, $P = 0{,}382$), permitindo a aplicação da ANOVA de medidas repetidas. A Tabela 22 apresenta as estatísticas descritivas por modelo para a métrica MRR@3.

Tabela 22 – Estatísticas descritivas por modelo para *MRR*@3.

Modelo	**N**	**Média**	**Desvio-padrão**	**SEM**
BERT-base	10	0,467	0,246	0,0778
BERT-large	10	0,517	0,448	0,1420
OpenAI-small	10	0,733	0,370	0,1170
OpenAI-large	10	0,883	0,249	0,0788

A Tabela 23 apresenta os resultados da ANOVA de medidas repetidas para MRR@3. O efeito de *Between Treatments* corresponde ao fator Modelo, enquanto Residual representa a variação não explicada dentro dos mesmos sujeitos e *Between Subjects* refere-se às diferenças entre os dez pesquisadores. Observou-se efeito significativo do fator *Modelo*, com $F(3, 27) = 4{,}110$ e $P = 0{,}016$, indicando que as médias de MRR@3 diferem significativamente entre os modelos. O poder observado foi 0,659, valor abaixo do ideal (0,80), sugerindo que as conclusões devem ser interpretadas com cautela, embora o resultado justifique a realização de comparações pós-hoc para identificar quais pares diferem.

Tabela 23 – ANOVA de medidas repetidas para *MRR@3*.

Fonte de variação	**gl**	**SQ (SS)**	**QM (MS)**	*F*	*P*
Between Subjects	9	1,670	0,186		
Between Treatments	3	1,128	0,376	4,110	0,016
Residual	27	2,469	0,0915		
Total	39	5,267			

Após o efeito global significativo, aplicou-se o teste pós-hoc de comparações múltiplas *Holm–Šidák* para identificar quais pares de modelos apresentaram diferenças relevantes, conforme Tabela 24. A coluna (Dif. de médias) indica a diferença entre as médias do primeiro e do segundo modelo, t é a estatística do teste e P é o valor ajustado. O nível de significância global foi mantido em $\alpha = 0{,}05$. A Tabela 24 apresenta as comparações pareadas entre os modelos.

Tabela 24 – Comparações pareadas entre modelos (*Holm–Šidák*) para *MRR@3*.

Comparação	**Dif. de médias**	*t*	*P*	**Significativo**
OpenAI-large vs. BERT-base	0,417	3,081	0,028	Sim
OpenAI-large vs. BERT-large	0,367	2,711	0,056	Não
OpenAI-small vs. BERT-base	0,267	1,972	0,216	Não
OpenAI-small vs. BERT-large	0,217	1,602	0,320	Não
OpenAI-large vs. OpenAI-small	0,150	1,109	0,478	Não
BERT-large vs. BERT-base	0,050	0,370	0,715	Não

Os resultados evidenciam efeito global significativo do fator Modelo sobre MRR@3, com desempenho superior do modelo OpenAI-large em relação ao BERTimbau-base após correção de *Holm–Šidák* ($P = 0{,}028$). As demais comparações não atingiram significância estatística ($P > 0{,}05$), embora se observe tendência de vantagem do OpenAI-large frente ao BERTimbau-large ($P = 0{,}056$). Esses achados sugerem que as configurações baseadas em OpenAI-large apresentam maior capacidade de posicionar, entre as primeiras posições do *ranking*, pelo menos um *expert* relevante para o pesquisador, quando comparadas às variantes BERTimbau.

5.1.8 Análise estatística aplicada à *nDCG@3*

Foram avaliados os mesmos dez pesquisadores nos quatro modelos, os resultados de desempenho foram comparados utilizando a Análise de Variância (ANOVA) de medidas repetidas. Os pressupostos para a análise paramétrica foram atendidos, com normalidade aprovada *Shapiro-Wilk*, ($P = 0{,}680$) e homogeneidade das variâncias confirmada *Brown-Forsythe*, ($P = 0{,}595$), o que permitiu a aplicação da ANOVA de medidas repetidas. A Tabela 25 apresenta as estatísticas descritivas por modelo para a métrica *nDCG@3*.

Tabela 25 – Estatísticas descritivas por modelo para *nDCG@3*.

Modelo	**N**	**Média**	**Desvio-padrão**	**SEM**
BERTimbau-base	10	0,372	0,217	0,0686
BERTimbau-large	10	0,370	0,354	0,1120
OpenAI-small	10	0,494	0,284	0,0899
OpenAI-large	10	0,623	0,244	0,0771

A Tabela 26 apresenta os resultados da ANOVA de medidas repetidas aplicada à métrica nDCG@3. O fator Modelo *Between Treatments* apresentou efeito estatisticamente significativo ($P = 0{,}041$), indicando que as médias de desempenho dos modelos diferem para essa métrica. A variação entre os participantes *Between Subjects* e a variação residual Residual representam, respectivamente, diferenças individuais entre os pesquisadores e a variabilidade não explicada. Esses resultados indicam que os modelos não possuem o mesmo nível de qualidade de ranqueamento, justificando a realização das comparações pós-hoc para identificar quais pares diferem entre si.

Tabela 26 – ANOVA de medidas repetidas para *nDCG*@3.

Fonte de variação	**gl**	**SQ (SS)**	**QM (MS)**	*F*	*P*
Between Subjects	9	1,570	0,174		
Between Treatments	3	0,436	0,145	3,159	0,041
Residual	27	1,243	0,0460		

O poder do teste realizado foi de 0,489, valor inferior ao poder ideal de 0,80, o que indica que os resultados devem ser interpretados com moderação devido ao tamanho amostral reduzido. Apesar disso, a presença de efeito global significativo evidencia diferenças relevantes entre os modelos avaliados.

A Tabela 27 apresenta as comparações pareadas realizadas com a correção de *Holm–Šidák*, utilizada para manter o nível de significância global em ($\alpha = 0{,}05$). Embora o efeito global tenha sido significativo, nenhuma comparação pareada atingiu significância estatística ($P > 0{,}05$), sugerindo que as diferenças observadas entre os modelos não foram suficientemente grandes para caracterizar distinções estatisticamente robustas entre pares específicos.

Tabela 27 – Comparações pareadas entre modelos (*Holm–Šidák*) para *nDCG@3*.

Comparação	**Dif. de médias**	*t*	*P*	**Significativo**
OpenAI-large vs. BERT-large	0,253	2,637	0,079	Não
OpenAI-large vs. BERT-base	0,252	2,626	0,068	Não
OpenAI-large vs. OpenAI-small	0,130	1,351	0,565	Não
OpenAI-small vs. BERT-large	0,123	1,287	0,505	Não
OpenAI-small vs. BERT-base	0,122	1,275	0,381	Não
BERT-base vs. BERT-large	0,001	0,012	0,991	Não

Apesar de nenhuma comparação individual ter sido significativa, observa-se tendência consistente de melhores médias para o modelo OpenAI-large, sugerindo maior capacidade deste modelo em ranquear os *experts* relevantes nas primeiras posições do *ranking*. Essa tendência, combinada ao efeito global significativo, reforça a hipótese de que modelos com *embeddings* mais densos e contextualizados apresentam vantagem na métrica nDCG@3, refletindo melhor ordenação semântica nas correspondências entre pesquisadores e *experts*.

5.1.9 Análise estatística aplicada à Similaridade do Cosseno

A análise estatística aplicada aos resultados obtidos da métrica de Similaridade do Cosseno que foi conduzida com base nos valores obtidos para os quatro modelos de *matching* BERTimbau-base, BERTimbau-large, OpenAI-small e OpenAI-large, considerando cem combinações entre pesquisadores e *expert* para cada modelo ou seja, para cada modelo o *dataset* pesquisadores tinha 10 amostras e para o *dataset experts* tinha 10 amostras totalizando cem combinações possíveisis por cada modelo. O objetivo foi verificar se havia diferenças estatisticamente significativas entre as distribuições de similaridade geradas por cada modelo.

O teste de normalidade de *Shapiro-Wilk* indicou violação do pressuposto de normalidade ($P < 0{,}050$), o que inviabilizou o uso da ANOVA paramétrica. Diante disso, aplicou-se o teste não paramétrico de *Friedman*, apropriado para comparações entre condições dependentes quando os dados não seguem distribuição normal.

A Tabela 28 apresenta as estatísticas descritivas aplicadas à metrica de Similaridade do Cosseno em termos de mediana e quartis (25% e 75%) para cada modelo. Observa-se que as arquiteturas baseadas em BERTimbau apresentaram medianas consideravelmente mais altas que as variantes OpenAI, indicando maior concentração de valores de similaridade entre os vetores de pesquisadores e *experts*.

Tabela 28 – Estatísticas descritivas da Similaridade do Cosseno

Modelo	**N**	**Mediana**	**25%**	**75%**
BERTimbau-base	100	0,847	0,827	0,866
BERTimbau-large	100	0,870	0,853	0,884
OpenAI-small	100	0,365	0,324	0,408
OpenAI-large	100	0,420	0,372	0,463

A Tabela 29 apresenta o teste de *Friedman*, que indicou diferença estatisticamente significativa entre os modelos ($\chi^2 = 291{,}000$; $gl = 3$; $P < 0{,}001$), confirmando que as distribuições de similaridade não são equivalentes . Esse resultado rejeita a hipótese nula de igualdade das medianas e demonstra que os modelos de linguagem analisados produzem padrões distintos de proximidade vetorial.

Tabela 29 – Resultado do teste de *Friedman* para *Similaridade do Cosseno*.

Teste	**Estatística**	**gl**	***P*-valor**
Friedman	$\chi^2 = 291{,}000$	3	$< 0{,}001$

Para identificar quais pares de modelos diferiram significativamente, aplicou-se o teste pós-hoc de comparações múltiplas de *Tukey* conforme resultado estão apresentados na tabela 30. Os resultados mostraram diferenças significativas ($P < 0{,}001$) em todas as comparações.

Tabela 30 – Comparações pareadas entre modelos (*Tukey*) para *Similaridade do Cosseno*.

Comparação	*Diff of Ranks*	*q*	***P***	$P < 0{,}050$
BERTimbau-large vs OpenAI-small	292,000	22,618	$< 0{,}001$	Sim
BERTimbau-large vs OpenAI-large	206,000	15,957	$< 0{,}001$	Sim
BERTimbau-large vs BERTimbau-base	98,000	7,591	$< 0{,}001$	Sim
BERTimbau-base vs OpenAI-small	194,000	15,027	$< 0{,}001$	Sim
BERTimbau-base vs OpenAI-large	108,000	8,366	$< 0{,}001$	Sim
OpenAI-large vs OpenAI-small	86,000	6,662	$< 0{,}001$	Sim

Os resultados revelam que todos os modelos diferiram significativamente entre si quanto à Similaridade do Cosseno. As arquiteturas BERTimbau apresentaram medianas substancialmente mais elevadas, refletindo maior proximidade vetorial média entre textos de pesquisadores aos *experts*. Esses achados evidenciam que a maior densidade vetorial dos modelos BERTimbau não necessariamente se traduz em melhor ordenação semântica, enquanto os *embeddings* mais dispersos dos modelos OpenAI favorecem a distinção entre conceitos e a precisão do ranqueamento nas tarefas de *matching*.

5.1.10 Eficiência Computacional dos Modelos

A avaliação da eficiência computacional dos modelos de *matching* foi conduzida considerando o tempo médio de execução para o processamento das consultas, a infraestrutura utilizada e os custos operacionais associados a cada abordagem. O objetivo desta análise é fornecer subsídios técnicos para a escolha de arquiteturas adequadas a diferentes contextos de produção, levando em conta restrições de tempo, orçamento e recursos computacionais.

Os modelos BERTimbau-base e BERTimbau-large foram executados localmente em ambiente de testes controlado, utilizando o *Google Colab* (sem suporte a GPU), com 12 GB de memória RAM e 107 GB de armazenamento disponível. Os tempos médios observados foram de aproximadamente 2,24 minutos para o BERTimbau-base e 6,37 minutos para o BERTimbau-large. Essa diferença reflete o impacto do número de camadas e parâmetros sobre o tempo de inferência, sendo o modelo Large aproximadamente três vezes mais lento que a versão Base.

Por outro lado, os modelos OpenAI-small e OpenAI-large foram utilizados via chamadas à API oficial da OpenAI. A geração dos *embeddings* ocorreu em infraestrutura remota, enquanto o pré-processamento textual, o envio das requisições e o armazenamento dos vetores resultantes foram realizados localmente no *Google Colab*, também sem o uso de GPU. Na Tabela 31, apresenta-se a comparação da eficiência computacional entre os modelos avaliados.

Tabela 31 – Comparação de eficiência computacional dos modelos

Modelo	**Tipo de Execução**	**Tempo Estimado**	**Observações**
BERTimbau-Base	Local (sem GPU)	~ 2,24 min	Baixo custo computacional local; execução estável
BERTimbau-Large	Local (sem GPU)	~ 6,37 min	Tempo mais elevado devido ao maior número de parâmetros
OpenAI-Small	API externa	~ 15 seg	Baixa latência; adequado para aplicações em tempo real
OpenAI-Large	API externa	~ 25 seg	Latência moderada; indicado para grandes volumes em lote

Em termos de custo, o modelo OpenAI-small utilizado via OpenAI apresenta um valor de aproximadamente US$ 0,02 por 1 milhão de *tokens* para o modelo OpenAI-large US$ 0,13 por 1 milhão de *tokens*, estimou-se o custo do processamento dos documentos utilizados na validação. Aproximadamente 49.699 *tokens* correspondentes aos 10 documentos dos pesquisadores, o custo total foi de aproximadamente US$ 0,001 utilizando o modelo OpenAI-small, e US$ 0,006 utilizando o modelo OpenAI-large. Para o conjunto de textos associados aos 10 especialistas, o custo foi inferior a US$ 0,001 em ambos os modelos. Esses valores são aplicáveis ao total de *tokens* enviados por requisição, o que inclui o texto original após pré-processamento. Assim, o custo total por documento está diretamente relacionado ao seu tamanho, tornando o modelo OpenAI-small mais econômico para aplicações com alta demanda. Os valores de custo por *token* utilizados para os cálculos foram obtidos junto à plataforma da OpenAI. Cabe destacar que tais valores correspondem ao momento da realização desta pesquisa, podendo sofrer alterações conforme política de preços da OpenAI.

Embora os modelos da OpenAI apresentem menor tempo de execução do que os modelos locais. No entanto, essa abordagem requer conectividade com a internet, acesso à API e está sujeita à cobrança por uso por *tokens* processado, o que pode impactar o orçamento em projetos de larga escala.

Já os modelos BERTimbau, apesar do tempo maior de execução, oferecem total controle sobre o ambiente de execução e não envolvem custos adicionais por requisição, sendo uma alternativa para ambientes com restrições de privacidade ou orçamentos limitados.

Portanto, a escolha do modelo mais adequado deve considerar o equilíbrio entre tempo de resposta, custo por uso, escalabilidade, necessidade de controle e os requisitos de precisão

definidos para a tarefa de *matching*.

5.2 Conexão com os Objetivos da Pesquisa

Os resultados obtidos confirmam o primeiro objetivo desta pesquisa: desenvolver um modelo de *matching* para recomendar os *experts* mais adequados para os pesquisadores, com base em análise semântica. O uso de *embeddings* contextuais mostrou-se superior às abordagens clássicas e demonstrou viabilidade prática ao simular com precisão a escolha realizada por *experts* humanos na etapa de *matching* da SciBees.

O segundo objetivo, relacionado à análise da eficiência computacional e escalabilidade, também foi contemplado. O tempo médio de execução por consulta nos modelos OpenAI variou entre 15 segundos para a versão OpenAI-small e 25 segundos para a versão OpenAI-large, o que os torna adequados para uso em fluxos operacionais contínuos. Os modelos locais BERTimbau, por outro lado, requerem maior tempo de processamento, sem apresentar ganho proporcional de performance.

5.3 Implicações Práticas para a Plataforma SciBees

A adoção do modelo OpenAI-large permite automatizar parcialmente as etapas de análise da demanda, identificação de área de conhecimento, habilidades e experiência técnica. Isso resulta em:

- Redução significativa do tempo de triagem de *experts*;
- Maior padronização e replicabilidade das recomendações;
- Liberação da equipe de gestão para atuar em tarefas de maior valor agregado.

Por outro lado, a dependência de uma API externa e os custos por volume de consulta devem ser considerados em um eventual plano de escalonamento. O modelo OpenAI-small pode se apresentar como uma alternativa viável em cenários de demanda com menor criticidade.

5.4 Limitações

Apesar dos resultados promissores, o estudo apresenta algumas limitações importantes que devem ser consideradas:

- A validação foi realizada com um conjunto reduzido de referência *ground truth* composto por 10 documentos. Essa limitação decorre do elevado esforço técnico exigido no processo manual de *matching*, atualmente conduzido pela equipe especializada da SciBees.

- Embora os dados utilizados representem o cenário real de funcionamento da SciBees, o conjunto de *ground truth* desta pesquisa não contemplou todos os 513 documentos disponíveis na base de pesquisadores. Assim, futuros experimentos podem ampliar esse conjunto de validação, permitindo avaliações mais completa.

- A avaliação do modelo concentrou-se nos critérios 1 (área de conhecimento), 2 (experiência) e 3 (habilidades técnicas) do processo manual da SciBees. Para os trabalhos futuros pode ser incluidos os critérios demais critérios, 4 (histórico de *feedbacks*), 5 (estilo de orientação) e 6 (disponibilidade).

- Estudos futuros podem incluir múltiplas iterações e ampliar o conjunto de casos utilizados para validação, com o intuito de fortalecer, de forma mais rigorosa, a robustez estatística do modelo.

- A dependência de modelos externos, como os da OpenAI, implica custos financeiros por uso da API, além de possíveis limitações de privacidade e controle sobre os dados processados.

6

Considerações Finais

Esta dissertação teve como objetivo desenvolver e validar um modelo de *matching* baseado em técnicas de PLN, com vistas à automatização da correspondência entre usuários e *experts* no contexto de assessoria científica. A proposta foi estruturada a partir da integração de abordagens contemporâneas com base na literatura, empregando *embeddings* semânticos e métricas de similaridade para identificar, ranquear *experts* com base em demandas reais.

Ao longo da pesquisa, quatro modelos de geração de *embeddings* foram implementados e avaliados: BERTimbau-base, BERTimbau-large, OpenAI-small e OpenAI-large. Cada modelo foi submetido a experimentação rigorosa, com validação empírica conduzida por meio de métricas consolidadas na literatura, como *Precision*@3, MRR@3, nDCG@3 e HR@3, além de medidas de eficiência computacional, como tempo médio de execução. Para garantir a confiabilidade dos resultados, o processo avaliativo contou com uma referência de *ground truth*, produzida manualmente pela equipe técnica da plataforma, assegurando comparação direta e objetiva entre as recomendações do modelo e as associações consideradas ideais por especialistas humanos.

Adicionalmente, foi conduzida uma análise estatística formal para verificar a significância das diferenças de desempenho entre os modelos. Testes de normalidade e homogeneidade de variância foram aplicados inicialmente, seguidos pela utilização de *ANOVA One-Way Repeated Measures* para as métricas que atenderam aos pressupostos paramétricos e do teste de *Friedman* para aquelas que violaram esses pressupostos. Nos casos de diferenças significativas, realizou-se análise pós-hoc com os métodos *Holm-Šidák* ou *Tukey*, conforme apropriado. Essa abordagem estatística possibilitou uma interpretação robusta dos resultados, reforçando a confiabilidade das conclusões.

O modelo OpenAI-small, apesar de sua arquitetura mais compacta, obteve resultados competitivos, com desempenho satisfatório em todas as métricas avaliadas e tempo de execução significativamente inferior. Essa combinação de eficiência e robustez o torna uma alternativa viável para aplicações com restrições de recursos computacionais.

Por outro lado, os modelos baseados no BERTimbau, tanto na configuração BERTimbau-base quanto na versão BERTimbau-large, apresentaram desempenho inferior em comparação aos modelos da OpenAI em quase todas as métricas. Apesar disso, mantiveram resultados consistentes e previsíveis. Contudo, o tempo de processamento foi consideravelmente maior especialmente no modelo BERTimbau-large devido ao maior número de parâmetros e à complexidade computacional envolvida. Em síntese, os achados deste estudo indicam que modelos modernos de linguagem disponibilizados via API, como os da OpenAI, oferecem vantagens significativas em termos de precisão, estabilidade e eficiência, sendo mais indicados para aplicações práticas de *matching* baseadas em textos semiestruturados.

Para a continuidade desta pesquisa, algumas direções podem ser exploradas com o objetivo de aprimorar a eficácia, a escalabilidade e a eficiência computacional do processo de *matching*. Uma das principais possibilidades é a implementação de uma arquitetura baseada em *Retrieval-Augmented Generation* (RAG), que permita a separação entre a etapa de indexação e a etapa de consulta. Com essa abordagem, os *embeddings* gerados a partir dos documentos de pesquisadores e *experts* poderiam ser armazenados previamente em um banco de dados vetorial, eliminando a necessidade de reprocessamento completo a cada nova consulta. Podem ser implementados os critérios de histórico, estilo de orientação e disponibilidade por meio de fontes alternativas de dados (ex.: formulários de *feedback*, preferências informadas por *experts*, logs de atendimento) e integrar esses atributos ao processo de *matching*.

Outra possibilidade para trabalhos futuros é a aplicação do modelo de *matching* desenvolvido em outros domínios de pesquisa e setores produtivos, como as áreas da saúde, jurídica ou comercial. Essa adaptação exigiria a coleta de novos conjuntos de dados específicos de cada domínio, além de eventuais ajustes nos atributos utilizados para caracterização dos perfis e nas estratégias de ponderação das dimensões de similaridade. Ao aplicar o modelo em contextos distintos, seria possível avaliar sua capacidade de generalização e identificar quais adaptações arquiteturais ou de pré-processamento seriam necessárias para atender às particularidades linguísticas e semânticas de cada área de aplicação.

Considera-se relevante a ampliação dos experimentos conduzidos neste estudo. Embora os testes realizados tenham demonstrado a viabilidade dos modelos propostos na tarefa de *matching* entre pesquisadores e *experts*, a utilização de um conjunto maior de documentos e de um número ampliado de *experts* pode fortalecer, de forma mais abrangente, a avaliação do desempenho das abordagens implementadas.

Outra possibilidade de expansão consiste na realização do processo de *matching* em duas etapas, em um fluxo semelhante ao adotado na prática pela SciBees. Inicialmente, seria executada uma pré-seleção de *experts* com maior alinhamento à área de atuação do pesquisador. Em seguida, a recomendação final poderia considerar critérios adicionais, tais como experiência metodológica, habilidades técnicas e características específicas da demanda. Essa estratégia tende a refletir de maneira mais precisa o processo operacional, além de potencialmente elevar as

taxas de recomendação correta.

Referências

AGRAWAL, A.; SHUKLA, P. Context aware automatic subjective and objective question generation using fast text to text transfer learning. *International Journal of Advanced Computer Science and Applications*, Science and Information (SAI) Organization Limited, v. 14, n. 4, 2023. DOI: <https://doi.org/10.14569/IJACSA.2023.0140451>. Citado 5 vezes nas páginas 60, 68, 70, 71 e 74.

AIZAWA, A. An information-theoretic perspective of tf–idf measures. *Information Processing & Management*, Elsevier, v. 39, n. 1, p. 45–65, 2003. DOI: <https://doi.org/10.1016/S0306-4573(02)00021-3>. Citado na página 39.

AKKASI, A.; MOENS, M.-F. Causal relationship extraction from biomedical text using deep neural models: A comprehensive survey. *Journal of biomedical informatics*, Elsevier, v. 119, p. 103820, 2021. DOI: <https://doi.org/10.1016/j.jbi.2021.103820>. Citado 4 vezes nas páginas 57, 68, 71 e 74.

AL-FARUK, M.; HUSSAIN, K.; SHAHRIAR, M. *A new string matching algorithm for analyzing university curriculum with respect to current job circular*. Tese (Doutorado) — East West University, 2018. Citado 2 vezes nas páginas 56 e 68.

ALMEIDA, K. F. R. d. et al. Efeito de cátions metálicos nas propriedades de filmes poliméricos a base de sericina e álcool polivinílico. Universidade Estadual do Oeste do Paraná, 2024. Citado 2 vezes nas páginas 78 e 79.

ALRUQI, T. N.; ALZAHRANI, S. M. Evaluation of an arabic chatbot based on extractive question-answering transfer learning and language transformers. *AI*, MDPI, v. 4, n. 3, p. 667–691, 2023. DOI: <https://doi.org/10.3390/ai4030035>. Citado 4 vezes nas páginas 60, 68, 69 e 71.

AMALIA, A. et al. Olcbot: Dissemination of interactive information related to indonesia's omnibus law with the implementation of fuzzy string matching algorithm and sastrawi stemmer. In: IEEE. *2022 6th International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM)*. [S.l.], 2022. p. 178–181. DOI: <https://doi.org/10.1109/ELTICOM57747.2022.10037966>. Citado 3 vezes nas páginas 21, 59 e 68.

ARANHA, C.; PASSOS, E. A tecnologia de mineração de textos. *Revista Eletrônica de Sistemas de Informação*, v. 5, n. 2, 2006. DOI: <https://doi.org/10.21529/RESI.2006.0502001>. Citado 2 vezes nas páginas 33 e 34.

ASKARI, A. et al. Injecting the score of the first-stage retriever as text improves bert-based re-rankers. *Discover Computing*, Springer, v. 27, n. 1, p. 15, 2024. DOI: <https://doi.org/10.1007/s10791-024-09435-8>. Citado 9 vezes nas páginas 62, 68, 69, 70, 71, 72, 73, 74 e 75.

ATHEY, S.; IMBENS, G. W. Machine learning methods that economists should know about. *Annual Review of Economics*, Annual Reviews, v. 11, n. 1, p. 685–725, 2019. DOI: <https://doi.org/10.1146/annurev-economics-080217-053433>. Citado na página 25.

AYODELE, T. O. Types of machine learning algorithms. *New advances in machine learning*, v. 3, n. 19-48, p. 5–1, 2010. DOI: <https://doi.org/10.5772/9385>. Citado na página 25.

BALAKRISHNAN, V.; LLOYD-YEMOH, E. Stemming and lemmatization: A comparison of retrieval performances. 2014. DOI: <https://doi.org/10.7763/LNSE.2014.V2.134>. Citado na página 37.

BARION, E. C. N.; LAGO, D. Mineração de textos. *Revista de ciências exatas e tecnologia*, v. 3, n. 3, p. 123–140, 2008. DOI: <https://doi.org/10.17921/1890-1793.2008v3n3p123-140>. Citado na página 38.

BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, IEEE, v. 5, n. 2, p. 157–166, 1994. DOI: <https://doi.org/10.1109/72.279181>. Citado na página 29.

BERNABÉ-MORENO, J. et al. An automatic skills standardization method based on subject expert knowledge extraction and semantic matching. *Procedia Computer Science*, Elsevier, v. 162, p. 857–864, 2019. DOI: <https://doi.org/10.1016/j.procs.2019.12.060>. Citado 5 vezes nas páginas 56, 68, 69, 70 e 74.

BOJANOWSKI, P. et al. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 5, p. 135–146, 2017. DOI: <https://doi.org/10.1162/tacl_a_00051>. Citado na página 41.

BORDES, A. et al. A semantic matching energy function for learning with multi-relational data: Application to word-sense disambiguation. *Machine learning*, Springer, v. 94, p. 233–259, 2014. DOI: <https://doi.org/10.1007/s10994-013-5363-6>. Citado na página 21.

BRAGA, A. d. P.; LUDERMIR, T. B.; CARVALHO, A. C. P. d. L. F. d. *Redes neurais artificiais: teoria e aplicações*. [S.l.]: LTC, 2000. Citado na página 26.

BRISSET, S. et al. Sftm: Fast matching of web pages using similarity-based flexible tree matching. *Information Systems*, Elsevier, v. 112, p. 102126, 2023. DOI: <https://doi.org/10.1016/j.is.2022.102126>. Citado 3 vezes nas páginas 60, 68 e 69.

BUCUR, A.-M.; COSMA, A.; DINU, L. P. Sequence-to-sequence lexical normalization with multilingual transformers. In: XU, W. et al. (Ed.). *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*. Online: Association for Computational Linguistics, 2021. p. 473–482. DOI: <https://doi.org/10.18653/v1/2021.wnut-1.53>. Citado na página 35.

CASELI, H. M.; NUNES, M. G. V. (Ed.). *Processamento de Linguagem Natural: Conceitos, Técnicas e Aplicações em Português*. [S.l.]: BPLN, 2023. <https://brasileiraspln.com/livro-pln>. ISBN 978-65-00-80693-9. Citado na página 35.

CHAKRAVARTI, R. et al. Cfo: A framework for building production nlp systems. In: . [S.l.: s.n.], 2019. p. 31–36. DOI:<https://doi.org/10.18653/v1/D19-3006>. Citado 2 vezes nas páginas 57 e 68.

CHEN, Y. et al. Automatic icd-10 coding: Deep semantic matching based on analogical reasoning. *Heliyon*, Elsevier, v. 9, n. 4, 2023. DOI: <https://doi.org/10.1016/j.heliyon.2023.e15570>. Citado 9 vezes nas páginas 20, 61, 68, 69, 70, 71, 72, 74 e 75.

CHO, K. et al. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: MOSCHITTI, A.; PANG, B.; DAELEMANS, W. (Ed.). *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Doha, Qatar: Association for Computational Linguistics, 2014. p. 1724–1734. DOI: <https://doi.org/10.3115/v1/D14-1179>. Citado na página 30.

CHUNG, J. et al. Empirical evaluation of gated recurrent neural networks on sequence modeling. abs/1412.3555, 2014. Citado na página 31.

CREMONESI, P.; KOREN, Y.; TURRIN, R. Performance of recommender algorithms on top-n recommendation tasks. In: *Proceedings of the fourth ACM conference on Recommender systems*. [S.l.: s.n.], 2010. p. 39–46. DOI: <https://doi.org/10.1145/1864708.1864721>. Citado na página 52.

CUI, X. et al. Bilstm-attention-crf model for entity extraction in internet recruitment data. *Procedia Computer Science*, Elsevier, v. 183, p. 706–712, 2021. DOI: <https://doi.org/10.1016/j.procs.2021.02.118>. Citado 3 vezes nas páginas 58, 68 e 70.

DEVLIN, J. et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In: BURSTEIN, J.; DORAN, C.; SOLORIO, T. (Ed.). *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. p. 4171–4186. DOI: <https://doi.org/10.18653/V1/N19-1423>. Citado 6 vezes nas páginas 20, 42, 43, 44, 69 e 71.

DEY, R.; SALEM, F. M. Gate-variants of gated recurrent unit (gru) neural networks. In: *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*. [S.l.: s.n.], 2017. p. 1597–1600. DOI: <https://doi.org/10.1109/MWSCAS.2017.8053243>. Citado na página 31.

DONG, J. Chinese short text matching model based on wobert word embedding representation and priori knowledge. In: IEEE. *2023 3rd International Conference on Consumer Electronics and Computer Engineering (ICCECE)*. [S.l.], 2023. p. 298–302. DOI: <https://doi.org/10.1109/ICCECE58074.2023.10135350>. Citado 2 vezes nas páginas 61 e 68.

DUAN, H.; WENG, Y.; GAO, X. Multi-task semantic matching model for small noisy data set. In: IEEE. *2021 16th International Conference on Computer Science & Education (ICCSE)*. [S.l.], 2021. p. 1114–1119. DOI: <https://doi.org/10.1109/ICCSE51940.2021.9569706>. Citado 5 vezes nas páginas 58, 68, 69, 70 e 71.

DUAN, Z. et al. Reviewer assignment based on sentence pair modeling. *Neurocomputing*, Elsevier, v. 366, p. 97–108, 2019. DOI: <https://doi.org/10.1016/j.neucom.2019.06.074>. Citado 8 vezes nas páginas 57, 68, 69, 70, 71, 72, 74 e 75.

ELGAMMAL, Z. et al. Matching applicants with positions for better allocation of employees in the job market. In: IEEE. *2021 22nd International Arab Conference on Information Technology (ACIT)*. [S.l.], 2021. p. 1–5. DOI: <https://doi.org/10.1109/ACIT53391.2021.9677374>. Citado 3 vezes nas páginas 21, 58 e 68.

FERREIRA, R.; SEMEDO, D.; MAGALHãES, J. *BERT Embeddings Can Track Context in Conversational Search*. 2021. Citado 4 vezes nas páginas 58, 68, 69 e 71.

FINGER, M. Inteligência artificial e os rumos do processamento do português brasileiro. *Estudos Avançados*, Instituto de Estudos Avançados da Universidade de São Paulo, v. 35, n. 101, p. 51–72, Jan 2021. ISSN 0103-4014. DOI: <https://doi.org/10.1590/s0103-4014.2021.35101.005>. Citado na página 42.

FUJISHIRO, N.; OTAKI, Y.; KAWACHI, S. Accuracy of the sentence-bert semantic search system for a japanese database of closed medical malpractice claims. *Applied Sciences*, MDPI, v. 13, n. 6, p. 4051, 2023. DOI: <https://doi.org/10.3390/app13064051>. Citado 8 vezes nas páginas 61, 68, 69, 70, 71, 72, 74 e 75.

GARCÍA-DÍAZ, J. A.; VALENCIA-GARCÍA, R. Compilation and evaluation of the spanish saticorpus 2021 for satire identification using linguistic features and transformers. *Complex & Intelligent Systems*, Springer, v. 8, n. 2, p. 1723–1736, 2022. DOI: <https://doi.org/10.1007/s40747-021-00625-1>. Citado 2 vezes nas páginas 59 e 68.

GARDAZI, N. et al. Bert applications in natural language processing: a review. *Artificial Intelligence Review*, v. 58, 03 2025. DOI: <https://doi.org/10.1007/s10462-025-11162-5>. Citado na página 40.

GOMES, D. T. Redes neurais recorrentes para previsão de séries temporais de memórias curta e longa. *Master's thesis, Department of Statistics, Campinas State University, Campinas, Brazil*, p. 153, 2005. DOI: <https://doi.org/10.47749/T/UNICAMP.2005.360549>. Citado na página 27.

HE, J. et al. Chemu 2020: Natural language processing methods are effective for information extraction from chemical patents. *Frontiers in Research Metrics and Analytics*, v. 6, 2021. ISSN 2504-0537. DOI: <https://doi.org/10.3389/frma.2021.654438>. Citado na página 36.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. DOI: <https://doi.org/10.1162/neco.1997.9.8.1735>. Citado 2 vezes nas páginas 29 e 30.

HOSSAIN, M. I. et al. Rematch: Research expert matching system. In: IEEE. *2018 International Symposium on Big Data Visual and Immersive Analytics (BDVA)*. [S.l.], 2018. p. 1–10. DOI: <https://doi.org/10.1109/BDVA.2018.8534021>. Citado 3 vezes nas páginas 56, 68 e 74.

HU, X.; ZHANG, H.; SUN, Y. Chinese medical short text matching model based on fine-tuning bert-attention-bilstm. In: IEEE. *2023 IEEE/ACIS 23rd International Conference on Computer and Information Science (ICIS)*. [S.l.], 2023. p. 91–96. DOI: <https://doi.org/10.1109/ICIS57766.2023.10210224>. Citado 8 vezes nas páginas 30, 61, 68, 69, 70, 71, 72 e 74.

HU, Y. et al. *Synthesizing Political Zero-Shot Relation Classification via Codebook Knowledge, NLI, and ChatGPT*. 2023. DOI: <https://doi.org/10.48550/arXiv.2308.07876>. Citado 3 vezes nas páginas 61, 68 e 70.

HUANG, Z.; ZHAO, W. A semantic matching approach addressing multidimensional representations for web service discovery. *Expert Systems with Applications*, Elsevier, v. 210, p. 118468, 2022. DOI: <https://doi.org/10.1016/j.eswa.2022.118468>. Citado 5 vezes nas páginas 59, 68, 69, 71 e 73.

IBRAHIMI, S. et al. Interactive exploration of journalistic video footage through multimodal semantic matching. In: *Proceedings of the 27th ACM International Conference on Multimedia*. [S.l.: s.n.], 2019. p. 2196–2198. DOI: <https://doi.org/10.1145/3343031.3350597>. Citado 3 vezes nas páginas 57, 68 e 69.

IYER, R. et al. *An End-to-End ML System for Personalized Conversational Voice Models in Walmart E-Commerce*. 2020. Citado 3 vezes nas páginas 57, 68 e 71.

JAIN, S.; MIAO, Y.; KAN, M.-Y. Comparative snippet generation. In: MALMASI, S. et al. (Ed.). *Proceedings of the Fifth Workshop on e-Commerce and NLP (ECNLP 5)*. Dublin, Ireland: Association for Computational Linguistics, 2022. p. 49–57. DOI: <https://doi.org/10.18653/v1/2022.ecnlp-1.7>. Citado 5 vezes nas páginas 59, 68, 69, 71 e 72.

JÄRVELIN, K.; KEKÄLÄINEN, J. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, ACM New York, NY, USA, v. 20, n. 4, p. 422–446, 2002. DOI: <https://doi.org/10.1145/582415.582418>. Citado na página 51.

JAYASUDHA, V.; DEEPA, N.; DEVI, T. Effective model for improving symptoms-based disease prediction by bimm-bert algorithm. In: IEEE. *2023 International Conference on Computer Communication and Informatics (ICCCI)*. [S.l.], 2023. p. 1–7. DOI: <https://doi.org/10.1109/ICCCI56745.2023.10128318>. Citado 6 vezes nas páginas 61, 68, 71, 72, 73 e 74.

JING, W. et al. Modeling and searching technology of distribution network data resources based on knowledge mapping. In: IEEE. *2020 7th International Conference on Information Science and Control Engineering (ICISCE)*. [S.l.], 2020. p. 837–845. DOI: <https://doi.org/10.1109/ICISCE50968.2020.00175>. Citado 2 vezes nas páginas 57 e 68.

KARYPIS, G. Evaluation of item-based top-n recommendation algorithms. In: *Proceedings of the tenth international conference on Information and knowledge management*. [S.l.: s.n.], 2001. p. 247–254. DOI: <https://doi.org/10.1145/502585.502627>. Citado na página 52.

KAUR, J.; BUTTAR, P. K. A systematic review on stopword removal algorithms. *International Journal on Future Revolution in Computer Science & Communication Engineering*, v. 4, n. 4, p. 207–210, 2018. Citado na página 38.

KHADILKAR, K.; KULKARNI, S.; BONE, P. Plagiarism detection using semantic knowledge graphs. In: IEEE. *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. [S.l.], 2018. p. 1–6. DOI: <https://doi.org/10.1109/ICCUBEA.2018.8697404>. Citado 4 vezes nas páginas 20, 56, 68 e 70.

KHAN, Z. et al. Densebert4ret: deep bi-modal for image retrieval. *Information Sciences*, Elsevier, v. 612, p. 1171–1186, 2022. DOI: <https://doi.org/10.1016/j.ins.2022.08.119>. Citado 2 vezes nas páginas 59 e 68.

KIM, J. et al. Nlp-fast: a fast, scalable, and flexible system to accelerate large-scale heterogeneous nlp models. In: IEEE. *2021 30th International Conference on Parallel Architectures and Compilation Techniques (PACT)*. [S.l.], 2021. p. 75–89. DOI: <https://doi.org/10.1109/PACT52795.2021.00013>. Citado na página 21.

KOTSIANTIS, S. B. et al. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, Amsterdam, v. 160, n. 1, p. 3–24, 2007. Citado na página 25.

KOVÁCS, Z. L. *Redes neurais artificiais*. [S.l.]: Editora Livraria da Fisica, 2006. Citado na página 26.

KRATZERT, F. et al. Rainfall–runoff modelling using long short-term memory (lstm) networks. *Hydrology and Earth System Sciences*, Copernicus Publications Göttingen, Germany, v. 22, n. 11, p. 6005–6022, 2018. DOI: <https://doi.org/10.5194/hess-22-6005-2018>. Citado na página 30.

KUMAR, A.; PATI, P. B. Offline hwr accuracy enhancement with image enhancement and deep learning techniques. *Procedia Comput. Sci.*, Elsevier Science Publishers B. V., NLD, v. 218, n. C, p. 35–44, jan. 2023. ISSN 1877-0509. Disponível em: <https://doi.org/10.1016/j.procs.2022.12.399>. Citado 2 vezes nas páginas 61 e 68.

LE, X.-H. et al. Application of long short-term memory (lstm) neural network for flood forecasting. *Water*, MDPI, v. 11, n. 7, p. 1387, 2019. DOI: <https://doi.org/10.3390/w11071387>. Citado na página 28.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group UK London, v. 521, n. 7553, p. 436–444, 2015. DOI: <https://doi.org/10.1038/nature14539>. Citado na página 25.

LEI, T.; JI, L.; LIU, S. Investigation of cross-social network user identification. In: IEEE. *2021 International Conference on Advanced Computing and Endogenous Security*. [S.l.], 2022. p. 1–7. DOI: <https://doi.org/10.1109/IEEECONF52377.2022.10013328>. Citado 2 vezes nas páginas 59 e 68.

LI, C.; JIANG, X.; ZHANG, K. A transformer-based deep learning approach for fairly predicting post-liver transplant risk factors. *Journal of Biomedical Informatics*, v. 149, p. 104545, 2024. ISSN 1532-0464. DOI: <https://doi.org/10.1016/j.jbi.2023.104545>. Citado 2 vezes nas páginas 62 e 68.

LI, Q.; HE, S. Similarity matching of medical question based on siamese network. *BMC Medical Informatics and Decision Making*, Springer, v. 23, n. 1, p. 55, 2023. DOI: <https://doi.org/10.1186/s12911-023-02161-z>. Citado 8 vezes nas páginas 61, 68, 69, 70, 71, 72, 73 e 74.

LIU, P. et al. QuadrupletBERT: An efficient model for embedding-based large-scale retrieval. In: TOUTANOVA, K. et al. (Ed.). *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, 2021. p. 3734–3739. DOI: <https://doi.org/10.18653/v1/2021.naacl-main.292>. Citado 4 vezes nas páginas 58, 68, 69 e 71.

LIU, Z. et al. *A Robustly Optimized BERT Pre-training Approach with Post-training*. Berlin, Heidelberg: Springer-Verlag, 2021. 471–484 p. DOI: <https://doi.org/10.1007/978-3-030-84186-7_31>. Citado na página 44.

LU, J. et al. Sampling wisely: Deep image embedding by top-k precision optimization. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2019. p. 7960–7969. DOI: <https://doi.org/10.1109/ICCV.2019.00805>. Citado na página 49.

MAHESH, B. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, v. 9, n. 1, p. 381–386, 2020. DOI: <https://doi.org/10.21275/ART20203995>. Citado na página 24.

MARTENOT, V. et al. Lisa: an assisted literature search pipeline for detecting serious adverse drug events with deep learning. *BMC medical informatics and decision making*, Springer, v. 22, n. 1, p. 338, 2022. DOI: <https://doi.org/10.1186/s12911-022-02085-0>. Citado 4 vezes nas páginas 59, 68, 69 e 71.

MEENAKSHI, D.; SHANAVAS, A. R. M. Transformer induced enhanced feature engineering for contextual similarity detection in text. *Bulletin of Electrical Engineering and Informatics*, v. 11, n. 4, p. 2124–2130, 2022. DOI: <https://doi.org/10.11591/eei.v11i4.3284>. Citado 5 vezes nas páginas 59, 68, 71, 72 e 74.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. *Proceedings of Workshop at ICLR*, v. 2013, 01 2013. Citado 2 vezes nas páginas 40 e 41.

MILOŠEVIĆ, N.; THIELEMANN, W. Comparison of biomedical relationship extraction methods and models for knowledge graph creation. *Journal of Web Semantics*, Elsevier, v. 75, p. 100756, 2023. DOI: <https://doi.org/10.1016/j.websem.2022.100756>. Citado 7 vezes nas páginas 61, 68, 69, 70, 71, 72 e 75.

MORAVVEJ, S. V. et al. An improved de algorithm to optimise the learning process of a bert-based plagiarism detection model. In: *2022 IEEE Congress on Evolutionary Computation (CEC)*. [S.l.]: IEEE Press, 2022. p. 1–7. DOI: <https://doi.org/10.1109/CEC55065.2022.9870280>. Citado 9 vezes nas páginas 31, 62, 68, 69, 70, 71, 72, 73 e 75.

MRIDHA, M. F. et al. L-boost: Identifying offensive texts from social media post in bengali. *Ieee Access*, IEEE, v. 9, p. 164681–164699, 2021. DOI: <https://doi.org/10.1109/ACCESS.2021.3134154>. Citado 2 vezes nas páginas 58 e 68.

NAIR, S. et al. Transfer learning approaches for building cross-language dense retrieval models. In: HAGEN, M. et al. (Ed.). *Advances in Information Retrieval*. Cham: Springer International Publishing, 2022. p. 382–396. ISBN 978-3-030-99736-6. DOI: <https://doi.org/10.1007/978-3-030-99736-6_26>. Citado 9 vezes nas páginas 60, 68, 69, 70, 71, 72, 73, 74 e 75.

PASCANU, R.; MIKOLOV, T.; BENGIO, Y. On the difficulty of training recurrent neural networks. In: *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*. [S.l.]: JMLR.org, 2013. (ICML'13), p. III–1310–III–1318. DOI: <https://dl.acm.org/doi/10.5555/3042817.3043083>. Citado na página 29.

PATIL, A.; JADON, A. Auto-labelling of bug report using natural language processing. In: IEEE. *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)*. [S.l.], 2023. p. 1–7. DOI: <https://doi.org/10.1109/I2CT57861.2023.10126470>. Citado 8 vezes nas páginas 20, 62, 68, 69, 71, 72, 73 e 74.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. [S.l.: s.n.], 2014. p. 1532–1543. DOI: <https://doi.org/10.3115/v1/D14-1162>. Citado na página 42.

PERREAULT-JENKINS, M. *A study of similarity measures for natural language processing as applied to candidate-project matching*. [S.l.]: McGill University (Canada), 2020. Citado 5 vezes nas páginas 39, 53, 57, 68 e 69.

PORTER, M. F. An algorithm for suffix stripping. *Program*, MCB UP Ltd, v. 14, n. 3, p. 130–137, 1980. DOI: <https://doi.org/10.1108/00330330610681286>. Citado 2 vezes nas páginas 36 e 37.

RADFORD, A. et al. Improving language understanding by generative pre-training. San Francisco, CA, USA, 2018. Citado na página 48.

RAFFEL, C. et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, v. 21, n. 140, p. 1–67, 2020. DOI: <https://doi.org/10.48550/arXiv.1910.10683>. Citado 2 vezes nas páginas 46 e 47.

RAUBER, T. W. Redes neurais artificiais. *Universidade Federal do Espírito Santo*, v. 29, 2005. Citado na página 26.

REN, J. et al. Matching algorithms: Fundamentals, applications and challenges. *IEEE Transactions on Emerging Topics in Computational Intelligence*, IEEE, v. 5, n. 3, p. 332–350, 2021. DOI: <https://doi.org/10.1109/TETCI.2021.3067655>. Citado 2 vezes nas páginas 20 e 21.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958. DOI: <https://doi.org/10.1037/h0042519>. Citado na página 27.

SADRI, N. *Evaluating Dense Passage Retrieval using Transformers*. 2022. Citado 4 vezes nas páginas 60, 68, 73 e 74.

SAIF, H. et al. On stopwords, filtering and data sparsity for sentiment analysis of Twitter. In: CALZOLARI, N. et al. (Ed.). *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA), 2014. p. 810–817. Disponível em: <https://aclanthology.org/L14-1265/>. Citado na página 38.

SANH, V. et al. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. 2019. Citado 2 vezes nas páginas 44 e 45.

SANTOS, V. dos; LIFSCHITZ, S. A semantic search approach for hyper relational knowledge graphs. In: SBC. *Anais Estendidos do XXXVI Simpósio Brasileiro de Bancos de Dados*. [S.l.], 2021. p. 106–112. DOI: <https://doi.org/10.5753/sbbd_estendido.2021.18171>. Citado 2 vezes nas páginas 58 e 68.

SARICA, S.; LUO, J. Stopwords in technical language processing. *Plos one*, Public Library of Science San Francisco, CA USA, v. 16, n. 8, p. e0254937, 2021. DOI: <https://doi.org/10.1371/journal.pone.0254937>. Citado na página 38.

SHAN, X. et al. Glow: global weighted self-attention network for web search. In: IEEE. *2021 IEEE International Conference on Big Data (Big Data)*. [S.l.], 2021. p. 519–528. DOI: <https://doi.org/10.1109/BigData52589.2021.9671546>. Citado 8 vezes nas páginas 58, 68, 69, 70, 71, 72, 74 e 75.

SHEN, G. et al. Deep learning with gated recurrent unit networks for financial sequence predictions. *Procedia computer science*, Elsevier, v. 131, p. 895–903, 2018. DOI: <https://doi.org/10.1016/j.procs.2018.04.298>. Citado na página 31.

SHIMADA, N.; YAMAZAKI, N.; TAKANO, Y. Multi-objective optimization models for many-to-one matching problems. *Journal of Information Processing*, Information Processing Society of Japan, v. 28, p. 406–412, 2020. DOI: <https://doi.org/10.2197/ipsjjip.28.406>. Citado na página 21.

SILVA, R. A. C. Inteligência artificial aplicada a ambientes de engenharia de software: Uma visão geral. *INFOCOMP Journal of Computer Science*, v. 4, n. 4, p. 27–37, 2005. Citado na página 24.

SOUTO, M. D. et al. Técnicas de aprendizado de máquina para problemas de biologia molecular. *Sociedade Brasileira de Computaçao*, v. 1, n. 2, 2003. Citado na página 24.

SOUZA, F.; NOGUEIRA, R.; LOTUFO, R. *Portuguese Named Entity Recognition using BERT-CRF*. 2019. Citado 2 vezes nas páginas 45 e 86.

SOUZA, L. F. S. d.; GONÇALVES, A. L.; SOUZA, J. A. d. Utilização prática de word embedding aplicada à classificação de texto. *Anais do Congresso Internacional de Conhecimento e Inovação – ciki*, v. 1, n. 1, nov. 2020. DOI: <https://doi.org/10.48090/ciki.v1i1.899>. Citado na página 40.

SRIDEVI, G.; SUGANTHI, S. K. Ai based suitability measurement and prediction between job description and job seeker profiles. *International Journal of Information Management Data Insights*, Elsevier, v. 2, n. 2, p. 100109, 2022. DOI: <https://doi.org/10.1016/j.jjimei.2022.100109>. Citado 3 vezes nas páginas 60, 68 e 74.

STANKOVIĆ, R. et al. Rule-based automatic multi-word term extraction and lemmatization. In: CALZOLARI, N. et al. (Ed.). *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), 2016. p. 507–514. Disponível em: <https://aclanthology.org/L16-1081/>. Citado na página 38.

STAUDEMEYER, R.; MORRIS, E. *Understanding LSTM a tutorial into Long Short-Term Memory Recurrent Neural Networks*. 2019. Citado na página 29.

SU, J. et al. Bert-hlstms: Bert and hierarchical lstms for visual storytelling. *Computer Speech Language*, v. 67, p. 101169, 2021. ISSN 0885-2308. DOI: <https://doi.org/10.1016/j.csl.2020.101169>. Citado 2 vezes nas páginas 58 e 68.

SU, W. et al. Pre-training for legal case retrieval based on inter-case distinctions. *ACM Trans. Inf. Syst.*, Association for Computing Machinery, New York, NY, USA, v. 43, n. 5, jul. 2025. ISSN 1046-8188. DOI: <https://doi.org/10.1145/3735127>. Citado 6 vezes nas páginas 62, 68, 69, 71, 74 e 75.

SZARKOWSKA, K. et al. Quality assessment of knowledge graph hierarchies using kg-bert. In: *DL4KG@ISWC*. [s.n.], 2021. Disponível em: <https://ceur-ws.org/Vol-3034/paper1.pdf>. Citado 4 vezes nas páginas 58, 68, 69 e 71.

TANBERK, S. et al. Resume matching framework via ranking and sorting using nlp and deep learning. In: IEEE. *2023 8th International Conference on Computer Science and Engineering (UBMK)*. [S.l.], 2023. p. 453–458. DOI: <https://doi.org/10.1109/UBMK59864.2023.10286605>. Citado 7 vezes nas páginas 62, 68, 69, 70, 71, 72 e 74.

TISSOT, H. C.; CAMARGO, L. C.; POZO, A. T. Treinamento de redes neurais feedforward: comparativo dos algoritmos backpropagation e differential evolution. In: SN. *Brazilian Conference on Intelligent Systems*. [S.l.], 2012. Citado 2 vezes nas páginas 27 e 28.

TRAN, R. K. T. Predicting mental conditions based on "history of present illness" in psychiatric notes with deep neural networks. *Biomedical*, n. 3, p. 138–148, 2017. DOI: <https://doi.org/10.1016/j.jbi.2017.06.010>. Citado na página 35.

TÜLÜMEN, N. et al. Hybrid job and resume matcher. In: IEEE. *2021 6th International Conference on Computer Science and Engineering (UBMK)*. [S.l.], 2021. p. 163–168. DOI: <https://doi.org/10.1109/UBMK52708.2021.9558932>. Citado 5 vezes nas páginas 59, 68, 69, 71 e 74.

UHLIG, F. et al. Combining ai and am – improving approximate matching through transformer networks. *Forensic Science International: Digital Investigation*, v. 45, p. 301570, 2023. ISSN 2666-2817. DOI: <https://doi.org/10.1016/j.fsidi.2023.301570>. Citado 2 vezes nas páginas 62 e 68.

VANETIK, N.; KOGAN, G. Job vacancy ranking with sentence embeddings, keywords, and named entities. *Information*, MDPI, v. 14, n. 8, p. 468, 2023. DOI: <https://doi.org/10.3390/info14080468>. Citado 5 vezes nas páginas 62, 68, 69, 73 e 74.

VASWANI, A. et al. Attention is all you need. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017. (NIPS'17), p. 6000–6010. ISBN 9781510860964. DOI: <https://dl.acm.org/doi/10.5555/3295222.3295349>. Citado 3 vezes nas páginas 20, 32 e 33.

VOORHEES, E. The trec-8 question answering track report. *Nat. Lang. Eng.*, v. 7, p. 361–378, 12 2001. DOI: <https://doi.org/10.1017/S1351324901002789>. Citado na página 50.

WADUD, M. A. H. et al. Deep-bert: Transfer learning for classifying multilingual offensive texts on social media. *Computer Systems Science & Engineering*, v. 44, n. 2, 2023. DOI: <https://doi.org/10.32604/csse.2023.027841>. Citado 8 vezes nas páginas 20, 62, 68, 69, 70, 71, 72 e 73.

WANG, P. et al. Intelligent radar hrrp target recognition based on cnn-bert model. *EURASIP Journal on Advances in Signal Processing*, Springer, v. 2022, n. 1, p. 89, 2022. DOI: <https://doi.org/10.1186/s13634-022-00909-9>. Citado 3 vezes nas páginas 60, 68 e 70.

WANG, T. et al. A joint framenet and element focusing sentence-bert method of sentence similarity computation. *Expert Systems with Applications*, Elsevier, v. 200, p. 117084, 2022. DOI: <https://doi.org/10.1016/j.eswa.2022.117084>. Citado 3 vezes nas páginas 31, 60 e 68.

WANG, Y. *Machine reading comprehension to answer COVID19 queries using Bio-Bert and multi-task learning*. Tese (Doutorado) — Laurentian University of Sudbury, 2022. Citado 5 vezes nas páginas 60, 68, 69, 71 e 74.

WANG, Z. et al. A lightweight iot intrusion detection model based on improved bert-of-theseus. *Expert Systems with Applications*, Elsevier, v. 238, p. 122045, 2024. DOI: <https://doi.org/10.1016/j.eswa.2023.122045>. Citado 6 vezes nas páginas 31, 63, 68, 70, 72 e 75.

WILLETT, P. The porter stemming algorithm: then and now. *Program*, Emerald Group Publishing Limited, v. 40, n. 3, p. 219–223, 2006. DOI: <https://doi.org/10.1108/00330330610681295>. Citado na página 37.

WU, K. et al. Named entity recognition of rice genes and phenotypes based on bigru neural networks. *Computational Biology and Chemistry*, Elsevier, v. 108, p. 107977, 2024. DOI: <https://doi.org/10.1016/j.compbiolchem.2023.107977>. Citado 4 vezes nas páginas 62, 68, 69 e 71.

XIONG, J. et al. Efficient reinforcement learning-based method for plagiarism detection boosted by a population-based algorithm for pretraining weights. *Expert Systems with Applications*, Elsevier, v. 238, p. 122088, 2024. DOI: <https://doi.org/10.1016/j.eswa.2023.122088>. Citado 3 vezes nas páginas 63, 68 e 70.

XU, R.; WUNSCH, D. Survey of clustering algorithms. *IEEE Transactions on neural networks*, Ieee, v. 16, n. 3, p. 645–678, 2005. DOI: <https://doi.org/10.1109/TNN.2005.845141>. Citado na página 25.

YANG, J. et al. Eeg-based emotion classification based on bidirectional long short-term memory network. *Procedia Computer Science*, Elsevier, v. 174, p. 491–504, 2020. DOI: <https://doi.org/10.1016/j.procs.2020.06.117>. Citado 3 vezes nas páginas 28, 57 e 68.

YANG, L. et al. anmm: Ranking short answer texts with attention-based neural matching model. In: *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. New York, NY, USA: Association for Computing Machinery, 2016. (CIKM '16), p. 287–296. ISBN 9781450340731. DOI: <https://doi.org/10.1145/2983323.2983818>. Citado na página 21.

YANG, Z. et al. Xlnet: generalized autoregressive pretraining for language understanding. In: ____. *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019. DOI: <https://dl.acm.org/doi/10.5555/3454287.3454804>. Citado na página 45.

YILMAZ, Z. A. *Cross-domain sentence modeling for relevance transfer with BERT*. Dissertação (Mestrado) — University of Waterloo, 2019. Disponível em: <http://hdl.handle.net/10012/15326>. Citado 5 vezes nas páginas 57, 68, 69, 71 e 73.

YUN-TAO, Z.; LING, G.; YONG-CHENG, W. An improved tf-idf approach for text classification. *Journal of Zhejiang University-Science A*, Springer, v. 6, n. 1, p. 49–55, 2005. DOI: <https://doi.org/10.1007/BF02842477>. Citado na página 39.

ZHANG, Q. et al. Unbert: User-news matching bert for news recommendation. In: *IJCAI*. [S.l.: s.n.], 2021. v. 21, p. 3356–3362. DOI: <https://doi.org/10.24963/ijcai.2021/462>. Citado 5 vezes nas páginas 59, 68, 69, 70 e 71.

ZHANG, S. et al. Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)*, ACM New York, NY, USA, v. 52, n. 1, p. 1–38, 2019. DOI: <https://doi.org/10.1145/3285029>. Citado na página 21.

ZHANG, Y. et al. Learning-based widget matching for migrating gui test cases. In: *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*. [S.l.: s.n.], 2024. p. 1–13. DOI: <https://doi.org/10.1145/3597503.3623322>. Citado 5 vezes nas páginas 63, 68, 70, 71 e 73.

ZHAO, X. Framework of bert-based and attention-based networks for community question answering tasks. 2021. Disponível em: <http://hdl.handle.net/10315/38666>. Citado 4 vezes nas páginas 59, 68, 69 e 71.