UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Programa de Pós-Graduação em Computação

Dissertação

Classificação Automática de Produções Científicas em Inteligência Artificial Utilizando Processamento de Linguagem Natural

Felipe Camargo Gruendemann

Pelotas, 2023

Felipe Camargo Gruendemann

Classificação Automática de Produções Científicas em Inteligência Artificial Utilizando Processamento de Linguagem Natural

Dissertação apresentada ao Programa de Pós-Graduação em Computação do Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Ricardo Matsumura de Araujo

Pelotas, 2023

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

G886c Gruendemann, Felipe Camargo

Classificação automática de produções científicas em inteligência artificial utilizando processamento de linguagem natural / Felipe Camargo Gruendemann ; Ricardo Matsumura de Araujo, orientador. — Pelotas, 2023.

58 f. : il.

Dissertação (Mestrado) — Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2023.

1. Inteligência artificial. 2. Classificação. 3. Produção científica. 4. Processamento de linguagem natural. I. Araujo, Ricardo Matsumura de, orient. II. Título.

CDD : 005

Elaborada por Simone Godinho Maisonave CRB: 10/1733

Felipe Camargo Gruendemann

Classificação Automática de Produções Científicas em Inteligência Artificial Utilizando Processamento de Linguagem Natural

Dissertação aprovada, como requisito parcial, para obtenção do grau de Mestre em Ciência da Computação, Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 23 de fevereiro de 2023

Banca Examinadora:

Prof. Dr. Ricardo Matsumura de Araujo (orientador)
Doutor em Ciência da Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Ulisses Brisolara Corrêa
Doutor em Computação pela Universidade Federal de Pelotas.

Profa. Dra. Larissa Astrogildo de Freitas
Doutora em Ciência da Computação pela Pontifícia Universidade Católica do Rio Grande do Sul.

Dedico este trabalho a todos que me apoiaram nesta etapa e que contribuíram para sua conclusão.

RESUMO

GRUENDEMANN, Felipe Camargo. **Classificação Automática de Produções Científicas em Inteligência Artificial Utilizando Processamento de Linguagem Natural**. Orientador: Ricardo Matsumura de Araujo. 2023. 58 f. Dissertação (Mestrado em Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2023.

A análise de comunidades científicas é um tema importante para compreensão de cenários científicos em diferentes perspectivas. Esse assunto ganha força com o desenvolvimento de bibliotecas digitais. Nesse contexto, existem diferentes repositórios virtuais que disponibilizam dados bibliográficos de produções científicas e informações sobre autores e veículos de publicação. No Brasil, a principal base de dados científicos é a Plataforma Lattes, que conta com milhões de currículos de pesquisadores. Contudo, a plataforma carece de formas automatizadas para análise de dados e enfrenta problemas relacionados ao preenchimento manual de texto livre. Dessa forma, a tarefa de analisar o volume de publicações por assunto pode ser uma tarefa difícil. Modelos de *machine learning* aplicados com técnicas de processamento de linguagem natural vêm se mostrando uma alternativa útil para classificação de texto. Assim, neste trabalho, foram desenvolvidos modelos para classificar subárea e especialidade de trabalhos, baseando-se apenas no título. Como caso de estudo, foi utilizada a subárea de Inteligência Artificial e suas especialidades. Assim, foram construídos conjuntos de dados extraídos da plataforma *The DBLP Computer Science Bibliography* (DBLP) para o desenvolvimento de dois modelos: um para classificar se um determinado título da computação está relacionado à subárea da IA; outro para classificar, dentre sete categorias, qual a especialidade da IA. Os modelos atingiram acurácia de 93% e 71%, respectivamente.

Palavras-chave: Classificação. Produção científica. Inteligência Artificial. Processamento de Linguagem Natural.

ABSTRACT

GRUENDEMANN, Felipe Camargo. **Titulo do Trabalho em Ingles**. Advisor: Ricardo Matsumura de Araujo. 2023. 58 f. Dissertation (Masters in Computer Science) – Technology Development Center, Federal University of Pelotas, Pelotas, 2023.

The analysis of scientific communities is an important topic for understanding scientific scenarios from different perspectives. This subject gains strength with the development of digital libraries. In this context, there are different virtual repositories that provide bibliographic data of scientific productions and information about authors and venues. In Brazil, the main scientific database is the Lattes Platform, which has millions of researchers' CVs. However, the platform lacks automated tools to analyze data and faces problems related to manual filling of free text fields. Thus, the task of analyzing the volume of publications by subject can be a difficult task. Machine learning models applied with natural language processing techniques are being shown to be a useful alternative for text classification. In this work, models were developed to classify subarea and specialty of cientific works, based only on the title. As a case study, the subarea of Artificial Intelligence and its specialties was used. In this way, datasets were created from extraction of The DBLP Computer Science Bibliography (DBLP) data for developing of two models: one to classify whether a given title of Computer Science is related to the subarea of AI; another to classify, among seven categories, which is the specialty of AI. The models achieved accuracy of 93% and 71%, respectively.

Keywords: Classification. Cientific production. Artificial intelligence. Natural language processing.

LISTA DE FIGURAS

Figura 1 Captura de tela da página "*Top publications*"na categoria "*Engineering & Computer Science*"com as subcategorias sendo exibidas. . . 23

Figura 2 Captura de tela da página "*Top publications*"na subcategoria "*Artificial Intelligence*"com os *top* veículos de publicação listados. 24

Figura 3 Títulos por categoria - Subáreas. 27

Figura 4 Títulos por categoria – Especialidades. 28

Figura 5 *Pipeline* BERT ilustrado. No topo, está representada a entrada de texto que é o título de cada produção. Dentro do retângulo "Construtor de exemplos", estão ilustrados componentes do método que transforma o texto em uma entrada válida para o encoder. As três saídas do construtor são as entradas do modelo Keras. O BERT é carregado como uma *layer* do modelo e sua saída é correspondente ao token [CLS]. O vetor de saída passa por uma camada de Dropout e, depois, serve como entrada para o modelo de classificação. O modelo Dense então calcula sua função de saída e esta é utilizada como classificação. Todos componentes dentro do retângulo "Modelo Keras"são treinados juntos. 32

Figura 6 Ilustração do Modelo Keras candidato à Modelo Subáreas. Na parte superior da imagem, está a camada de entrada do modelo, composta pelas três saídas do construtor de exemplos. Abaixo, a *keras_layer* é o modelo BERT incorporado. Em sequência, a camada de Dropout e, finalmente, o modelo de classificação. Nesta figura, está ilustrado apenas o modelo desenvolvido para Subáreas, visto que sua saída é de tamanho um. O modelo candidato à Modelo Especialidades desenvolvido, tem especificações muito parecidas, nessa ilustração, a única diference seria a *output* apresentada, no lugar de 1 seria 7. 33

Figura 7 Ilustração do processo que resultou no melhor Modelo Subáreas e Modelo Especialidades. A sequência em que os passos foram executados é da esquerda para direita. No primeiro retângulo à esquerda, está ilustrada a etapa descrita na validação inicial, onde as combinações dentro da chave azul são os *pipelines* da scikit-learn; já a chave amarela compreende o *pipeline* que foi desenvolvido inicialmente para o BERT; Cada seta que conecta um modelo (círculo) com um tipo de representação (quadrado) expressa uma combinação testada. As melhores combinações foram modelo LinearSVC com processamento TF e o modelo Dense treinado com o BERT. Estes modelos estão representados no segundo retângulo. Assim, sob o texto "Melhor modelo", está ilustrado o resultado da etapa de busca de hiper-parâmetros, onde o modelo BERT teve resultados melhores. Por fim, estão as etapas de teste e resultados finais do modelo. Sendo assim, para ambos modelos Subáreas e Especialidades, o *pipeline* BERT foi o que obteve melhores resultados. . . . 36
Figura 8 Acurácia no treino Dense + BERT – Subáreas: No gráfico da figura é possível observar que a acurácia mais alta na validação foi obtida na segunda época (linha pontilhada vermelha). 38
Figura 9 *Loss* no treino Dense + BERT – Subáreas: No gráfico da figura é possível observar que a *loss* mais baixa na validação foi obtida na segunda época (linha pontilhada vermelha). 39
Figura 10 Matriz de confusão Modelo Subáreas. 40
Figura 11 Acurácia no treino Dense + BERT – Especialidades: No gráfico da figura é possível observar que a acurácia mais alta na validação foi obtida na segunda época (linha pontilhada vermelha). 42
Figura 12 *Loss* no treino Dense + BERT – Especialidades: No gráfico da figura é possível observar que a *loss* mais baixa na validação foi obtida na segunda época (linha pontilhada vermelha). 42
Figura 13 Matriz de confusão Modelo Especialidades. 43
Figura 14 Matriz de confusão Modelo Subáreas - Sucupira. 46
Figura 15 Matriz de confusão Modelo Especialidades - Sucupira. 47

LISTA DE TABELAS

Tabela 1 Subáreas selecionadas. 23
Tabela 2 Veículos de publicação na subárea de Inteligência Artificial. 24
Tabela 3 Especialidades selecionadas. 25
Tabela 4 Volume de exemplos por conjunto de dados. 28

Tabela 5 Validação inicial: Relatório de classificação modelo SVC + TF – Subáreas. 37
Tabela 6 Validação inicial: Relatório de classificação modelo Dense + BERT – Subáreas 37
Tabela 7 Busca de hiper-parâmetros: Relatório de classificação modelo Dense + BERT – Subáreas. 38
Tabela 8 Teste: Relatório de classificação modelo Dense + BERT – Subáreas. 39
Tabela 9 Validação inicial: Relatório de classificação modelo SVC + TF – Especialidades. 41
Tabela 10 Validação Inicial: Relatório de classificação modelo Dense + BERT – Especialidades. 41
Tabela 11 Busca de hiper-parâmetros: Relatório de classificação modelo SVC + TF – Especialidades. 41
Tabela 12 Busca de hiper-parâmetros: Relatório de classificação modelo Dense + BERT – Especialidades. 42
Tabela 13 Teste: Relatório de classificação modelo Dense + BERT – Especialidades. 43
Tabela 14 Ocorrência de termos selecionados nos exemplos da classe *knowledge representation & reasoning* classificados como *machine learning*. 44

Tabela 15 Veículos de publicação selecionados para as produções fora da subárea da IA. 55
Tabela 16 Veículos de publicação selecionados para as especialidades da IA. 56

Tabela 17 Nome dos veículos de publicação e nome de busca na API DBLP. . 58

LISTA DE ABREVIATURAS E SIGLAS

BERT	*Bidirectional Encoder Representations from Transformers*
DBLP	*The DBLP Computer Science Bibliography*
IA	Inteligência Artificial
PLN	Processament de Lnguagem Natural
SVC	*Linear Support Vector Classification*
TF	*Term Frequency*
TF-IDF	*Term frequency–inverse document frequency*

SUMÁRIO

1 INTRODUÇÃO 14

2 TRABALHOS RELACIONADOS 17
2.1 Classificação automática de artigos 17
2.2 Desambiguação de autores 18
2.3 Estudos de comunidades científicas em bases digitais 19
2.4 Estudos com dados da plataforma Lattes 20

3 OBJETIVOS 21
3.0.1 Objetivos Específicos 21

4 METODOLOGIA 22
4.1 Coleta de dados 22
4.1.1 Conjunto de dados Subáreas 22
4.1.2 Conjunto de dados Especialidades 24
4.1.3 *Scripts* de extração desenvolvidos 25
4.2 Limpeza dos dados 26
4.3 Definição de conjuntos de treino e teste 27
4.4 Especificação da tarefa dos modelos 28
4.5 Representações utilizadas 29
4.6 Seleção e treinamento dos modelos 30
4.6.1 Validação inicial 30
4.6.2 Busca de hiper-parâmetros 33

5 RESULTADOS 35
5.1 Modelo de Subáreas 36
5.1.1 Validação inicial 37
5.1.2 Busca de hiper-parâmetros 37
5.1.3 Testes 38
5.2 Modelo de Especialidades 40
5.2.1 Validação inicial 40
5.2.2 Busca de hiper-parâmetros 40
5.2.3 Teste 41
5.3 Teste com dados da plataforma Sucupira 45

6 CONCLUSÃO 49

REFERÊNCIAS 50

APÊNDICE A VEÍCULOS DE PUBLICAÇÃO 54
APÊNDICE B MAPA DE BUSCAS NA API DO DBLP 57

1 INTRODUÇÃO

As comunidades científicas vêm sendo estudadas de diferentes pontos de vista e, desde os anos de 1990, são assunto em análise de redes sociais, desenvolvimento de tópicos e distribuição de comunidades científicas (BIRYUKOV; DONG, 2010). Nessas perspectivas, as bibliotecas científicas digitais são um tipo de recurso que pode ser utilizado como artifício para desenvolvimento de trabalhos nesses temas.

Nesse contexto, as bibliotecas científicas digitais têm sido uma importante ferramenta para a pesquisa científica, proporcionando acesso fácil e rápido a uma grande quantidade de artigos e outras publicações científicas. Uma base frequentemente utilizada é a DBLP, que é uma biblioteca digital da computação com mais de 5 milhões de registros. Ela inclui informações sobre artigos publicados em conferências e revistas na área de computação, bem como informações sobre autores e coautores. A base DBLP é mantida pelo centro de pesquisa *Schloss Dagstuhl* e tem sido utilizada como uma fonte confiável de informações sobre a produção científica na área de computação.

No cenário brasileiro, a principal base de dados é plataforma Lattes. Esta é uma ferramenta digital que dispõe de informações acadêmicas e científicas. Desenvolvida pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), a plataforma reúne dados de currículos, contendo área de atuação dos pesquisadores, assim como suas produções científicas e veículo de publicação. A plataforma é amplamente utilizada por instituições de pesquisa e universidades para avaliação de currículos, busca de colaboradores e financiamento de projetos.

A Ciência da Computação é uma área fundamental na pesquisa científica no Brasil, tendo figurado como uma das áreas de atuação importantes na plataforma Lattes, pois foi uma das áreas com maior número de currículos associados (DIAS; MOITA; DIAS, 2016). Fato este que demonstra a relevância em estudar a área e buscar formas de entender como o conhecimento vem sendo produzido. No entanto, existem dificuldades associadas ao estudo da produção científica com dados vindos da plataforma, pois a inserção de informações na base é realizada em formato de texto livre, o que pode levar a problemas de fiabilidade e impossibilidade de análises automatizadas em

alguns casos.

Um problema existente é a classificação de produções científicas em assuntos específicos. No currículo Lattes, há a indicação das áreas de atuação do pesquisador, entretanto os assuntos das produções científicas não são indicadas com precisão. Em artigos publicados em revistas ou congressos, pode haver a informação do veículo de publicação, dado que pode contribuir para descoberta da área/assunto do artigo. Ainda assim, para uma análise automatizada, seria necessário ter uma relação de inúmeros veículos de publicação e a área associada. Algo que torna ainda mais impraticável esse tipo de solução é o fato de que podem existir revistas e congressos com nomes/siglas ambíguos ou com erros de digitação do usuário.

Em relação à classificação de produções científicas, a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) tem importante influência. A CAPES é uma instituição vinculada ao Ministério da Educação do Brasil responsável pelo fomento e estabelecimento da pós-graduação no país. Ela estabelece uma organização para as Áreas do Conhecimento, sendo definida a hierarquia: Grande Área, Área, Subárea e Especialidade, da mais abrangente à mais específica, respectivamente. A Ciência da Computação é uma área dentro da grande área de Ciências Exatas e da Terra, mas ainda não existe uma subárea da Ciência da Computação dedicada à Inteligência Artificial. Esse é um importante campo de estudo o qual ainda não possui uma forma de identificação enquanto subárea e que servirá como estudo de caso nesse trabalho. Nesse sentido, para o presente trabalho, o assunto Inteligência Artificial (IA) foi considerado como uma subárea da Ciência da Computação e os tópicos específicos desse assunto foram tratados como Especialidades dentro da IA.

Abordagens inseridas na subárea de IA como Aprendizado de Máquina e técnicas de Processamento de Linguagem Natural (PLN), já são práticas consolidadas para resolver problemas de classificação (KADHIM, 2019). Esse tipo de abordagem é especialmente interessante nesse contexto, pois alguns métodos de PLN têm estratégias para lidar com lacunas nos textos ou erros de digitação (DEVLIN et al., 2018; LI et al., 2020), características presentes no problema levantado.

Diante da relevância da Ciência da Computação para a ciência brasileira, da inexistência de uma subárea em Inteligência Artificial e da limitação dos dados para classificar as publicações, esse trabalho busca desenvolver uma metodologia para classificar a subárea e especialidades em IA de publicações baseando-se apenas nos títulos e usando os dados do DBLP. Uma vez que metodologias envolvendo IA e PLN têm se apresentado como solução para diversos problemas de classificação, serão desenvolvidos modelos baseados nessas metodologias que permitam a identificação da subárea de Inteligência Artificial e das especialidades da mesma.

Espera-se que este trabalho sirva como um exemplo de metodologia para desenvolvimento de modelos de classificação de assuntos de trabalhos científicos. Além

disso, é esperado que os dados coletados possam ser utilizados em outros estudos e/ou sirvam como exemplo de construção de conjuntos de dados para os pesquisadores. Também é desejado que a metodologia empregada seja de valor para a comunidade científica e que possa ser incorporada em outros estudos e soluções.

2 TRABALHOS RELACIONADOS

No contexto da pesquisa em bibliotecas acadêmicas digitais, existem diversos trabalhos que se propõem analisar, criar metologias ou aplicações com dados de bibliográficos. Durante o desenvolvimento desse trabalho foram feitas buscas nas principais bases acadêmicas com o objetivo de encontrar trabalhos relacionados.

Foram encontrados diversos estudos que possuem relação com esse trabalho: pesquisas no DBLP ou outras bibliotecas digitais, trabalhos de análise e metodologias para dados bibliográficos, entre outros. Contudo, não foram encontradas produções com aplicação de técnicas de Processamento de Linguagem Natural e Inteligência Artificial para classificação automática de artigos baseadas apenas nos seus títulos. Neste capítulo discorre-se sobre alguns avanços em pesquisas que se relacionam com a proposta atual, contextualizando o trabalho não só em Computação, mas em outras áreas com trabalhos afins.

2.1 Classificação automática de artigos

Das frentes apresentadas nesse capítulo, esta seção aborda a mais próxima do trabalho desenvolvido, porém a que menos se encontraram trabalhos relacionados. O que evidencia a importância do desenvolvimento da pesquisa nesse sentido.

Em Joorabchi; Mahdi (2011), foi proposta uma abordagem para classificação não supervisionada de literatura científica armazenada em bibliotecas e repositórios digitais. Esse trabalho propõe-se a criar uma metodologia para classificar produções científicas de acordo com um padrão de biblioteca. O método se baseia na identificação de todas as referências citadas no documento a ser classificado para inferir a classe mais provável para o próprio documento com a ajuda de um mecanismo de ponderação. A aplicação do método foi demonstrada com um *software* protótipo para classificação automática de acordo com o esquema de Classificação Decimal de Dewey.

A abordagem proposta, porém, depende de metadados de classificação de assuntos das publicações que citam o documento a ser classificado ou uma de suas refe-

rências. No trabalho foram usados metadados de referências já catalogadas em bibliotecas convencionais existentes, com objetivo de inferir probabilisticamente a classe mais adequada para o documento.

2.2 Desambiguação de autores

Um problema comum em bases com múltiplos trabalhos é o da desambiguação de autores, isto é, da identificação correta do autor baseado em seu nome. Publicações nas quais o nome de um autor apresenta diferentes grafias ou abreviações podem ser armazenadas separadamente e o conhecimento produzido por ele não lhe será atribuído. Por outro lado, autores com nomes idênticos ou cujas abreviações são iguais podem ser agregados erroneamente em um único autor que não representa nenhum pesquisador.

Atualmente, a literatura dispõe de inúmeros algoritmos para o processo de desambiguação automática de autores. Diferentes técnicas foram analisadas em duas publicações (FERREIRA; GONÇALVES; LAENDER, 2012; HUSSAIN; ASGHAR, 2017), contemplando diferentes classes: técnicas supervisionadas, não-supervisionadas, baseadas em grafos e baseadas em heurísticas. Entre as suas principais conclusões está que as técnicas supervisionadas, embora com melhor performance, são bastante demandantes em termos de dados, uma vez que precisam de um conjunto rotulado para o treinamento.

Em Sanyal; Bhowmick; Das (2021) os autores mapeiam todas as técnicas que foram utilizadas para essa atribuição de crédito em uma base de dados digital de ciências biomédicas, a PubMed. Além de catalogar as soluções, esse artigo aponta para uma lacuna na área, que é a falta de uma comparação (*benchmark*) de diferentes algoritmos no mesmo conjunto de dados abertos. Cada publicação incluída nessa revisão se ocupava de uma fração da base do PubMed e não permitiu uma análise comparativa entre elas. Também em Ferreira; Gonçalves; Laender (2012) os autores reiteram esse fato, e apontam para a falta de uma implementação em bibliotecas digitais reais, que crescem continuamente e demandam pela capacidade de desambiguação incremental.

Kim (2018), por sua vez, propõe uma forma de avaliar a desambiguação de nomes de autores em bibliotecas digitais com uma abordagem de triangulação com conjuntos rotulados. O desenvolvimento da proposta foi feito sobre um caso de estudo com o DBLP. Com a análise realizada, foi identificado que o algoritmo de desambiguação do DBLP é competitivo em comparação aos concorrentes nas métricas avaliadas. Com esse estudo foi demonstrado como a performance de desambiguação de bibliotecas digitais pode ser avaliada e foram fornecidos aprendizados para aplicações futuras da abordagem em outras bibliotecas digitais.

2.3 Estudos de comunidades científicas em bases digitais

Em uma escala acima da detecção de autoria, há as relações entre os agentes: autores publicam juntos e citam ou são citados em novas publicações. Nesse contexto, os autores Biryukov; Dong (2010) analisaram publicações feitas em eventos e conferências em ciência da computação em uma janela de 50 anos, extraídas da biblioteca DBLP. Nesse trabalho os autores particionaram manualmente a da ciência da computação em 14 áreas e analisaram a dinâmica de autores e das comunidades em cada uma dessas. Entre as tendências identificadas, foi de que algumas áreas haviam estabilizado a taxa de crescimento do número de publicações em eventos, como de arquitetura de computadores e redes, enquanto outras apresentavam expansão, como a área de processamento de linguagem natural (NLP). Além disso, os autores identificaram que há áreas cujos eventos têm comunidade mais fixa (*e.g.* criptografia), com poucos novos ingressantes e pouca debandada, e outras com muito dinamismo (*e.g.* redes), com muito fluxo de pesquisadores.

Ambos resultados podem ser aliados aos resultados do trabalho de Chakraborty et al. (2013), onde os autores analisaram a dinâmica de comunidades na ciência da computação ao longo de cinco décadas. Nesta publicação, os autores identificaram as comunidades-líder dentro da ciência da computação em termos de produção científica e citações. A área de preponderância em cada janela de tempo analisada é aquela que recebe o maior número de citações de outras áreas, caracterizada pela métrica de *inwardness*. Em geral, as áreas com maior expansão são aquelas capazes de produzir artigos que estimulam mais produções e que têm áreas correlatas dando suporte mais imediato. Espera-se assim que as áreas de comunidade mais fixa e fechada tenham menor probabilidade de tomar a frente da pesquisa em ciência da computação. De fato, a área mais fechada no trabalho de Biryukov; Dong (2010)(criptografia) não atingiu em nenhuma janela analisada o status de maior área pela métrica de citações. Além disso, áreas em expansão, como PLN, têm seu crescimento amparado por áreas correlatas, como a área de Aprendizado de Máquina.

Por fim, também usando a mesma biblioteca DBLP, os autores Zaiane; Chen; Goebel (2007) construíram um algoritmo protótipo, DBConnect, para minerar dados sobre comunidades em eventos. Uma diferença com relação à publicação de Biryukov; Dong (2010) é que este este sistema também constrói grafos tri-partidos (autor-conferência-tópico), onde os tópicos são extraídos dos próprios títulos das publicações. Entre as métricas que são extraídas desses grafos, uma em especial fornece uma listagem de tópicos relacionados, obtidos a partir de um passeio aleatório no modelo tripartite.

2.4 Estudos com dados da plataforma Lattes

Assim como as bibliotecas científicas digitais de publicações internacionais são temas recorrentes de pesquisa, a plataforma Lattes também é amplamente estudada sob o viés do estudo da ciência brasileira. Esta é uma grande base de dados que já teve seu crescimento retratado. No estudo de Dias; Moita (2018) foi feito um retrato da pesquisa brasileira e apresentado o grande volume de currículos cadastrados e sua curva crescente ao longo do tempo.

Nos artigos de Digiampietri et al. (2012); Dias; Moita; Dias (2016) foram realizadas extrações de dados do Lattes e feitas análises bibliométricas e de caracterização de redes sociais científicas. Nesse sentindo, foi demonstrado como a base fornece dados importantes para o estudos sobre a pesquisa no Brasil. A base, na época de ambos estudos, já contava com milhões de currículos. Uma informação interessante apresentada por Dias; Moita; Dias (2016) foi de que a Ciência da Computação pode ser considerada uma das áreas de atuação mais importantes na plataforma, já que estava entre as que mais possuíam currículos atrelados.

Em relação à análise por assuntos *i. e.* área, subárea ou especialidade específicos, foi proposta por Brito; Quoniam; Mena-chalco (2016) uma metodologia de busca por currículos baseada em termos. Nesse estudo, foi utilizada a área de Nanotecnologia como caso de estudo e, para esse assunto, foi criada uma lista de termos para serem usados na ferramenta de busca do Lattes para, assim, filtrar currículos de pesquisadores do assunto. A metologia pode ser replicada para outros assuntos com a criação de listas de termos correspondentes. Contudo, essa abordagem depende de especialistas para criar as listas de termo e se limita a encontrar pesquisadores que atuam ou atuaram de alguma forma com o determinado assunto, mas não é aplicada para cada produção dos pesquisadores em específico.

Um problema existente na plataforma, relatado em todos trabalhos apresentados, é que muitos dados são inseridos manualmente pelos pesquisadores em seus currículos. Essa característica cria algumas dificuldades para analisar os dados e pode gerar problemas como duplicação de artigos, inserção incorreta de nomes de conferências e revistas.

3 OBJETIVOS

O objetivo geral desse trabalho é desenvolver e avaliar algoritmos de processamento de linguagem natural para classificação automática da subárea e especialidade de produções científicas em inteligência artificial com base no título.

3.0.1 Objetivos Específicos

Considerando o objetivo geral do trabalho, os objetivos específicos são:

1. Criar um conjunto de dados de produções científicas de diferentes subáreas da computação, incluindo a subárea de inteligência artificial e contendo o título e o nome da subárea de cada publicação.

2. Criar um conjunto de dados de produções científicas de diferentes especialidades da inteligência artificial, contendo o título e o nome da especialidade de cada publicação.

3. Desenvolver e avaliar algoritmos para classificar se uma publicação pertence ou não à subárea de inteligência artificial, utilizando o conjunto de dados do objetivo específico 1;

4. Desenvolver e avaliar algoritmos para classificar a especialidade da IA à qual uma publicação pertence, utilizando o conjunto de dados do objetivo específico 2.

4 METODOLOGIA

4.1 Coleta de dados

Nesta seção, serão apresentadas as etapas de aquisição e organização dos conjuntos de dados. Os dados foram dispostos em dois conjuntos Subáreas e Especialidades, respectivos aos objetivos específicos 1 e 2. O texto apresenta, inicialmente, a definição dos conjuntos e, depois, como os dados foram coletados para atender o formato proposto.

4.1.1 Conjunto de dados Subáreas

O primeiro conjunto de dados a ser definido foi o Subáreas. A proposta é reunir o título de produções científicas e a subárea da computação à qual cada título pertence. Como a criação deste *data set* está relacionada ao objetivo específico 1, é necessário incluir dados de produções de diferentes subáreas da Computação além da Inteligência Artificial. Por isso, o primeiro passo foi selecionar quais as subáreas seriam incluídas.

Para tomar essa decisão, foi usada como referência a página de *top publications* do Google Scholar, em substituição às subáreas definidas pela CAPES. Essa página lista, por ordem de relevância, os principais veículos de publicação em diferentes categorias e subcategorias. Assim, foi utilizada a página da categoria "*Engineering & Computer Science*" (Figura 1) para selecionar algumas das subcategorias existentes. Das subcategorias presentes, foram desconsideradas as que são ligadas apenas à Engenharia da Computação. Além disso, foram escolhidas somente algumas subcategorias mais relacionadas com disciplinas base da Ciência da Computação, tendo como referência os cursos da Universidade Federal de Pelotas. Dessa maneira, a lista de subáreas da Ciência da Computação selecionadas para o conjunto foram retiradas dessa lista de subcategorias e dispostas na Tabela 1.

Tendo a lista das subáreas, o próximo passo foi definir como a subárea de cada exemplo do conjunto de dados deveria ser identificada. A decisão foi de definir a subárea de cada trabalho pelo seu veículo de publicação. Consequentemente, também

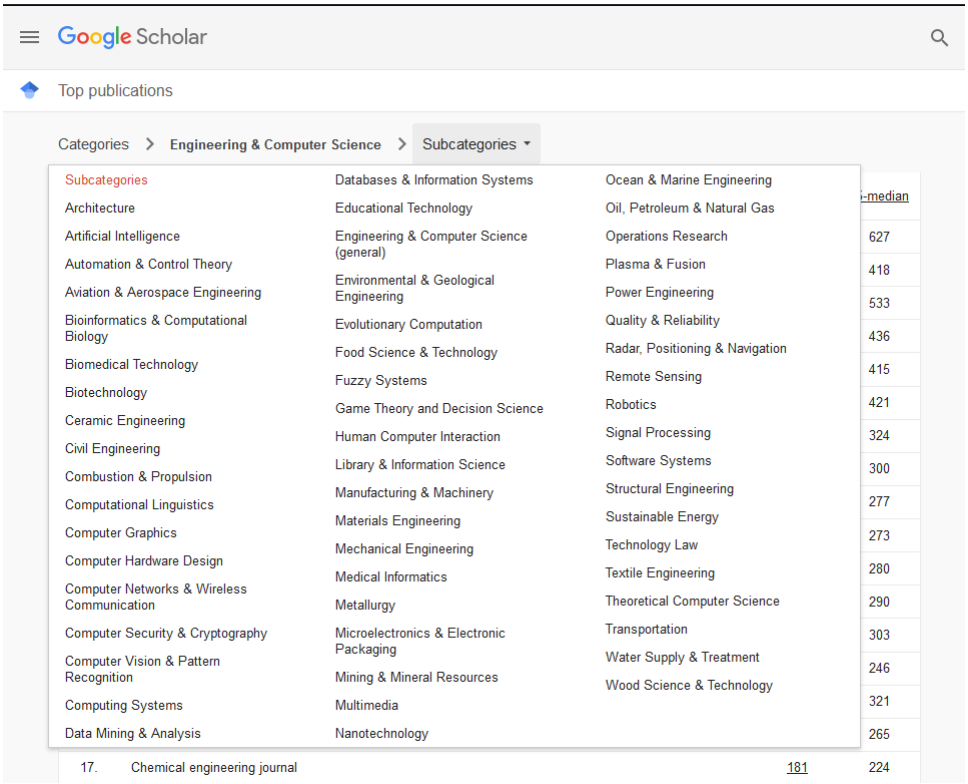


Figura 1 – Captura de tela da página "*Top publications*" na categoria "*Engineering & Computer Science*" com as subcategorias sendo exibidas.

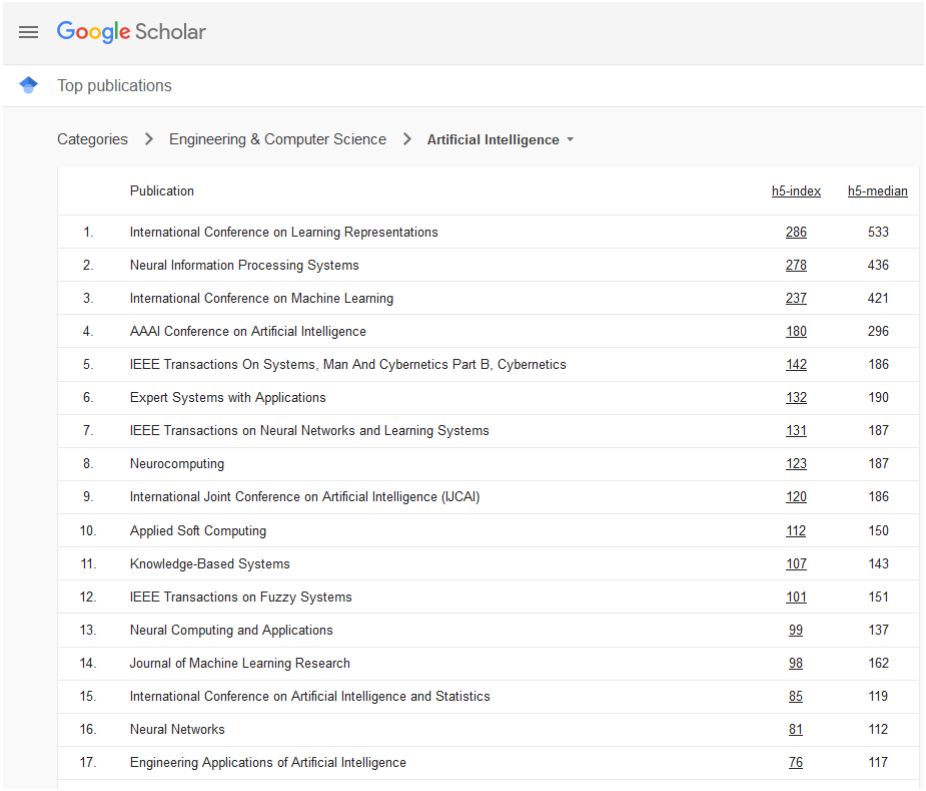
Tabela 1 – Subáreas selecionadas.

	Subárea
1	Artificial Intelligence
2	Computer Graphics
3	Computer Hardware Design
4	Computer Networks & Wireless Communication
5	Theoretical Computer Science
6	Human Computer Interaction
7	Microelectronics & Electronic Packaging

foi necessária a seleção de veículos que representassem cada subárea. Para essa seleção, foi utilizada novamente a página *top publications* do Google Scholar. Nesse caso, foram selecionados alguns veículo de cada subárea da Tabela 1, que também são subcategorias dentro da página.

Os veículos de publicação selecionados e subárea a que foram atribuídos podem ser conferidos na Tabela 15 do Apêndice A. No caso específico dos veículos de publicação da subárea de Inteligência Artificial, a seleção foi um pouco diferente. Ao escolher os veículos, foram considerados apenas aqueles com temática genérica de IA e não conferências ou revistas com especialidades específicas (*e.g. Knowledge-Based Systems, Neural Networks*). Assim, foram selecionados apenas alguns veículos da subcategoria "*Artificial Intelligence*"(Figura 2), visto que muitos da lista eram de campos específicos da IA. Além disso, foram incluídos dois veículos de publicação do

Brasil que se encaixam nesses critérios e são importantes para a área no país. Dessa forma, os veículos de publicação em IA foram os dispostos na Tabela 2.


Google Scholar

Top publications

Categories > Engineering & Computer Science > **Artificial Intelligence**

	Publication	h5-index	h5-median
1.	International Conference on Learning Representations	286	533
2.	Neural Information Processing Systems	278	436
3.	International Conference on Machine Learning	237	421
4.	AAAI Conference on Artificial Intelligence	180	296
5.	IEEE Transactions On Systems, Man And Cybernetics Part B, Cybernetics	142	186
6.	Expert Systems with Applications	132	190
7.	IEEE Transactions on Neural Networks and Learning Systems	131	187
8.	Neurocomputing	123	187
9.	International Joint Conference on Artificial Intelligence (IJCAI)	120	186
10.	Applied Soft Computing	112	150
11.	Knowledge-Based Systems	107	143
12.	IEEE Transactions on Fuzzy Systems	101	151
13.	Neural Computing and Applications	99	137
14.	Journal of Machine Learning Research	98	162
15.	International Conference on Artificial Intelligence and Statistics	85	119
16.	Neural Networks	81	112
17.	Engineering Applications of Artificial Intelligence	76	117

Figura 2 – Captura de tela da página "*Top publications*" na subcategoria "*Artificial Intelligence*" com os *top* veículos de publicação listados.

Tabela 2 – Veículos de publicação na subárea de Inteligência Artificial.

	veículo de publicação
1	AAAI Conference on Artificial Intelligence
2	International Joint Conferences on Artificial Intelligence
3	Artificial Intelligence - Journal
4	The Florida AI Research Society
5	European Conference on Artificial Intelligence
6	International Journal on Artificial Intelligence Tools
7	Brazilian Symposium on Artificial Intelligence
8	Brazilian Conference on Intelligent Systems

4.1.2 Conjunto de dados Especialidades

O conjunto Especialidades tem o objetivo de reunir os títulos de publicações e suas respectivas Especialidades da IA, de acordo com o objetivo específico 2. Assim como no conjunto Subáreas, a especialidade de cada título foi definida pelo veículo de publicação. Contudo, a forma de escolher as especialidades e os veículos a comporem o conjunto foi diferente da apresentada anteriormente.

Primeiramente, foram escolhidas as especialidades a serem utilizadas no conjunto.

Para isso, foi feita uma busca no histórico de chamadas de trabalhos da Conferência AAAI. Essa é uma grande conferência internacional cujo propósito é promover a pesquisa em IA e a troca científica entre pesquisadores, cientistas e engenheiros. Sendo assim, foram buscados os tópicos das chamadas de trabalho de 2016 à 2022 apresentados no site oficial. Os tópicos mais frequentes no período foram estabelecidos como especialidades para esse estudo e estão apresentados na Tabela 3.

Tabela 3 – Especialidades selecionadas.

	Especialidade
1	Natural Language Processing
2	Machine Learning
3	Planning
4	Knowledge Representation & Reasoning
5	Robotics and Perception
6	Multiagent Systems
7	Computer Vision

Assim como na definição do conjunto Subáreas, a etapa seguinte à definição das especialidades foi a seleção dos veículos de publicação. Nesse caso, os veículos não foram extraídos das listas de *top publications* do Google Scholar, pois as especialidades selecionadas não foram propriamente categorias ou subcategorias do *rank* de veículos da plataforma. Ao invés disso, os veículos de publicação foram buscados através da pesquisa dos nomes das especialidades no Google Acadêmico e os resultados das buscas foram analisados manualmente. Nessa análise manual, foi feita a procura e leitura de *websites* de veículos de publicação para encontrar aqueles que mais se encaixavam nas especialidades escolhidas de acordo com a descrição fornecida. Uma abordagem semelhante para definição dos tópicos de veículos foi feita por Alwahaishi; Martinovič; Snášel (2011). Dessa maneira, foi gerada a lista de veículos de publicação presente na Tabela 16 do Apêndice A.

4.1.3 *Scripts* de extração desenvolvidos

A fonte de dados escolhida para extração foi o *DBLP computer science bibliography*. Esse repositório provê informação bibliográfica dos maiores *journals* e anais de conferências da Ciência da Computação. Ele foi, originalmente, criado pela Universidade de Trier em 1993 e atualmente é operado pela Schloss Dagstuhl.

Essa base de dados foi escolhida por permitir acesso público via internet. Além disso, é uma base criada especificamente para produções em Computação, o que vai de encontro com o estudo de caso proposto nesse trabalho. Ainda, oferece uma *Application Programming Interface* (API) para fazer consultas e, portanto, também facilita o desenvolvimento de programas de computador próprios para obter dados.

Para se obter os dados do DBLP, foram desenvolvidos programas próprios para consultas automatizadas na API. O desenvolvimento foi feito na linguagem de programação Python. Assim, dois scripts foram desenvolvidos: um para baixar dados pertinentes ao conjunto de dados Subáreas e o outro para o conjunto Especialidades.

A API utilizada permite fazer buscas textuais e acessar resultados paginados em formato *JavaScript Object Notation* (JSON). Dessa forma, é possível buscar produções científicas pelo nome do veículo de publicação. Entretanto, o nome do veículo no DBLP é representado por um campo textual (*venue*) que pode ser uma sigla ou abreviação, por isso é necessário saber qual sigla representa cada veículo. Sendo assim, todos os veículos selecionados precisaram ser mapeados manualmente e, para cada um, foi atribuída uma sigla que representa seu nome dentro da base DBLP (*e.g.* a revista Artificial Intelligence é representada por "Artif. Intell." no JSON resultante da busca). Desse modo, as siglas/abreviações foram incorporadas nos scripts desenvolvidos. A lista completa do mapeamento criado pode ser vista na Tabela 17 no Apêndice B.

A tarefa dos scripts criados foi baixar os dados pertinentes aos conjuntos Subáreas e Especialidades para salvá-los em arquivos no formato *comma-separated values* (CSV). Nesse sentido, foram baixados todos os registros encontrados com o campo *venue* pertencentes ao conjunto de siglas definidas nesse trabalho. Além do título em língua inglesa e do nome do veículo de publicação, foram extraídos os dados de tipo e ano. Esses atributos foram utilizados na etapa de limpeza de dados apresentada na Seção 4.2.

4.2 Limpeza dos dados

Com todos os dados de ambos os conjuntos baixados, ainda é necessária a limpeza e a aplicação de alguns filtros para melhorar a qualidade dos mesmos. Sendo assim, foi desenvolvido em linguagem Python um programa para aplicar essas modificações nos conjuntos originais. Objetivamente, os processos aplicados foram:

1. **Filtro dos últimos 10 anos**: foram removidos registros anteriores ao ano de 2002, pois o estudo considera dados de subáreas, especialidades e veículos de publicação mais recentes e, como a base possui registros antigos, estes foram removidos. A janela de tempo de 10 anos é maior do que a que foi usada na análise do histórico de chamadas da AAAI, mas isso foi feito para manter um volume de dados mais alto, o que influencia nas etapas de treino dos modelos aplicados.

2. **Remoção de editorias**: um dos tipos de registro retornados pela API é a editoria. Esses registros são indicados pelo campo *type* com o valor *editorship* e não

possuem um título de artigo propriamente, mas são os *proceedings* ou anais de eventos. Por isso, esses registros também foram descartados.

3. **Produções com títulos que incluem o nome do veículo**: um problema encontrado nos dados extraídos foi a presença de registros que, mesmo não sendo do tipo *editorship*, são anais ou algum outro tipo de editoria e que não representam apenas um trabalho. Para lidar com essa característica, esses registros também foram removidos.

4. **Outros casos de editorias não identificadas**: mesmo com as etapas anteriores aplicadas, ainda permaneceram alguns registros de editorias sem a classificação de tipo adequada. Por isso, aqueles que foram encontrados manualmente, foram também removidos.

Após a execução dos processos de limpeza de dados listados, a quantidade de exemplos foi reduzida em relação ao tamanho inicial dos conjuntos. Além disso, no caso do conjunto Subáreas, todos os títulos publicados em veículos de subárea diferente de Inteligência Artificial foram rotulados como Não IA. Pois, a tarefa de classificação aplicada a esses dados foi uma classificação binária (detalhada na Seção 4.4). Nessas circunstâncias, o número de registros resultantes e a distribuição entre categorias nos conjuntos de Subáreas e Especialidades foram os apresentados nas Figura 3 e Figura 4, respectivamente.

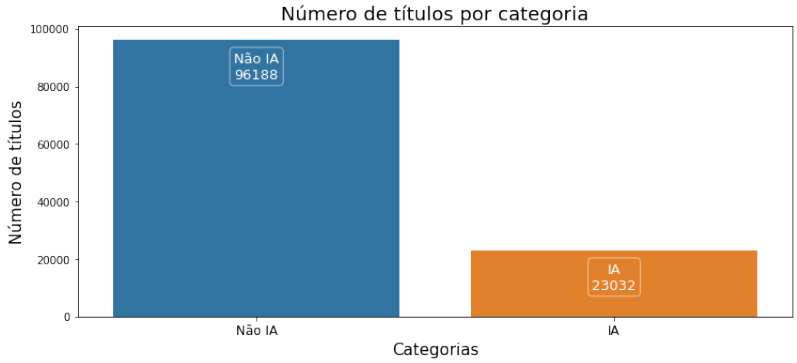


Figura 3 – Títulos por categoria - Subáreas.

4.3 Definição de conjuntos de treino e teste

Os modelos de classificação propostos nesse trabalho são modelos de aprendizado de máquina supervisionados, portanto precisam de conjuntos rotulados de treino e teste para serem desenvolvidos (SOKOLOVA; LAPALME, 2009). Sendo assim, foi necessária a divisão dos conjuntos de dados originais em porções destinadas às duas categorias.

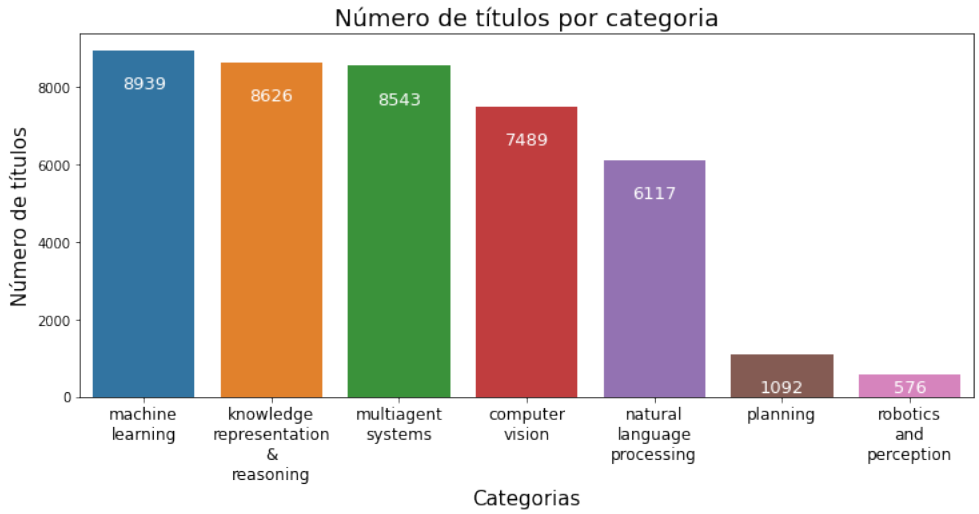


Figura 4 – Títulos por categoria – Especialidades.

Para realizar essa separação, foi desenvolvido um programa na linguagem Python para aplicar o método *train_test_split* da biblioteca scikit-learn (PEDREGOSA et al., 2011). Esse método permite fazer uma divisão estratificada de treino e teste, a qual foi empregada. Isso significa que, ao separar os conjuntos, a distribuição de exemplos das classes é mantida. Essa característica é necessária para garantir um número razoável de exemplos das classes minoritárias em ambos treino e teste. Nessa etapa, foram utilizados 20% de ambos os conjuntos para teste. Os 80% restantes foram usados como conjunto de treino/desenvolvimento. Os números absolutos de exemplos estão dispostos na Tabela 4.

Tabela 4 – Volume de exemplos por conjunto de dados.

	Subáreas	Especialidades
Exemplos de treino	95.376 (80%)	33.105 (80%)
Exemplos de teste	23.844 (20%)	8.277 (20%)
Total	119.220 (100%)	41.382 (100%)

4.4 Especificação da tarefa dos modelos

Assim como foram criados dois conjuntos de dados, também foram desenvolvidos dois modelos: Modelo Subáreas e Modelo Especialidades para atender os objetivos específicos 3 e 4, respectivamente. Em Aprendizado de Máquina existem diversos tipos de classificação, entre eles a binária e a multi-classe (SOKOLOVA; LAPALME, 2009), as quais foram empregadas pelos modelos. Sendo assim, a tarefa de cada modelo foi definida como:

- **Modelo Subáreas:** a tarefa a ser executada é uma classificação binária, ou seja, o modelo deve classificar cada exemplo como positivo ou negativo, sendo

os positivos os títulos pertencentes à Subárea da IA e negativos caso contrário. A entrada para esse modelo deve ser o título de um artigo da computação em língua inglesa.

- **Modelo Especialidades:** deve realizar uma classificação multi-classe, ou seja, dentre todas as especialidades selecionadas, o modelo deve indicar qual a correta para cada exemplo. A entrada para esse modelo deve ser o título de um artigo de inteligência artificial em língua inglesa.

4.5 Representações utilizadas

Com os dados disponíveis é necessário prepará-los para que os modelos possam ler os exemplos de entrada para realizar as a etapas de treino e teste. Como a entrada são títulos de trabalhos, ou seja, uma sequência de caracteres que forma uma sentença, esses exemplos não podem ser lidos diretamente pelos modelos propostos. Assim, é necessária uma codificação ou transformação desses textos em um tipo de representação numérica que seja acessível aos modelos. Esse tipo de transformação é comum para aplicações de Processamento de Linguagem Natural).

Existem diferentes técnicas, algoritmos e modelos para fazerem esse tipo de transformação de texto. Os tipos de processamentos testados no desenvolvimento das soluções propostas foram:

- **TF:** *Term frequency* (frequência do termo) é uma forma de calcular um valor para cada palavra do documento. Esses valores, por sua vez, podem ser utilizados para classificação. Nesse tipo de processamento, cada exemplo pode ser transformado em um vetor cujas posições representam as palavras do corpus e os valores são a frequência que a palavra aparece no documento. Assim, o vetor pode ser usado como um exemplo para um modelo de aprendizado de máquina.

- **TF-IDF:** *Term frequency–inverse document frequency*(frequência do termo–inverso da frequência nos documentos) pode ser calculada e distribuída em um vetor posicional para cada palavra, assim como no anterior. A diferença é que a forma de calcular o valor, nesta abordagem, tenta considerar a importância das palavras com base na frequência em que ocorrem no corpus, então não leva em consideração apenas o documento. A forma de calcular esse valor é o expresso na Equação 1.

$$TFIDF_{i,j} = TF_{i,j} \cdot \log\left(\frac{N}{DF_i}\right) \tag{1}$$

Na equação, $TF_{i,j}$ representa a frequência do termo i no documento j, DF_i o número de documentos contendo i e N, o total de documentos.

- **Spacy en_core_web_sm:** esse é um *pipeline* da biblioteca SpaCy treinado em língua inglesa e que suporta várias operações, dentre elas, transformar uma sentença ou documento em um vetor através do método *vector*.

- **BERT:** *Bidirectional Encoder Representations from Transformers* (BERT) (DEVLIN et al., 2018) é um modelo desenvolvido pela Google que utiliza uma técnica de "*embeddings* contextualizados" que permite que o modelo aprenda representações para as palavras em um contexto específico. Além disso, o modelo também possui uma maneira de representar as sentenças com vetores e pode ser refinado ao ser treinado juntamente com algum classificador em uma tarefa específica, prática recomendada pelos autores.

4.6 Seleção e treinamento dos modelos

Nesta seção são descritas as atividades realizadas para a escolha e desenvolvimento dos modelos. O objetivo foi avaliar diferentes abordagens e tipos de modelos de aprendizado de máquina, considerando-os como candidatos à Modelo Subáreas e Modelo Especialidades. Dessa forma, foi necessário desenvolver *pipelines* para executar as etapas de preparação de dados, treinamento e teste. Os modelos finais escolhidos foram aqueles que desempenharam melhor nos testes realizados.

Nesse sentido, a transformação de dados textuais em representações numéricas compõe apenas uma parte dos *pipelines* de classificação. Após essas transformações, próxima etapa foi utilizar os dados gerados como entradas para os modelos de aprendizado de máquina. Contudo, para fazer a escolha e treinamento dos modelos, foram necessários alguns processos de validação.

4.6.1 Validação inicial

Nessas circunstâncias, inicialmente, foi realizada uma seleção de modelos e, após, foram desenvolvidos *pipelines* para implementá-los. O objetivo foi testar diferentes modelos de aprendizado de máquina e os diferentes tipos de representações apresentados na seção anterior. Assim, foram testadas combinações entre os modelos e tipos de processamento de texto. Tendo em vista economizar tempo de computação, nessa etapa, os modelos não foram submetidos a busca dos melhores hiper-parâmetros. Objetivamente, essa seleção consistiu em escolher alguns modelos conhecidos para classificação e treiná-los sem otimização de hiper-parâmetros para encontrar as melhores combinações entre *encoding* e modelo.

4.6.1.1 Pipelines scikit-learn

Para validar as técnicas apresentadas, foram criados *pipelines* para os modelos e *encodings*. Assim, cada um dos modelos foi testado em combinação com todas as

formas de representação listadas na seção anterior. A única codificação que não foi testada com todos modelos foi a BERT, que teve um *pipeline* próprio desenvolvido. Para os outros casos foram aplicados os modelos seguintes modelos:

- **RandomForestClassifier**: é um método de aprendizado de máquina de classificação baseado em árvores de decisão. Ele constrói várias árvores de decisão de forma aleatória e combina as suas saídas para aumentar a precisão. Isso o torna mais robusto contra sobreajuste comparado com o uso de uma única árvore de decisão (BREIMAN, 2001).
- **LinearSVC**: O LinearSVC foi criado como uma alternativa ao SVM (Support Vector Machine), que foi originalmente apresentado por Cortes; Vapnik; Saitta (1995). Ele é uma versão linear do SVM, desenhada para ser mais eficiente ao lidar com grandes conjuntos de dados. O LinearSVC foi projetado para ser mais rápido e usar menos memória do que outras implementações do SVM.
- **XGBoost**: é um algoritmo de aprendizado de máquina baseado em floresta aleatória aprimorado, apresentado por Chen; Guestrin (2016). Ele é construído sobre o algoritmo de Gradient Boosting e inclui várias otimizações para torná-lo mais eficiente e preciso. O XGBoost foi projetado para melhorar a eficiência do Gradient Boosting e possibilitar o manuseio de grandes conjuntos de dados e grande número de recursos.

Esses modelos foram empregados juntamente com as codificações usando o recurso de *pipelines* da scikit-learn. Dos modelos citados, apenas o XGBoost não faz parte da biblioteca, porém o método é compatível com os *pipelines*.

4.6.1.2 Pipeline BERT

Um bom motivo para utilizar o modelo BERT é que ele pode ser treinado junto com um modelo de classificação para aprender os *embeddings* para uma tarefa específica, processo conhecido como ajuste fino. Por esse motivo foi necessária a criação de um *pipeline* específico. Assim, foi desenvolvido um *pipeline* diferente dos que foram usados nos modelos apresentados anteriormente.

Na arquitetura desenvolvida, a versão utilizada foi a "bert_en_uncased_L-12_H-768_A-12" disponibilizada pelo Tensorflow Hub (ABADI et al., 2015) e carregada como uma camada Keras (CHOLLET et al., 2015), biblioteca que facilita o uso de modelos de redes neurais. Em relação à preparação dos dados, o modelo necessita de três entradas *input_word_ids*, *input_mask*, *input_type_ids*, que são geradas a partir do texto original. Para transformar os títulos para esse formato, foi utilizado o construtor de exemplos do próprio pacote do modelo baixado do TensorFlow Hub. Esse método transforma o texto para o formato esperado pelo modelo utilizando o tokenizador

BERT. A saída do modelo utilizada foi a *pooled output* referente ao *token* [CLS]. Esse token é o *embedding* que representa toda sentença e é a saída que foi utilizada, pois o objetivo é classificar o título. Assim, esse vetor serve como entrada para o modelo de classificação. A ilustração do *pipeline* completo está na Figura 5 e as camadas do modelo Keras na Figura 6.

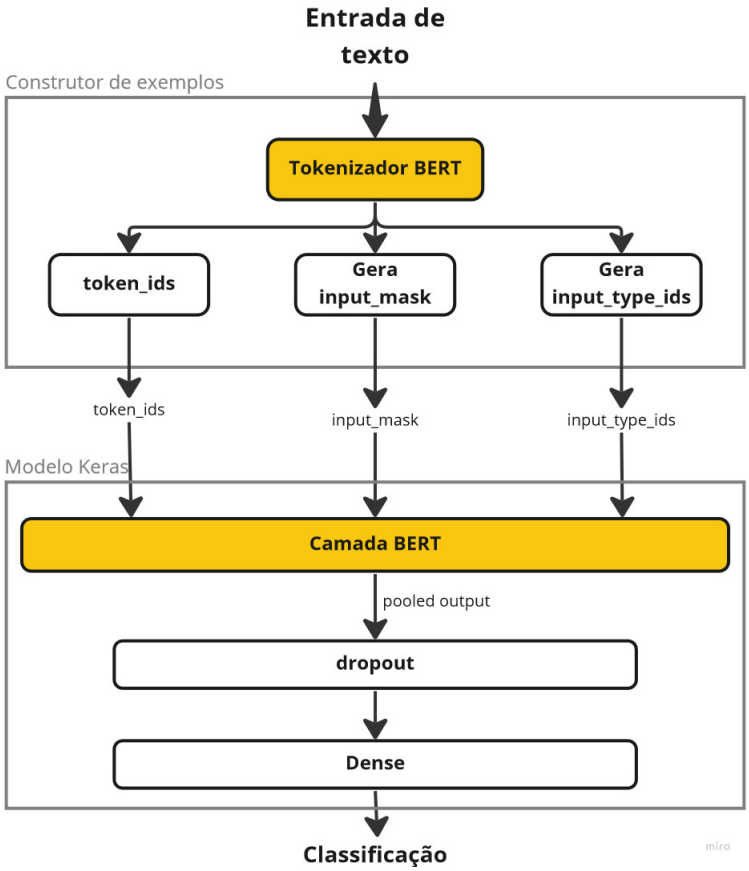


Figura 5 – *Pipeline* BERT ilustrado. No topo, está representada a entrada de texto que é o título de cada produção. Dentro do retângulo "Construtor de exemplos", estão ilustrados componentes do método que transforma o texto em uma entrada válida para o encoder. As três saídas do construtor são as entradas do modelo Keras. O BERT é carregado como uma *layer* do modelo e sua saída é correspondente ao token [CLS]. O vetor de saída passa por uma camada de Dropout e, depois, serve como entrada para o modelo de classificação. O modelo Dense então calcula sua função de saída e esta é utilizada como classificação. Todos componentes dentro do retângulo "Modelo Keras" são treinados juntos.

No treinamento, os parâmetros utilizados foram: 4 épocas, otimizador Adam (KINGMA; BA, 2014) com *learning rate* $2e-5$ e tamanho dos *batches* de treinamento de 32. Esses parâmetros são os mesmos ou estão na mesma faixa dos apresentados no artigo original (DEVLIN et al., 2018). Além disso, foi incluída uma camada de Dropout (HINTON et al., 2012) entre a saída do BERT e a entrada da rede neural com o intuito de diminuir a chance de sobreajuste do modelo, o valor utilizado foi de 0.1 pois também foi um dos valores utilizados pelos autores nos testes de ajuste fino.

O modelo de classificação utilizado junto ao BERT foi uma rede neural artificial, mais precisamente, a implementação Dense da biblioteca Keras. As redes neurais

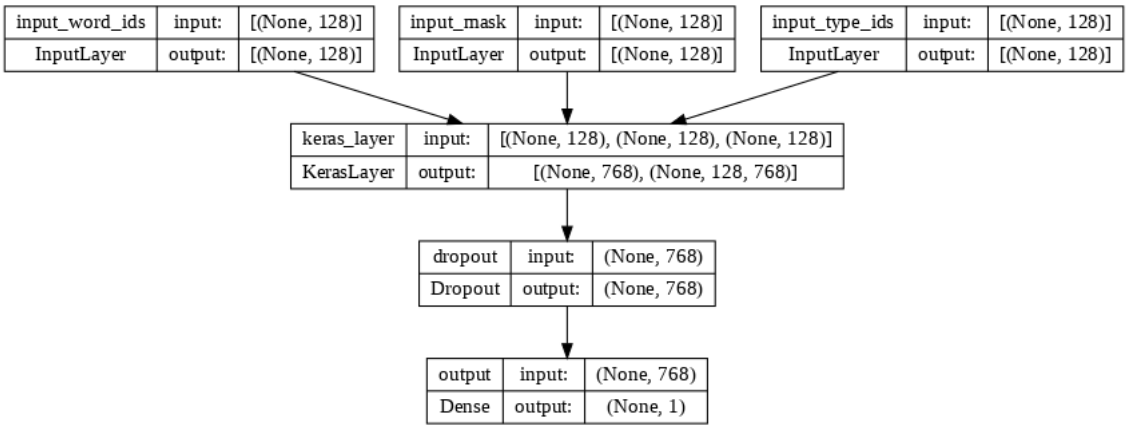


Figura 6 – Ilustração do Modelo Keras candidato à Modelo Subáreas. Na parte superior da imagem, está a camada de entrada do modelo, composta pelas três saídas do construtor de exemplos. Abaixo, a *keras_layer* é o modelo BERT incorporado. Em sequência, a camada de Dropout e, finalmente, o modelo de classificação. Nesta figura, está ilustrado apenas o modelo desenvolvido para Subáreas, visto que sua saída é de tamanho um. O modelo candidato à Modelo Especialidades desenvolvido, tem especificações muito parecidas, nessa ilustração, a única diference seria a *output* apresentada, no lugar de 1 seria 7.

são modelos versáteis que podem ser aplicados em diversas tarefas, entre elas, a classificação. Esse modelo foi escolhido pela facilidade de implementá-lo na biblioteca Keras e porque não foi incluído nos *pipelines* do scikit-learn desenvolvidos.

Neste ponto, os dois modelos a serem desenvolvidos tiveram algumas diferenças. Para o modelo Especialidades a função de *loss* (perda) foi a *SparseCategoricalCrossentropy* e a saída classificação foi um vetor de tamanho sete com ativação *softmax*. Já no modelo Subáreas a função foi *BinaryCrossentropy* e a saída foi um único valor. Para ativação foram testadas as funções sigmoid e ReLu (KRIZHEVSKY; SUTSKEVER; HINTON, 2017). Ambas implementações das funções de perda são da biblioteca Keras.

Ambos *pipelines* foram desenvolvidos e validados com o conjunto de treino. Isso foi feito dividindo o conjunto de treino novamente para que os modelos treinassem com 80% e fossem validados com 20% sem usar o conjunto de teste original em nenhuma etapa. Para fazer essa divisão, foi usado o método *train_test_split* com estratificação de classes. No final da validação, os melhores modelos foram selecionados para uma posterior etapa de busca de hiper-parâmetros.

4.6.2 Busca de hiper-parâmetros

Após a validação inicial dos *pipelines* propostos, os melhores, de acordo com as métricas adotadas, foram selecionados para personalização de hiper-parâmetros. Esses foram submetidos a mais testes com o conjunto de treino/validação com o objetivo de encontrar o modelo ótimo. Após, o melhor candidato a Modelo Subáreas e o melhor candidato a Modelo Especialidades nessa busca foram selecionados para teste. O

teste consistiu em usar os modelos para classificar o conjunto de teste dos conjuntos Subáreas e Especialidades, separado na etapa inicial de construção dos conjuntos de dados e nunca visto pelos modelos nas etapas anteriores. Os resultados detalhados cada etapa estão no Capítulo 5.

4.6.2.1 Pipelines scikit-learn

O modelo do scikit-learn que melhor performou na primeira etapa de validação foi selecionado para busca de hiper-parâmetros. O modelo em questão foi LinearSVC. Assim, foi feita uma busca pelos melhores hiper-parâmetros com o método GridSearchCV da biblioteca. Contudo, apenas alguns hiper-parâmetros foram incluídos na busca. Essa limitação foi colocada pois seria muito custoso testar todas variações possíveis. A escolha teve como base a descrição encontrada na documentação da biblioteca. Os hiper-parâmetros buscados foram *loss*, *dual* e *fit-intercept*, para o possível Modelo Subáreas. Para o candidato Modelo Especialidades foi incluída a busca pelo melhor valor para o hiper-parâmetro *multi_class*.

Todos os hiper-parâmetros citados são do tipo *booleano* ou uma categoria definida por *string*. Sendo assim, todos os valores possíveis para cada hiper-parâmetro foram testados, pois o método GridSearch testa todas combinações. As combinações que não foram testadas são aqueles que desobedecem alguma regra do modelo (*e.g.* a combinação *penalty*='l2' e *loss*='hinge' não são suportadas quando *dual*=False).

4.6.2.2 Pipeline BERT

O modelo BERT também foi destaque e teve resultados próximos ao do modelo SVC. Por isso, foram testadas algumas variações no *pipeline* com intuito de melhorar os resultados na validação. Os parâmetros ajustados foram o número de épocas de treinamento e a função de ativação, que foram a *sigmoid* e ReLu, como mencionado previamente.

5 RESULTADOS

Neste capítulo serão apresentados os resultados obtidos com os modelos nas etapas de validação e teste. O objetivo final é elencar a melhor combinação de processamento de texto e modelo de aprendizado de máquina para o Modelo Subáreas e Modelo Especialidades e medir o seu desempenho de acordo com algumas métricas.

A seleção da métrica adequada para discriminar a solução ótima a fim de obter um classificador otimizado é uma etapa crucial (HOSSIN; SULAIMAN, 2015). Assim, as métricas de avaliação utilizadas foram calculadas com base em algumas variáveis fundamentais: verdadeiros positivos (tp), verdadeiros negativos(tn), falsos positivos(fp) e falsos negativos(fn). Dessa forma, as medidas avaliadas foram:

- **Acurácia:** representa a proporção de classificações corretas;

$$Acuracia = \frac{tp + tn}{tp + tn + fp + fn} \tag{2}$$

- **Acurácia balanceada:** é uma medida de acurácia para conjuntos de dados desbalanceados, onde 0 indica o pior valor possível e 1 indica o melhor, valores maiores que 0,5 são considerados melhores que uma classificação aleatória;

$$AcuraciaBalanceada = \frac{1}{2} \times \left(\frac{tp}{tp + fn} + \frac{tn}{tn + fp} \right) \tag{3}$$

- **Precisão:** é a habilidade de não rotular exemplos negativos como positivos, expressa pela Equação 4;

$$Precisao = \frac{tp}{tp + fp} \tag{4}$$

- ***Recall*:** é habilidade de rotular todos exemplos positivos corretamente, expressa pela Equação 5;

$$Recall = \frac{tp}{tp + fn} \tag{5}$$

- **F1-score:** é a média harmônica entre precisão e *recall*, expressa pela Equação 6.

$$F1 = \frac{2 * Precisao * Recall}{Precisao + Recall} = \frac{2 * tp}{2 * tp + fp + fn} \quad (6)$$

Os resultados estão divididos em Modelo Subáreas e Modelo Especialidades e, ainda, organizados por etapas: validação inicial, busca de hiper-parâmetros e teste. Para ilustrar o processo, existe a Figura 7, que indica as atividades que foram executadas e a combinação de modelos que performou melhor como ambos modelos Subáreas e Especialidades.

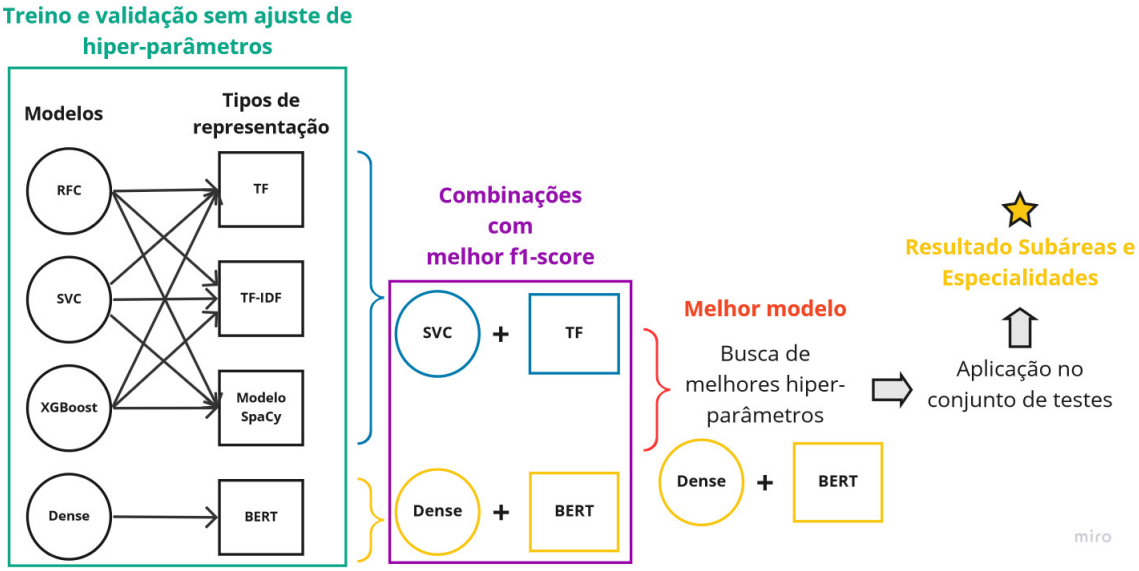


Figura 7 – Ilustração do processo que resultou no melhor Modelo Subáreas e Modelo Especialidades. A sequência em que os passos foram executados é da esquerda para direita. No primeiro retângulo à esquerda, está ilustrada a etapa descrita na validação inicial, onde as combinações dentro da chave azul são os *pipelines* da scikit-learn; já a chave amarela compreende o *pipeline* que foi desenvolvido inicialmente para o BERT; Cada seta que conecta um modelo (círculo) com um tipo de representação (quadrado) expressa uma combinação testada. As melhores combinações foram modelo LinearSVC com processamento TF e o modelo Dense treinado com o BERT. Estes modelos estão representados no segundo retângulo. Assim, sob o texto "Melhor modelo", está ilustrado o resultado da etapa de busca de hiper-parâmetros, onde o modelo BERT teve resultados melhores. Por fim, estão as etapas de teste e resultados finais do modelo. Sendo assim, para ambos modelos Subáreas e Especialidades, o *pipeline* BERT foi o que obteve melhores resultados.

5.1 Modelo de Subáreas

Todas abordagens de classificação desenvolvidas sobre o conjunto de dados Subáreas foram consideradas, inicialmente, como candidatas à Modelo Subáreas. Contudo, somente o classificador que teve os melhores resultados nas etapas de validação e validação após o refinamento de hiper-parâmetros foi definido como Modelo Subáreas.

Para as etapas de validação, foi utilizado o método *classification_report* da scikit-learn como base. Esse método calcula todas as métricas apresentadas nesse capítulo e também o número de exemplos em cada classe. Para etapa final de teste, também foi utilizada uma visualização de matriz de confusão.

5.1.1 Validação inicial

Inicialmente, os modelos aplicados no scikit-learn foram treinados e validados sem alteração dos hiper-parâmetros padrões da biblioteca. Já para o caso da arquitetura desenvolvida para o BERT, foi criado um *pipeline* base no qual não foi feita uma busca pelos melhores hiper-parâmetros. Esse modelo base foi criado de acordo com a arquitetura apresentada na Subseção 4.6.1, sendo que a função de ativação base foi a sigmoid.

Nessas circunstâncias, de todas as combinações testadas, as que desempenharam melhor foram o modelo SVC com processamento TF e o modelo Dense com processamento BERT. Os relatórios de classificação dessas abordagens podem ser vistos nas Tabela 5 e Tabela 6, respectivamente. Além disso, a acurácia foi de 93% e o f1-score macro 88% no SVC, já na rede neural foram 93% e 89%, respectivamente. O f1-score macro foi utilizado nas análises pois é a média entre o f1-score das classes, dando igual importância para cada classe e, portanto, sendo mais adequado para classes desbalanceadas.

Tabela 5 – Validação inicial: Relatório de classificação modelo SVC + TF – Subáreas.

Classe	**Precisão**(%)	**Recall**(%)	**F1-score**(%)	**Exemplos**
Não IA	95	97	96	15391
IA	84	77	80	3685

Tabela 6 – Validação inicial: Relatório de classificação modelo Dense + BERT – Subáreas

Classe	**Precisão**(%)	**Recall**(%)	**F1-score**(%)	**Número de exemplos**
Não IA	95	97	96	15391
IA	86	78	82	3685

5.1.2 Busca de hiper-parâmetros

Como os modelos SVC e Dense foram os que desempenharam melhor na validação inicial, somente esses passaram pela busca de hiper-parâmetros. No caso do modelo SVC, a busca foi feita com a aplicação do método GridSearch para os parâmetros *loss*, *dual* e *fit_intercept*. A melhor combinação resultante foi a própria combinação padrão, portanto o resultado da do refinamento do modelo nessa etapa foi o mesmo da validação inicial.

Já o modelo utilizando BERT teve outras modificações testadas. O modelo foi treinado, inicialmente, por 4 épocas. Durante o treinamento foi observado que a melhor acurácia e *loss* no conjunto de validação eram obtidas treinando por apenas 2 épocas (Figura 8 e Figura 9, respectivamente). Então essa alteração foi feita no modelo. Além disso, também foi testada a função de ativação ReLu, mas os resultados foram significativamente inferiores. Dessa forma, o modelo final manteve-se com a configuração de 2 épocas e ativação sigmoid. Os resultados no conjunto de validação estão dispostos na Tabela 7. A acurácia atingida pelo modelo foi de 93% e f1-score macro de 89%, os mesmos valores da etapa anterior, entretanto a precisão para classe IA melhorou.

Como o *pipeline* utilizando o modelo Dense em conjunto com o BERT obteve as melhores métricas, ele foi definido como Modelo Subáreas.

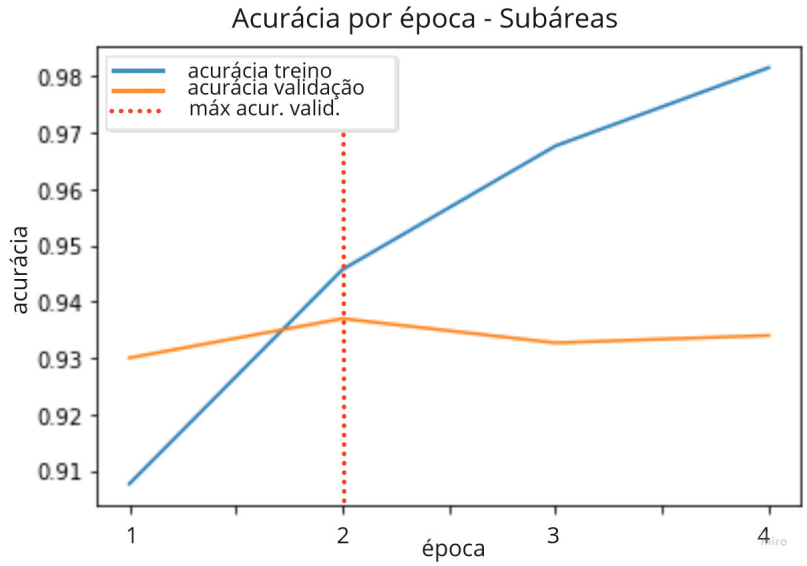


Figura 8 – Acurácia no treino Dense + BERT – Subáreas: No gráfico da figura é possível observar que a acurácia mais alta na validação foi obtida na segunda época (linha pontilhada vermelha).

Tabela 7 – Busca de hiper-parâmetros: Relatório de classificação modelo Dense + BERT – Subáreas.

Classe	**Precisão**(%)	**Recall**(%)	**F1-score**(%)	**Número de exemplos**
Não IA	95	97	96	15391
IA	88	77	82	3685

5.1.3 Testes

A última etapa de desenvolvimento do modelo foi a aplicação no conjunto de testes. Interessantemente, as métricas se mantiveram muito parecidas com o que foi observado no conjunto de validação. Como o conjunto de testes é do mesmo domínio do treino e foi extraído com estratificação, mantendo a mesma distribuição, idealmente

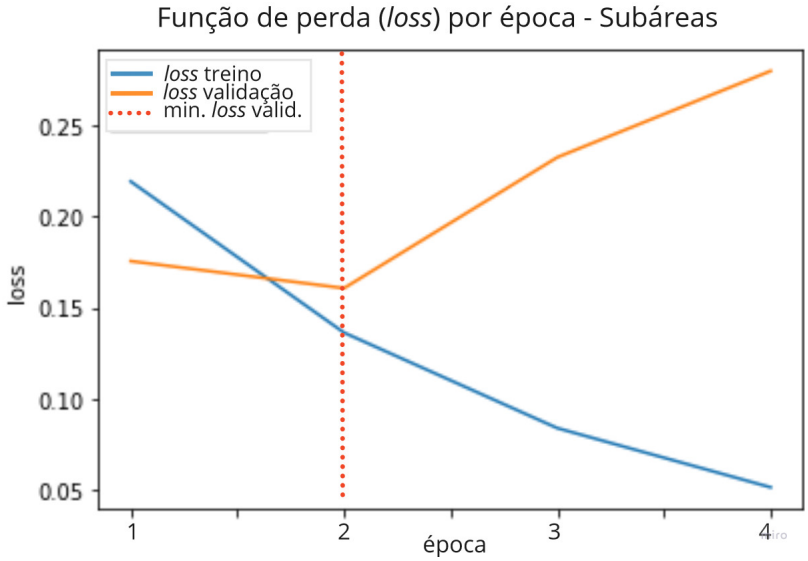


Figura 9 – *Loss* no treino Dense + BERT – Subáreas: No gráfico da figura é possível observar que a *loss* mais baixa na validação foi obtida na segunda época (linha pontilhada vermelha).

o modelo deve ter esse tipo de comportamento. Assim, a acurácia final obtida foi de 93% e f1-score macro de 89%. Para avaliar mais precisamente a acurácia do modelo, nesse teste, a acurácia balanceada foi calculada e seu valor foi de 87%.

Os resultados do modelo foram equilibrados, sendo a métrica menos robusta o *recall* para a classe IA. Ainda assim, o modelo foi capaz de classificar corretamente a maioria dos exemplos sem prejudicar significativamente as métricas da classe minoritária. Também é esperada alguma taxa de erro associada, por conta da forma como o conjunto de dados foi construído.

Tabela 8 – Teste: Relatório de classificação modelo Dense + BERT – Subáreas.

Classe	**Precisão**(%)	**Recall**(%)	**F1-score**(%)	**Exemplos**
Não IA	94	97	96	19238
IA	87	76	81	4606
Acurácia Balanceada(%):	-	-	89	23844

Como trabalhos que envolvem inteligência artificial não são exclusivamente publicados nos veículos associados à IA no conjunto de dados, pode haver títulos que tratam sobre IA nos exemplos de Não IA. Isso tende a fazer o modelo classificar um falso positivo, mas que não é necessariamente um erro de classificação em termos práticos; *e.g.* o título "*Deep Reinforcement Learning for Building HVAC Control*" foi classificado como IA, o que faz sentido já que o título é sobre o assunto, mas como esse trabalho foi publicado na "*Design Automation Conference* (DAC)" o título foi considerado Não IA na construção do conjunto de dados. A situação inversa também pode acontecer, porque existem trabalhos sobre outras subáreas da computação que utilizam IA e podem ser publicados nos veículos de publicação associados à IA.

Além dos resultados do relatório de classificação também foi criado um recurso

visual com uma matriz de confusão. Esses dados de avaliação estão ilustrados na Figura 10, que permite uma melhor visualização da distribuições das classificações.

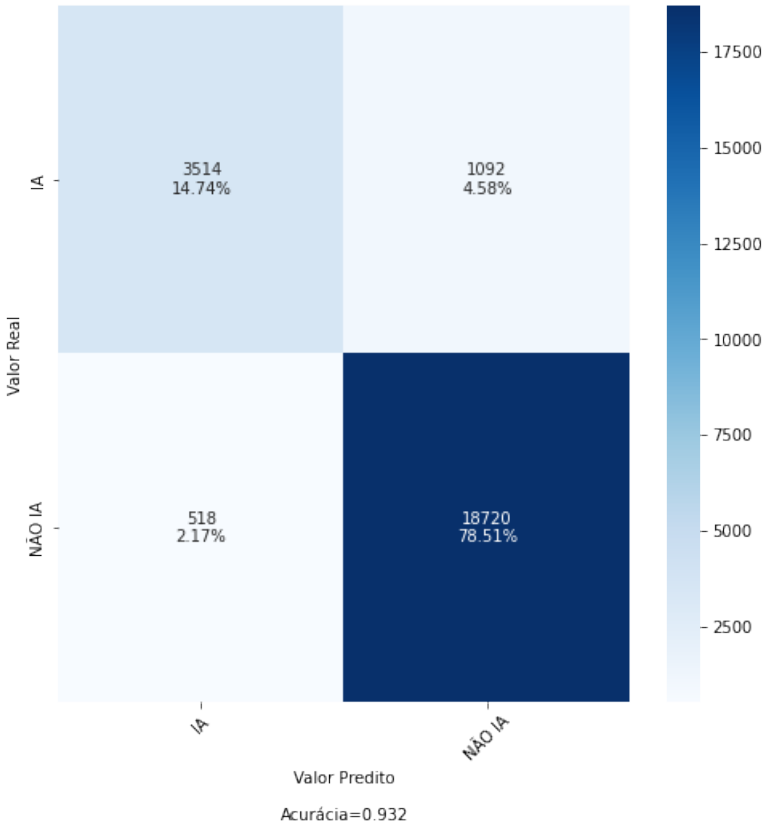


Figura 10 – Matriz de confusão Modelo Subáreas.

5.2 Modelo de Especialidades

A metodologia de desenvolvimento e organização dos resultados do Modelo Especialidades foi semelhante ao modelo anterior. Houveram apenas algumas diferenças nas etapas de busca de hiper-parâmetros.

5.2.1 Validação inicial

Na validação inicial, assim como na do modelo de subáreas, foram testadas as combinações apresentadas na Figura 7 sem o refinamento dos modelos. Nessa etapa, os melhores resultados foram obtidos pelas combinações SVC e TF, Dense e BERT. Os resultados estão dispostos nas Tabela 9 e Tabela 10, respectivamente.

O modelo SVC teve f1-score macro de 65% e acurácia de 72% , enquanto o Dense obteve 65% e 73% respectivamente.

5.2.2 Busca de hiper-parâmetros

A busca de hiper-parâmetros foi aplicada apenas nas duas melhores combinações da etapa anterior. No caso do modelo SVC, foram testados valores para *loss*, *dual*,

Tabela 9 – Validação inicial: Relatório de classificação modelo SVC + TF – Especialidades.

Classe	Precisão (%)	Recall (%)	F1-score (%)	Número de exemplos
computer vision	69	71	70	1198
knowledge representation & reasoning	50	51	51	1380
machine learning	63	64	63	1430
multiagent systems	79	80	80	1367
natural language processing	74	71	73	979
planning	64	54	59	175
robotics and perception	33	24	28	92

Tabela 10 – Validação Inicial: Relatório de classificação modelo Dense + BERT – Especialidades.

Classe	Precisão (%)	Recall (%)	F1-score (%)	Número de exemplos
computer vision	76	79	78	1198
knowledge representation & reasoning	58	56	57	1380
machine learning	72	64	68	1430
multiagent systems	80	84	82	1367
natural language processing	75	85	80	979
planning	61	65	63	175
robotics and perception	50	26	34	92

fit_intercept e *multi_class*, como descrito na metodologia. A melhor configuração encontrada foi utilizando os valores padrões da biblioteca, exceto para a *loss*, a qual performou melhor com o valor *hinge*, enquanto a função padrão é a *squared_hinge*. Os resultados deste modelo após o refinamento, estão dispostos na Tabela 11.

Já para o *pipeline* BERT, a única modificação testada foi a alteração do número de épocas. Assim como no Modelo Subáreas, durante o treinamento, as melhores métricas foram observadas com treino por duas épocas (Figura 11 e Figura 12). Por isso, o modelo final foi treinado por duas épocas. Os resultados das métricas avaliadas estão na Tabela 12.

Tabela 11 – Busca de hiper-parâmetros: Relatório de classificação modelo SVC + TF – Especialidades.

Classe	Precisão (%)	Recall (%)	F1-score (%)	Número de exemplos
computer vision	74	80	77	1198
knowledge representation & reasoning	65	47	54	1380
machine learning	65	73	69	1430
multiagent systems	80	86	83	1367
natural language processing	75	84	79	979
planning	66	55	60	175
robotics and perception	59	11	18	92

Após a busca de parâmetros o modelo SVC atingiu f1-score macro de 63% e acurácia de 72%. O modelo BERT, por sua vez, alcançou 66% e 72%, respectivamente. Portanto o último foi definido como Modelo Especialidades.

5.2.3 Teste

A última etapa foi a aplicação do Modelo Especialidades no conjunto de testes. A acurácia e f1-score macro atingidos foram 71% e 65%, respectivamente. Os resulta-

Tabela 12 – Busca de hiper-parâmetros: Relatório de classificação modelo Dense + BERT – Especialidades.

Classe	Precisão (%)	Recall (%)	F1-score (%)	Número de exemplos
computer vision	85	70	76	1198
knowledge representation & reasoning	63	48	54	1380
machine learning	61	82	70	1430
multiagent systems	83	83	83	1367
natural language processing	77	85	80	979
planning	62	67	64	175
robotics and perception	49	28	36	92

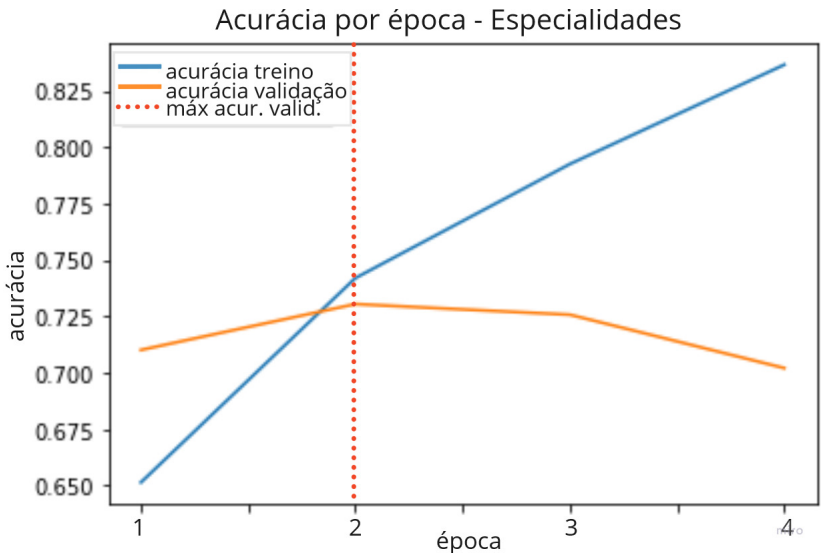


Figura 11 – Acurácia no treino Dense + BERT – Especialidades: No gráfico da figura é possível observar que a acurácia mais alta na validação foi obtida na segunda época (linha pontilhada vermelha).

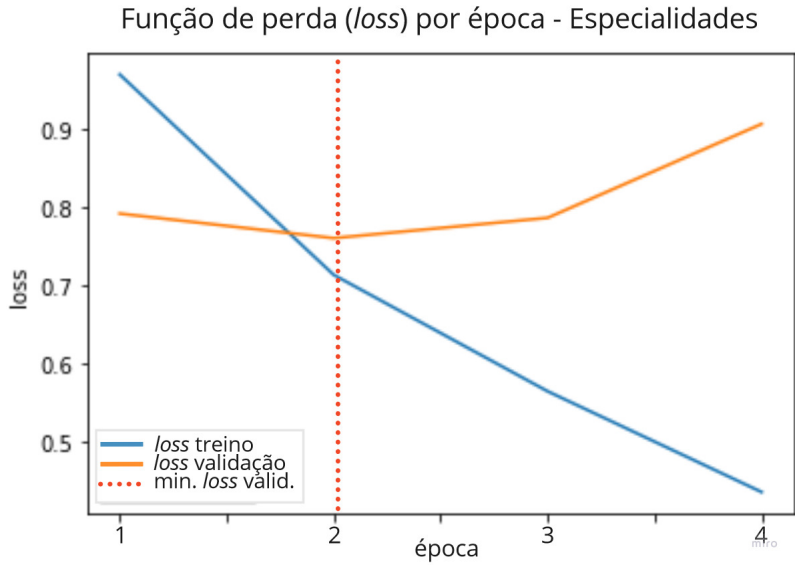


Figura 12 – *Loss* no treino Dense + BERT – Especialidades: No gráfico da figura é possível observar que a *loss* mais baixa na validação foi obtida na segunda época (linha pontilhada vermelha).

dos completos estão apresentados na Tabela 13. Nesse teste, também foi calculada a acurácia balanceada para se obter uma avaliação mais precisa, o valor obtido para essa métrica foi de 66%. O resultado foi abaixo da acurácia padrão, porém ainda significativamente melhor que um classificador aleatório. Na tabela, é possível observar as classes com menos exemplos no conjunto de dados tiveram f1-score abaixo da média. Uma hipótese de possível forma de melhorar esses resultados é realizar o treino com mais exemplos de títulos em *planning* e *robotics and perception*.

Tabela 13 – Teste: Relatório de classificação modelo Dense + BERT – Especialidades.

Classe	Precisão (%)	Recall (%)	F1-score (%)	Número de exemplos
computer vision	81	67	73	1498
knowledge representation & reasoning	61	47	53	1725
machine learning	59	79	68	1788
multiagent systems	83	83	83	1709
natural language processing	78	85	82	1224
planning	58	62	60	218
robotics and perception	51	32	40	115
Acurácia Balanceada(%):	-	-	67	8227

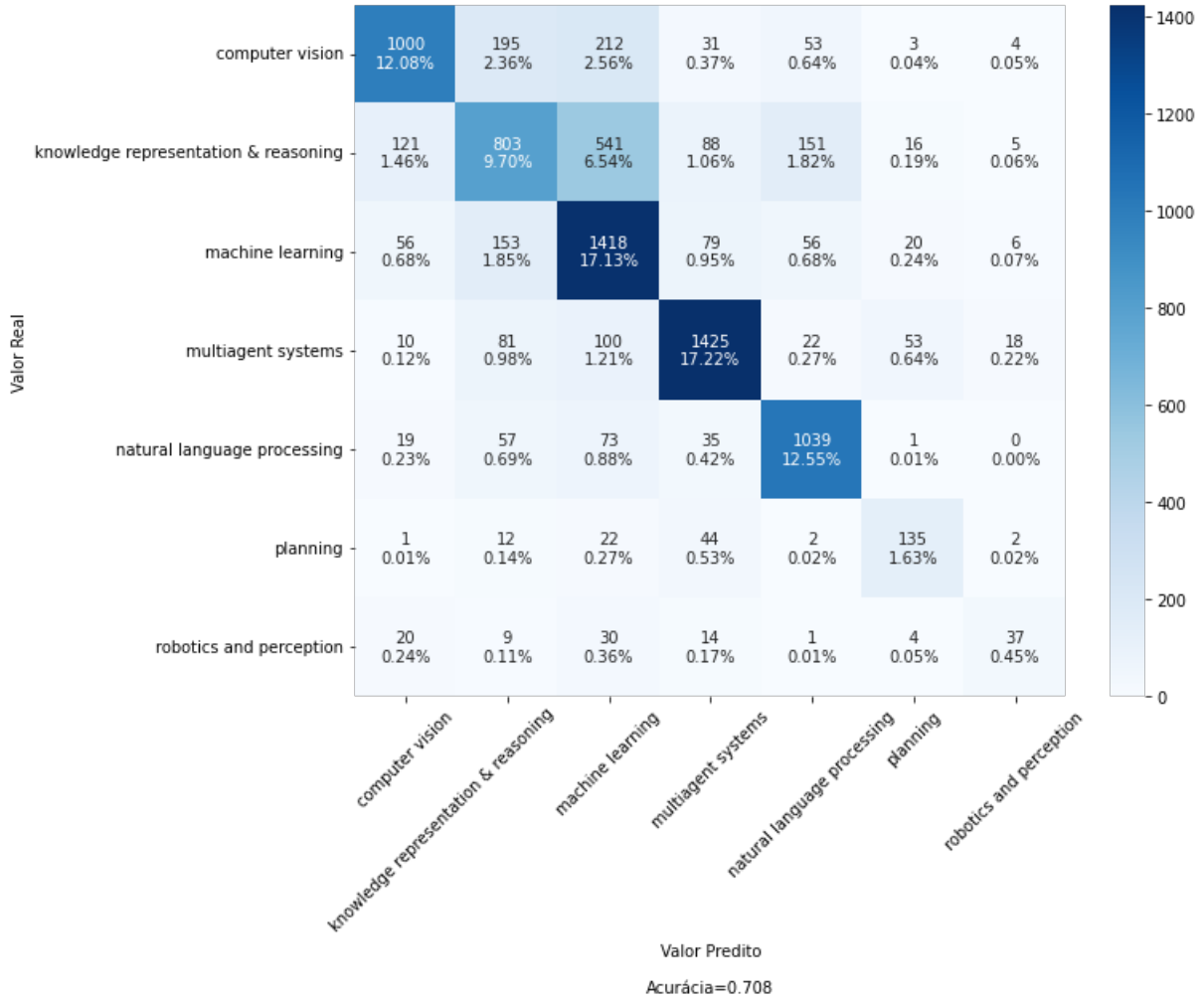


Figura 13 – Matriz de confusão Modelo Especialidades.

A distribuição das classificações também pode ser observada na Figura 13. A

imagem ajuda a entender quais as classes o modelo "confundiu". Além das classes minoritárias, a classe *knowledge representation & reasoning*, apesar de ser a segunda com mais exemplos, também teve o f1-score abaixo da média. No gráfico, pode-se notar que a classe *machine learning* concentra a maioria dos falsos negativos em quase todas as classes. Isso significa que quando o modelo erra a classe, ele tende a classificar erroneamente como *machine learning*.

Tendo em mente que *machine learning* não é um assunto exclusivamente publicado nos veículos associados à especialidade no *dataset*, podemos entender que parte dos erros podem não ser erros de fato. Sendo assim, uma parcela dos erros pode ser atribuída à construção do conjunto de dados, assim como discutido no caso do modelo Subáreas.

5.2.3.1 Análise machine learning

Como os exemplos da especialidade *knowledge representation & reasoning* apresentaram um volume mais alto de classificações em *machine learning* do que as demais, foi realizada uma análise específica para esse caso. A análise consistiu em fazer a contagem da ocorrência de termos que estão relacionados à *machine learning* nos exemplos de *knowledge representation & reasoning* classificados como *machine learning*.

Para escolher os termos a serem contados, foi feita a contagem dos termos mais frequentes nos títulos dos exemplos de *machine learing* desconsiderando *stop words*. Além disso, foi feita uma exploração manual do conjunto dos 541 títulos que foram classificados incorretamente. Assim, foi criada a lista de termos apresentadas na Tabela 14. Na tabela também é possível conferir a contagem de cada termo.

Tabela 14 – Ocorrência de termos selecionados nos exemplos da classe *knowledge representation & reasoning* classificados como *machine learning*.

Termo	**Ocorrências**
adversarial learning	1
adversarial net	4
adversarial train	3
deep learning	17
machine learning	7
meta-learning	10
neural net	45
reinforcement learning	23
supervised learning	8
transfer learning	5
TOTAL:	123

Com essa análise é possível entender que existem artigos de *machine learning* que foram atribuídos à classe *knowledge representation & reasoning* na construção

do *data set*. Além disso, é provável que exista uma interseção entre essas duas especialidades e, como *machine learning* tem diversas aplicações e está presente em diferentes especialidades, é provável que este fenômeno aconteça nas demais especialidades. Dessa forma, é possível entender o porquê de alguns erros. Naturalmente, o modelo não é perfeito e também tem erros "reais", entretanto é importante levar em consideração os pontos apresentados.

5.3 Teste com dados da plataforma Sucupira

Os modelos propostos foram desenvolvidos com o objetivo de classificar corretamente a subárea/especialidade dos títulos de artigos científicos. Contudo, em aprendizado de máquina pode existir um problema de generalização incorreta do objetivo (DI LANGOSCO et al., 2022). Um dos fatores que podem ocasionar esse tipo de problema é uma distribuição de dados de treino que condiciona o modelo a aprender a tarefa errada. Por isso, como uma tentativa de averiguar se os modelos aprenderam o objetivo correto, foram realizados testes com uma distribuição de dados diferente do observado no conjunto de treino.

Para isso, foi gerado um novo conjunto de dados com títulos de artigos extraídos da plataforma Sucupira. A Plataforma Sucupira é um sistema desenvolvido e mantido pela CAPES que tem como objetivo coletar, analisar e avaliar informações sobre os programas de pós-graduação no Brasil. Na plataforma é possível consultar dados de produções intelectuais de programas de pós-graduação de universidades brasileiras. Dessa forma, existe como obter-se títulos de artigos publicados em determinadas áreas utilizando filtros de busca.

Com o uso desta ferramenta, foram selecionados 478 títulos de artigos publicados em periódicos no ano de 2021 em programas de Computação. Esses títulos são de trabalhos publicados pelas seguintes universidades: UFRGS, UFPE, UFMG e UNICAMP. A escolha do ano de publicação e universidades a serem filtradas foi feita com base nos filtros com maior número de registros retornados. Além disso, o volume de dados mencionado foi atingido após a exclusão dos títulos escritos em línguas diferentes da inglesa.

Com os dados extraídos da plataforma, foi feito um rotulamento manual dos títulos para construir um conjunto de dados compatível com o Modelo Subáreas. Este rotulamento foi executado pelo autor do presente trabalho. Dessa maneira, os títulos foram classificados em "IA" ou "Não IA" estritamente através da leitura dos mesmos.

Após a definição do conjunto de dados, o Modelo Subáreas foi aplicado aos títulos e teve seus resultados comparados às classificações manuais. Nesse teste, o modelo obteve 79% de acurácia, 76% de acurácia balanceada e 77% de f1-score. Após a aplicação do Modelo Subáreas, foram selecionados apenas os títulos classi-

ficados corretamente como IA (122 exemplos). Então, esses títulos em IA também foram classificados manualmente de acordo com suas especialidades e, dessa forma, foi construído um conjunto de dados compatível com o Modelo Especialidades. Isso permitiu aplicar o Modelo Especialidades para classificar os títulos. Assim, este atingiu acurácia de 61% e f1-score de 44% na tarefa de classificação. Contudo, nesse teste a acurácia balanceada foi de 49%, algo semelhante a uma classificação aleatória. Porém, é importante notar que existem três classes com menos de 5 exemplos, um número muito baixo para avaliar adequadamente o modelo. Nesse sentido, caso a acurácia balanceada fosse calculada excluindo as classes em questão, o valor obtido seria de 61%. A matriz de confusão dos resultados dos modelos estão dispostas na Figuras 14 e Figura 15, respectivamente.

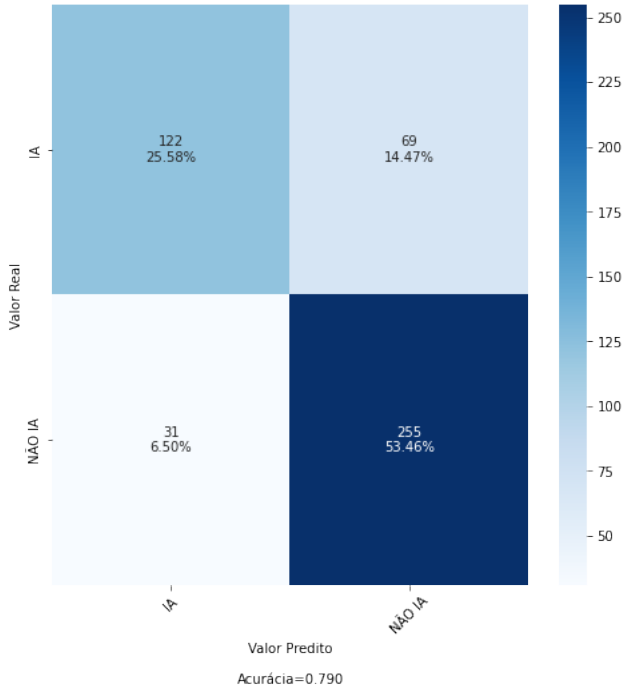


Figura 14 – Matriz de confusão Modelo Subáreas - Sucupira.

Ao comparar os resultados da classificação do novo conjunto de dados com a do conjunto de testes, pode-se notar um desempenho inferior de ambos os modelos na última. Contudo, o Modelo Subáreas manteve resultados razoáveis de acurácia e f1-score. Já o Modelo Especialidades apresentou resultados piores e o valor de f1-score não ficou próximo da acurácia, o que significa que o equilíbrio das taxas de precisão e *recall* não acompanhou a taxa de acerto.

O desempenho inferior do modelo Especialidades pode ser parcialmente explicado por alguns fatores. Algo que comprometeu o resultado foi a pequena quantidade de exemplos de algumas classes, o que afeta diretamente as métricas de f1-score e sua média macro entre as classes. Este fator fica explícito ao calcularmos a acurácia balanceada desconsiderando as classes com poucos exemplos. Além disso, durante

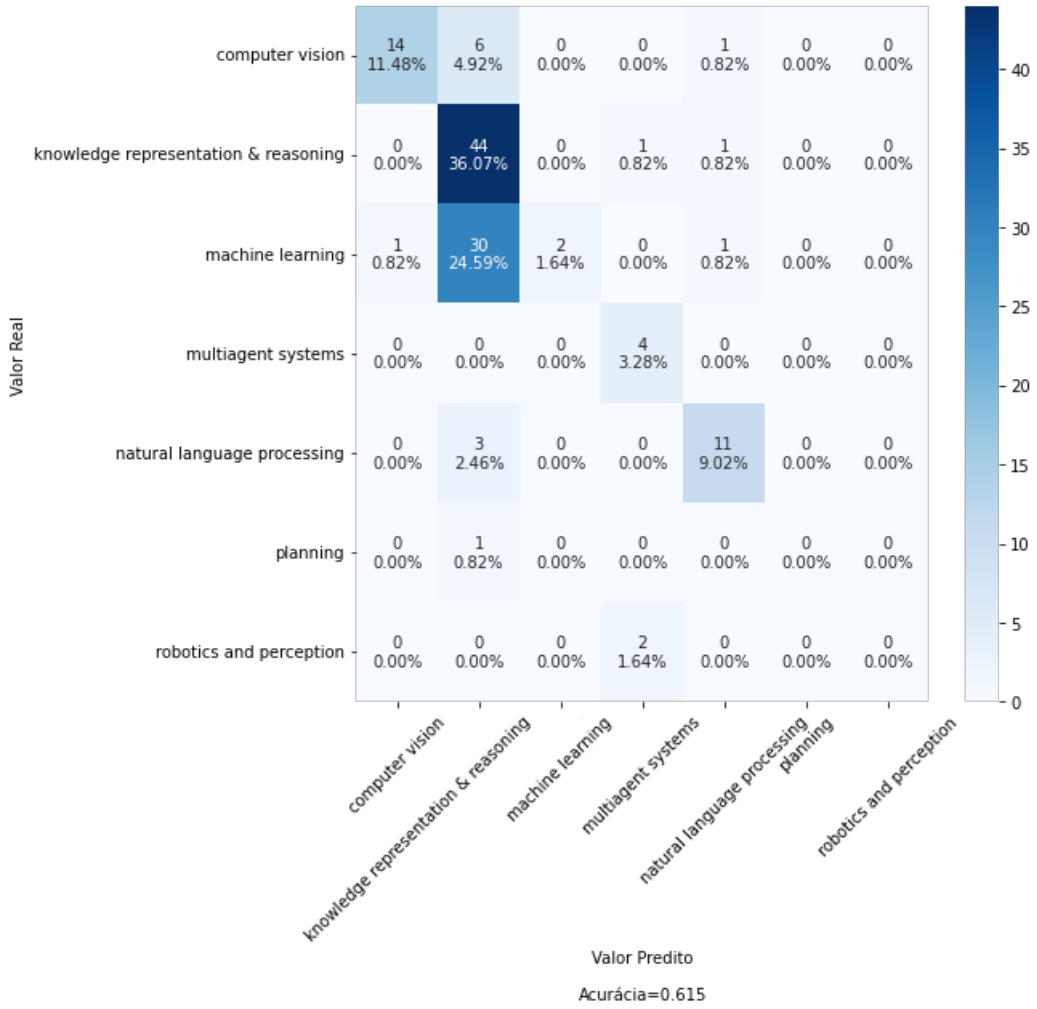


Figura 15 – Matriz de confusão Modelo Especialidades - Sucupira.

a classificação manual e análise dos resultados, foi percebido que o modelo deveria considerar mais classes. Esta situação foi evidenciada pela dificuldade de atribuir uma das sete classes a certos exemplos, quando, na verdade, o ideal seria a classificação em outras classes inexistentes (*e.g.* computação evolutiva, lógica fuzzy). Nesse tipo de situação, a maioria dos exemplos acabou sendo classificada manualmente como *Machine Learning* ou *Knowledge representation & Reasoning*.

Uma circunstância que chamou a atenção em relação às classificações do modelo Subáreas foi que o modelo não classificou corretamente alguns exemplos mesmo com a presença de termos importantes. Isso foi observado, principalmente, nos exemplos que deveriam ter sido classificados como *Machine Learning*. Diversos exemplos com termos como "Deep learning", "Neural Networks" e "Train" foram classificados erroneamente como *Knowledge Representation & Reasoning*.

Nesse contexto, é importante destacar que a classificação manual do pesquisador também está sujeita a erros e foi utilizada como uma referência aproximada. Além disso, a própria definição das categorias pode ser algo subjetivo e discutível. Ademais, uma característica do problema que dificulta a avaliação do desempenho real destes modelos é que, em muitos casos, um título pode estar relacionado a mais de uma classe. Sendo assim, o problema também poderia ser estudado sob o ponto de vista de uma classificação multi-rótulo.

6 CONCLUSÃO

Para realização deste trabalho foram coletados, do repositório DBLP, dados bibliográficos de produções científicas na área da Ciência da Computação e, com base nestes, foram desenvolvidos modelos de classificação automática para a subárea da IA e suas especialidades. O único recurso utilizado pelos modelos desenvolvidos foi o título das publicações. Nos testes realizados, o melhor desempenho de f1-score e acurácia foi de um modelo de rede neural artificial (Dense) treinado com o modelo BERT em ajuste fino. Este modelo foi capaz de classificar corretamente 93% dos títulos entre IA e Não IA. Um modelo semelhante foi desenvolvido para classificar a especialidade dos títulos e obteve acurácia de 71% na classificação contendo sete classes possíveis. Os resultados obtidos são mais uma evidência da eficiência do modelo BERT e sua capacidade de ajuste fino em domínios específicos.

Como o dado de entrada dos modelos propostos é essencialmente texto, estes podem ser aplicados a qualquer base de dados de títulos de publicações, com a ressalva de que devem ser considerados os recortes de subárea e áreas em que os modelos foram treinados. Dessa maneira, esses modelos podem contribuir em estudos que analisam dados do Lattes, servindo como classificadores dos títulos contidos na plataforma. Os testes realizados com os dados da plataforma Sucupira servem como um exemplo para esse tipo de estudo e também explicita algumas limitações atuais dos modelos propostos, como a falta de algumas especialidades e a tendência do Modelo Especialidades de classificar falsos positivos para a classe *Knowledge Representation & Reasoning*.

Adicionalmente, a própria construção dos conjuntos de dados é também uma contribuição, pois a metodologia de aquisição de dados pode ser replicada, bem como a base de dados construída pode servir de referência para outros estudos.

Como temas de trabalhos futuros podem ser sugeridos: inclusão de novas classes e classificação multi-rótulos, novas aplicações e testes dos modelos desenvolvidos em comparação com classificação manual humana, expansão da metodologia para outras áreas do conhecimento, integração com ferramentas de extração de dados do Lattes e mapeamento da produção científica em IA no Brasil utilizando os modelos propostos.

REFERÊNCIAS

ABADI, M. et al. **TensorFlow**: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org. Disponível em: <https://www.tensorflow.org/>.

ALWAHAISHI, S.; MARTINOVIČ, J.; SNÁŠEL, V. Analysis of the DBLP Publication Classification Using Concept Lattices. In: DIGITAL ENTERPRISE AND INFORMATION SYSTEMS, 2011, Berlin, Heidelberg. **Anais...** Springer Berlin Heidelberg, 2011. p.99–108.

BIRYUKOV, M.; DONG, C. Analysis of Computer Science Communities Based on DBLP. In: RESEARCH AND ADVANCED TECHNOLOGY FOR DIGITAL LIBRARIES, 2010, Berlin, Heidelberg. **Anais...** Springer Berlin Heidelberg, 2010. p.228–235.

BREIMAN, L. Random forests. **Machine Learning**, Países Baixos, v.45, p.5–32, 10 2001.

BRITO, A. G. C. de; QUONIAM, L.; MENA-CHALCO, J. P. Exploração da Plataforma Lattes por assunto: proposta de metodologia. **Transinformação**, Campinas, SP, v.28, p.77–86, 2016.

CHAKRABORTY, T. et al. Computer science fields as ground-truth communities: Their impact, rise and fall. In: SPRINGER BERLIN HEIDELBERG, 2013, Niagra, Ontario. **Anais...** Association for Computing Machinery, 2013. p.426–433.

CHEN, T.; GUESTRIN, C. XGBoost: A scalable tree boosting system. In: ASSOCIATION FOR COMPUTING MACHINERY, 2016, San Francisco, CA. **Anais...** Association for Computing Machinery, 2016. p.785–794.

CHOLLET, F. et al. **Keras**. Disponível em: <https://keras.io>.

CORTES, C.; VAPNIK, V.; SAITTA, L. Support-vector networks. **Machine Learning**, Boston, v.20, p.273–297, 9 1995.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. **BERT**: Pre-training of Deep Bidirectional Transformers for Language Understanding. [S.l.]: arXiv, 2018. Disponível em: <https://arxiv.org/abs/1810.04805>.

DI LANGOSCO, L. L. et al. Goal Misgeneralization in Deep Reinforcement Learning. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 39., 2022, Baltimore. **Proceedings...** PMLR, 2022. p.12004–12019. (Proceedings of Machine Learning Research, v.162).

DIAS, T. M. R.; MOITA, G. F. Um retrato da produção científica brasileira baseado em dados da plataforma lattes. **Brazilian Journal of Information Science: research trends**, Marília, SP, v.12, n.4, p.62–74, 2018.

DIAS, T. M. R.; MOITA, G. F.; DIAS, P. M. Adoption of the lattes platform as the data source for the characterization of scientific networks. **Encontros Bibli: revista eletrônica de biblioteconomia e ciência da informação; v. 21, n. 47 (2016); 16-26**, [S.l.], v.24, n.2, p.26–16, 2016.

DIGIAMPIETRI, L. et al. Minerando e Caracterizando Dados de Currículos Lattes. In: I BRAZILIAN WORKSHOP ON SOCIAL NETWORK ANALYSIS AND MINING, 2012, Porto Alegre, RS, Brasil. **Anais...** SBC, 2012.

FERREIRA, A. A.; GONÇALVES, M. A.; LAENDER, A. H. A brief survey of automatic methods for author name disambiguation. **Acm Sigmod Record**, [S.l.], v.41, n.2, p.15–26, 2012.

HINTON, G. E. et al. **Improving neural networks by preventing co-adaptation of feature detectors**. [S.l.]: arXiv, 2012. Disponível em: <https://arxiv.org/abs/1207.0580>.

HOSSIN, M.; SULAIMAN, M. N. A review on evaluation metrics for data classification evaluations. **International journal of data mining & knowledge management process**, [S.l.], v.5, n.2, p.1, 2015.

HUSSAIN, I.; ASGHAR, S. A survey of author name disambiguation techniques: 2010–2016. **The Knowledge Engineering Review**, Cambridge, v.32, p.e22, 2017.

JOORABCHI, A.; MAHDI, A. E. An unsupervised approach to automatic classification of scientific literature utilizing bibliographic metadata. **Journal of Information Science**, [S.l.], v.37, n.5, p.499–514, 2011.

KADHIM, A. I. Survey on supervised machine learning techniques for automatic text classification. **Artificial Intelligence Review**, [S.l.], v.52, n.1, p.273–292, 2019.

KIM, J. Evaluating author name disambiguation for digital libraries: a case of DBLP. **Scientometrics**, Budapeste, v.116, n.3, p.1867–1886, 2018.

KINGMA, D. P.; BA, J. **Adam**: A Method for Stochastic Optimization. [S.l.]: arXiv, 2014. Disponível em: <https://arxiv.org/abs/1412.6980>.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. **Communications of the ACM**, [S.l.], v.60, n.6, p.84–90, 2017.

LI, Q. et al. **A Survey on Text Classification**: From Shallow to Deep Learning. [S.l.]: arXiv, 2020. Disponível em: <https://arxiv.org/abs/2008.00364>.

PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, [S.l.], v.12, p.2825–2830, 2011.

SANYAL, D. K.; BHOWMICK, P. K.; DAS, P. P. A review of author name disambiguation techniques for the PubMed bibliographic database. **Journal of Information Science**, [S.l.], v.47, n.2, p.227–254, 2021.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. **Information Processing Management**, [S.l.], v.45, n.4, p.427–437, 2009.

ZAIANE, O. R.; CHEN, J.; GOEBEL, R. Dbconnect: mining research community on dblp data. In: WEBKDD AND 1ST SNA-KDD 2007 WORKSHOP ON WEB MINING AND SOCIAL NETWORK ANALYSIS, 9., 2007, San Jose, Califórnia. **Proceedings...** [S.l.: s.n.], 2007. p.74–81.

Apêndices

APÊNDICE A – Veículos de publicação

Tabela 15 – Veículos de publicação selecionados para as produções fora da subárea da IA.

veículo de Publicação	Subárea
ACM Transactions on Graphics	Computer Graphics
IEEE Transactions on Visualization and Computer Graphics	Computer Graphics
Computer Graphics Forum	Computer Graphics
Computers & Graphics	Computer Graphics
IEEE Computer Graphics and Applications	Computer Graphics
The Visual Computer	Computer Graphics
IEEE Transactions on Computers	Computer Hardware Design
IEEE Journal of Solid-State Circuits	Computer Hardware Design
IEEE International Solid-State Circuits Conference	Computer Hardware Design
IEEE Transactions on Circuits and Systems I: Regular Papers	Computer Hardware Design
Design Automation Conference (DAC)	Computer Hardware Design
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems	Computer Hardware Design
IEEE Communications Surveys & Tutorials	Computer Networks & Wireless Communication
IEEE Communications Magazine	Computer Networks & Wireless Communication
IEEE Transactions on Wireless Communications	Computer Networks & Wireless Communication
IEEE Transactions on Vehicular Technology	Computer Networks & Wireless Communication
Journal of Network and Computer Applications	Computer Networks & Wireless Communication
ACM Symposium on Theory of Computing	Theoretical Computer Science
IEEE Symposium on Foundations of Computer Science (FOCS)	Theoretical Computer Science
ACM SIAM Symposium on Discrete Algorithms	Theoretical Computer Science
SIAM Journal on Computing	Theoretical Computer Science
Journal of the ACM (JACM)	Theoretical Computer Science
Theoretical Computer Science	Theoretical Computer Science
ACM/IEEE International Conference on Human Robot Interaction	Human Computer Interaction
IEEE Transactions on Affective Computing	Human Computer Interaction
ACM Symposium on User Interface Software and Technology	Human Computer Interaction
International Journal of Human-Computer Studies	Human Computer Interaction
IEEE Transactions on Human-Machine Systems	Human Computer Interaction
Microelectronics Reliability	Microelectronics & Electronic Packaging
Symposium on VLSI Circuits	Microelectronics & Electronic Packaging
Microelectronics Journal	Microelectronics & Electronic Packaging
IEEE International Reliability Physics Symposium (IRPS)	Microelectronics & Electronic Packaging

Tabela 16 – Veículos de publicação selecionados para as especialidades da IA.

veículo de Publicação	Especialidade
International Conference on Machine Learning	Machine learning
Journal of Machine Learning Research	Machine learning
International Natural Language Generation Conference	Natural language processing
Annual Meeting of the Association for Computational Linguistics	Natural language processing
International Conference on Automated Planning and Scheduling	Planning
Principles of Knowledge Representation and Reasoning	Knowledge representation & reasoning
Knowledge-Based Systems	Knowledge representation & reasoning
International Conference on Learning Representations	Knowledge representation & reasoning
Conference on Robot Learning	Robotics and perception
Cognitive Intelligence and Robotics	Robotics and perception
International Conference on Autonomous Agents and Multiagent Systems	Multiagent systems
Practical Applications of Agents and Multi-Agent Systems	Multiagent systems
Autonomous Agents and Multi-Agent Systems	Multiagent systems
IEEE Conference on Computer Vision and Pattern Recognition	Computer vision
International Conferences on Artificial Intelligence and Computer Vision	Computer vision

APÊNDICE B – Mapa de buscas na API do DBLP

Tabela 17 – Nome dos veículos de publicação e nome de busca na API DBLP.

Nome do veículo de publicação	Nome encontrado na API do DBLP
ACM Transactions on Graphics	ACM Trans. Graph.
IEEE Transactions on Visualization and Computer Graphics	IEEE Trans. Vis. Comput. Graph.
Computer Graphics Forum	Comput. Graph. Forum
Computers & Graphics	Comput. Graph.
IEEE Computer Graphics and Applications	IEEE Computer Graphics and Applications
The Visual Computer	Vis. Comput.
IEEE Transactions on Computers	IEEE Trans. Computers
IEEE Journal of Solid-State Circuits	IEEE J. Solid State Circuits
IEEE International Solid-State Circuits Conference	ISSCC
IEEE Transactions on Circuits and Systems I: Regular Papers	IEEE Trans. Circuits Syst. I Regul. Pap.
Design Automation Conference (DAC)	DAC
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems	IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.
IEEE Communications Surveys & Tutorials	IEEE Commun. Surv. Tutorials
IEEE Communications Magazine	IEEE Commun. Mag.
IEEE Transactions on Wireless Communications	IEEE Trans. Wirel. Commun.
IEEE Transactions on Vehicular Technology	IEEE Trans. Veh. Technol.
Journal of Network and Computer Applications	J. Netw. Comput. Appl.
ACM Symposium on Theory of Computing	STOC
IEEE Symposium on Foundations of Computer Science (FOCS)	FOCS
ACM SIAM Symposium on Discrete Algorithms	SODA
SIAM Journal on Computing	SIAM J. Comput.
Journal of the ACM (JACM)	J. ACM
Theoretical Computer Science	Theor. Comput. Sci.
ACM/IEEE International Conference on Human Robot Interaction	HRI
IEEE Transactions on Affective Computing	IEEE Trans. Affect. Comput.
ACM Symposium on User Interface Software and Technology	UIST
International Journal of Human-Computer Studies	Int. J. Hum. Comput. Stud.
IEEE Transactions on Human-Machine Systems	IEEE Trans. Hum. Mach. Syst.
Microelectronics Reliability	Microelectron. Reliab.
Symposium on VLSI Circuits	VLSI Circuits
Microelectronics Journal	Microelectron. J.
IEEE International Reliability Physics Symposium (IRPS)	IRPS
AAAI Conference on Artificial Intelligence	AAAI
International Joint Conferences on Artificial Intelligence	IJCAI
Artificial Intelligence - Journal	Artif. Intell.
The Florida AI Research Society	FLAIRS Conference
European Conference on Artificial Intelligence	ECAI
International Journal on Artificial Intelligence Tools	Int. J. Artif. Intell. Tools
Brazilian Symposium on Artificial Intelligence	SBIA
Brazilian Conference on Intelligent Systems	BRACIS
International Conference on Machine Learning	ICML
Journal of Machine Learning Research	J. Mach. Learn. Res.
International Natural Language Generation Conference	INLG
Annual Meeting of the Association for Computational Linguistics	ACL
International Conference on Automated Planning and Scheduling	ICAPS
Principles of Knowledge Representation and Reasoning	KR
Knowledge-Based Systems	Knowl. Based Syst.
International Conference on Learning Representations	ICLR
Conference on Robot Learning	CoRL
Cognitive Intelligence and Robotics	Cognitive Intelligence and Robotics
International Conference on Autonomous Agents and Multiagent Systems	AAMAS
Practical Applications of Agents and Multi-Agent Systems	PAAMS
Autonomous Agents and Multi-Agent Systems	Auton. Agents Multi Agent Syst.
IEEE Conference on Computer Vision and Pattern Recognition	CVPR
International Conferences on Artificial Intelligence and Computer Vision	AICV