UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Lucas Roges de Araujo

CLASSIFICAÇÃO AUTOMÁTICA DE QUESTÕES DE PROVAS: ANÁLISE COMPARATIVA DE ALGORITMOS E APLICAÇÃO AO ENADE

Santa Maria, RS
2021

Lucas Roges de Araujo

CLASSIFICAÇÃO AUTOMÁTICA DE QUESTÕES DE PROVAS: ANÁLISE COMPARATIVA DE ALGORITMOS E APLICAÇÃO AO ENADE

Trabalho Final de Graduação apresentado ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Bacharel em Ciência da Computação**.

ORIENTADORA: Prof.ª Andrea Schwertner Charão

TG N. 484
Santa Maria, RS
2021

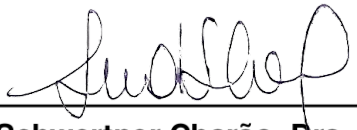
End. Eletr.: lraraujo@inf.ufsm.br

Lucas Roges de Araujo

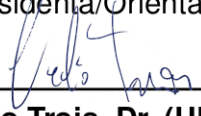
CLASSIFICAÇÃO AUTOMÁTICA DE QUESTÕES DE PROVAS: ANÁLISE COMPARATIVA DE ALGORITMOS E APLICAÇÃO AO ENADE

Trabalho Final de Graduação apresentado ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Bacharel em Ciência da Computação**.

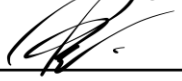
Aprovado em 12 de fevereiro de 2021:



Andrea Schwertner Charão, Dra. (UFSM)
(Presidenta/Orientadora)



Celio Trois, Dr. (UFSM)



Rodrigo da Silva Guerra, Dr. (UFSM)

Santa Maria, RS
2021

DEDICATÓRIA

Dedico este trabalho ao meu avô Alber (in memorian), por toda sua dedicação e comprometimento em me educar desde os anos iniciais da minha vida até parte da graduação. Seu esforço foi fundamental para que eu pudesse chegar até aqui.

AGRADECIMENTOS

Agradeço à minha orientadora por todas as oportunidades de aprendizado e todo o conhecimento que me proporcionou, durante boa parte da graduação, desde os trabalhos de pesquisa até o desenvolvimento deste trabalho.

Agradeço à minha família por todo apoio e incentivo durante o período da graduação, assim como aos amigos que estiveram comigo durante esse período.

The advance of technology is based on making it fit in so that you don't really even notice it, so it's part of everyday life.

(Bill Gates)

RESUMO

CLASSIFICAÇÃO AUTOMÁTICA DE QUESTÕES DE PROVAS: ANÁLISE COMPARATIVA DE ALGORITMOS E APLICAÇÃO AO ENADE

AUTOR: Lucas Roges de Araujo
ORIENTADORA: Andrea Schwertner Charão

Apesar da grande quantidade de dados textuais armazenada digitalmente, existem diversos desafios para processá-los, que diminuem seu aproveitamento. Com dificuldades ligadas ao processamento de dados não estruturados, os estudos evitam a exploração e extração de informação desses elementos e, muitas vezes, esses conjuntos de texto são deixados de lado, ficando o foco em outros dados pertencentes à mesma base de dados. Uma situação deste tipo ocorre com os dados da educação brasileira, onde os microdados, compostos pelas notas das provas e respostas de questionários, são avaliados com frequência. Por outro lado, há uma quantidade reduzida de trabalhos que associam essas variáveis dos microdados com os conteúdos que formam esse resultado, que são os dados textuais das provas. Apesar das dificuldades para extração e processamento desses dados, já existem alguns avanços nessa área. Esses avanços incluem a automatização da classificação de questões, o que torna mais viável analisar esses dados em conjunto. Considerando esses avanços e a escassez de exploração dos dados textuais fornecidos por órgãos responsáveis pelo sistema educacional brasileiro, o presente trabalho tem o objetivo de analisar técnicas de processamento e classificação de dados textuais e utilizá-las no contexto de provas do Enade (Exame Nacional de Desempenho de Estudantes). A partir da seleção de algoritmos que se prestam a essa finalidade, buscou-se avaliar seu desempenho e acurácia na classificação das questões, de acordo com a categorização definida.

Palavras-chave: Classificação de questões. Mineração de dados. Aprendizado de máquina. Enade.

ABSTRACT

AUTOMATIC CLASSIFICATION OF EXAMS QUESTIONS: A COMPARATIVE ANALYSIS OF ALGORITHMS AND APPLICATION TO ENADE

AUTHOR: Lucas Roges de Araujo
ADVISOR: Andrea Schwertner Charão

Despite the large amount of textual data stored digitally, there are several challenges to process it, which reduce its use. With difficulties attached to processing unstructured data, the studies avoid the exploration and extraction of information from these elements, and many times, they leave out these sets of texts, so the focus turns to other data belonging to the same database. This situation happens with the Brazilian educational data, where the microdata, composed of grades of exams and answers of questionnaires, is frequently evaluated. On the other hand, there is a minor quantity of studies that associates these microdata variables with the content that forms this result, which is the textual data from the exams. Despite the difficulties to extract and process this sort of data, there are some advances in this area. These advances include the automatization of question classification, which makes it practical to analyze these data together. Considering these advances and the lack of textual data exploration provided by the agencies responsible for the Brazilian educational system, this study aims to analyze techniques for processing and classifying textual data in the context of ENADE (Brazilian National Student Performance Exam). After selecting algorithms suitable for this purpose, we have evaluated their performance and accuracy, according to the chosen categorization.

Keywords: Question classification. Data mining. Machine learning. Enade.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CBOW	*Continuous Bag-of-Words*
CNN	*Convolutional Neural Network*
CRISP-DM	*CRoss-Industry Standard Process for Data Mining*
Enade	Exame Nacional de Desempenho dos Estudantes
Enem	Exame Nacional do Ensino Médio
INEP	Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira
KDD	*Knowledge Discovery in Databases*
KNN	*K-Nearest Neighbors*
LR	*Logistic Regression*
LSTM	*Long Short-Term Memory*
NB	*Naive Bayes*
NLTK	*Natural Language Toolkit*
OCR	*Optical Character Recognition*
PAIUB	Programa de Avaliação Institucional das Universidades Brasileiras
PLN	Processamento de Linguagem Natural
RF	*Random Forest*
RN	Redes Neurais
RNN	*Recurrent Neural Network*
SEMMA	*Sample, Explore, Modify, Model, Assess*
SVM	*Support Vector Machine*
t-SNE	*t-distributed Stochastic Neighbor Embedding*
TF-IDF	*Term Frequency Inverse Document Frequency*

SUMÁRIO

1 INTRODUÇÃO

O aumento da produção de dados, nos últimos tempos, é consequência do uso massivo de tecnologia nas atividades do cotidiano. A evolução da tecnologia permite que dados sejam produzidos periodicamente, de maneira direta, além de ser acompanhada de artifícios que permitem uma constante coleta de dados através dos dispositivos em uso pela população, de maneira indireta. Porém, o dado, por si só, traz pouco significado além da sua existência, visto que seu entendimento acontece à medida que o processamento é realizado e transformamos o dado em outros conceitos, como informação, conhecimento e sabedoria (BELLINGER; CASTRO; MILLS, 2004). É importante transformar e processar os dados ao longo dessa cadeia de conceitos, para expandir o conceito mínimo até algo mais amplo e útil, além de facilitar seu entendimento.

No segmento de processamento de dados, existem diversos desafios, principalmente ligados a dados não estruturados. Nesse contexto, apesar do fato de palavras e textos serem elementos ricos em estrutura, vide as classificações ligadas a morfologia, sintaxe e semântica da linguagem, na análise de dados esses elementos são considerados não estruturados. Dentre as soluções para tratar esse tipo de dado, existem abordagens com o objetivo de transformar os dados não estruturados em dados estruturados (TSUMURA et al., 2016), possibilitando que os algoritmos de mineração possam executar com esses dados como entrada.

Além dos desafios relacionados ao formato não estruturado, dados textuais são muito heterogêneos, o que dificulta uma busca uniforme por soluções para processamento. Esses dados possuem uma organização distinta, tamanho e conteúdos variados, o que, por si só, já é suficiente para que o tratamento no processamento desses dados seja realizado de maneira distinta, de acordo com cada caso. Por exemplo, a classificação de documentos apresenta vantagens, em relação a classificação de questões em avaliações, pois dispõe um número maior de informações em seu conteúdo (MOHAMMED; OMAR, 2020).

Ainda que existam diversos desafios, alguns avanços já foram realizados na área de PLN (Processamento de Linguagem Natural). Técnicas para extração de características textuais e representação dessa informação foram desenvolvidas recentemente, muitas delas já atreladas a modelos pré-treinados para o processamento da linguagem (QIU et al., 2020). Estas técnicas facilitam a transformação de dados textuais para um formato mais apropriado para classificação, enquanto buscam manter o sentido e relações do texto original. A partir do momento em que os dados encontram-se em formato adequado de entrada, a quantidade de algoritmos para classificar esses dados é grande e vários deles podem ser utilizados, de acordo com o objetivo desejado.

Tirando proveito desses avanços e da disponibilidade de algoritmos de classificação,

diversos trabalhos obtiveram resultados relevantes na área de classificação automática de questões (SANGODIAH; MUNIANDY; HENG, 2015; SILVA; BITTENCOURT; MALDONADO, 2019). Ainda assim, existe um foco em uma classificação que avalia o nivelamento das questões em relação a sua dificuldade, de maneira a avaliar se existe um equilíbrio. Essa classificação é a taxonomia dos objetivos educacionais, popularmente conhecida como taxonomia de Bloom. Este conceito e os trabalhos com foco nele serão explorados na Seção 2.3.

No contexto de classificação de questões do Enade (Exame Nacional de Desempenho dos Estudantes), não foram encontrados trabalhos que explorem alternativas de automatização da classificação de questões. Assim, até então somente processos manuais (CHARAO et al., 2020) ou semi-automáticos (LIMA et al., 2018) já foram empregados para auxiliar na análise dos dados da prova. Dessa forma, esse trabalho busca reunir essas lacunas, preenchê-las e modelar a automatização da classificação de questões do Enade, a fim de contribuir para a análise de dados da avaliação.

1.1 OBJETIVOS

Este trabalho teve como objetivo geral buscar e utilizar técnicas e algoritmos para extração de atributos de texto e classificação de questões de provas, de maneira supervisionada, utilizando uma lista de tópicos pré-definida. No estudo de caso, foram utilizadas questões do Enade aplicadas em anos anteriores, para o curso de Ciência da Computação, como base de dados.

Para etapas de extração e pré-processamento, foram exploradas diversas técnicas para filtragem do texto inicial, a fim de reunir termos que contribuem mais para os processos realizados na sequência. Além disso, a possibilidade de explorar um algoritmo de OCR (*Optical Character Recognition*) para extração do texto das imagens foi uma etapa extra em relação a maioria de trabalhos na área de PLN, mas que agregou conhecimento.

Na extração de atributos, a investigação culminou no uso de técnicas clássicas da literatura. O uso destes algoritmos com ajuste de hiperparâmetros, e utilização de diferentes dados como entrada, culminou em algumas observações em relação aos resultados obtidos. Algoritmos de classificação habitualmente usados em classificação de textos também foram utilizados e mostraram os melhores resultados, enquanto a utilização de uma arquitetura mais simples de Rede Neural trouxe os piores resultados.

A avaliação e comparação dos algoritmos foi realizada através do uso de recursos gráficos da linguagem escolhida, para diversos contextos: visualização bidimensional de *embedding* de palavras, gráficos de erro e acurácia e matrizes de confusão. Todos estes recursos contribuíram para essa análise final entre as técnicas utilizadas.

1.2 JUSTIFICATIVA

Uma grande fonte de dados atualmente está em órgãos governamentais, que passaram a tornar os dados abertos à população, como medida de transparência. Nesse meio, as bases de dados sobre a educação brasileira são muito ricas na quantidade e qualidade dos dados que as compõem. Entretanto, nem todos os dados disponibilizados são bem aproveitados, visto que o foco dos trabalhos na área são os microdados[1] do INEP (Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira). Os microdados representam a menor fração de um dado e podem estar relacionados a uma pesquisa ou avaliação (QEDU, 2020).

A existência desses dados abertos possibilita estudos para verificação da situação atual das instituições, dos profissionais, dos alunos e do aprendizado, além de ser uma maneira segura e amparada de prover informações para a tomada de decisões em relação a cada uma dessas camadas do sistema educacional. Além das análises estatísticas, é importante incorporar a mineração de dados, a qual é definida como a aplicação de métodos estatísticos, análise de dados e aprendizado de máquina para exploração de bases de dados, com o objetivo de extrair informações novas e úteis para o benefício daquele escopo em que o dado foi originado (TUFFÉRY, 2011).

Ir além da análise dos microdados também é essencial, para que as considerações em relação aos dados educacionais possam ter um amparo maior e gerar propostas e soluções embasadas em uma quantidade maior de informações. Em um contexto de associação dos microdados com o conteúdo das respectivas avaliações, o ganho de informações é significativo, pois pode trazer um motivo para o desempenho obtido em relação a um dado específico. Esse motivo pode estar atrelado a diferentes contextos: tipo de questão, tamanho de questão, dificuldade da questão, tema da questão, entre outros. Porém, para que essas informações acerca das questões estejam disponíveis, a criação de modelos para classificá-las, dentro desses nichos, é um processo necessário e que facilita a realização desse estudo mais completo, pois substitui um processo manual e custoso.

1.3 ORGANIZAÇÃO DO TEXTO

Este trabalho está organizado em capítulos. O Capítulo 2 apresenta uma revisão de literatura sobre estudos com os dados abertos da educação, descoberta de conhecimento, algoritmos para preparação e para classificação de dados textuais e estudos acerca de classificação de questões. O Capítulo 3 mostra a metodologia utilizada para o desenvolvimento deste trabalho. O Capítulo 4 apresenta a acurácia dos classificadores e as dificuldades encontradas na utilização de cada um deles. Por fim, o Capítulo 5 traz a conclusão e

[1] http://inep.gov.br/microdados

as considerações finais.

2 REVISÃO DE LITERATURA

Neste capítulo, são mostrados os estudos relacionados à exploração dos dados abertos da educação, através de abordagens estatísticas e de mineração de dados. Apresenta-se estudos acerca da descoberta de conhecimento, um processo fundamental para o processamento de textos e também de outros tipos de dados. E, por fim, são exibidos trabalhos com foco na classificação de textos, com uma parte da seção dedicada à classificação de questões e os tipos de classificação abordados na literatura.

2.1 DADOS ABERTOS DA EDUCAÇÃO

A avaliação do ensino superior brasileiro teve início em 1993, com o PAIUB (Programa de Avaliação Institucional das Universidades Brasileiras). Após, em 2004, a implantação do Enade foi um passo importante para a avaliação do rendimento nesse âmbito. Uma revisão sistemática de trabalhos relacionados à análise de dados do Enade e do Enem (Exame Nacional do Ensino Médio) foi realizada por LIMA et al. (2019). Para os trabalhos relacionados ao Enade, os autores mostram que há uma maioria de trabalhos com foco nas notas do exame e uma quantia reduzida que aborda o conteúdo da prova. Ambos os temas são abordados em análises puramente estatísticas, sendo que os trabalhos que tratam sobre o conteúdo da prova não estão especificamente interessados no conteúdo das questões. Por outro lado, um único trabalho utilizou mineração de dados, mas para analisar as notas dos alunos na avaliação.

Em meio ao estudo de desempenho nas avaliações, a classificação de questões pode cumprir um papel importante. No contexto do Enade, a classificação de questões já foi utilizada, em busca de atrelar os resultados aos temas das questões, por CHARAO et al. (2020). Neste trabalho, a classificação das questões foi realizada de maneira manual, pois o objetivo do trabalho é usar essa classificação, junto as notas, para determinar desempenho específico de cada conteúdo curricular. O desempenho é obtido para uma instituição em específico, em comparação com os dados gerais, a fim de subsidiar argumentos para revisão do projeto pedagógico de curso da instituição. A existência de um modelo automático classificador de questões para o Enade, pode facilitar a ampliação de estudos do tipo para outros cursos e instituições.

No âmbito de classificação e análise de questões do Enade, LIMA et al. (2018) utilizam uma abordagem de classificação de questões através de dicionários. Além da classificação, as questões, posteriormente categorizadas, são atreladas a notas, para realização das considerações acerca dos resultados do exame, de acordo com o tema das questões. Apesar do trabalho tratar da classificação e fazer essa relação com as notas,

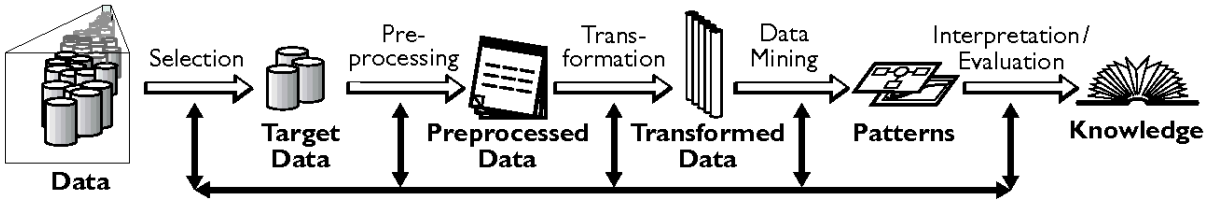
a classificação de questões através de dicionários utiliza um processo manual, pois há a necessidade de construir o dicionário, e estático, visto que está atrelado a pré-definição do que vem a ser o conteúdo da questão, através de palavras-chave.

2.2 DESCOBERTA DE CONHECIMENTO

Para PROVOST; FAWCETT (2013), a conversão de eventos como o aumento da quantidade de dados, o aumento do poder computacional e o desenvolvimento de algoritmos para análises mais profundas de conjuntos de dados resultou na ascensão do fenômeno conhecido como ciência de dados, sendo que consideram o aperfeiçoamento da tomada de decisões o objetivo mestre desse fenômeno. A ciência de dados está envolvida com a extração de informações e de conhecimento acerca de um conjunto de dados, tendo como um dos possíveis resultados essa tomada de decisões, em um determinado contexto, baseado no que se é extraído dos dados.

Tal tarefa, pode ser modelada de diferentes maneiras, seguindo métodos que foram desenvolvidos e apresentados ao longo dos anos, de maneira a estabelecer padrões de exploração dos dados. O estudo apresentado por AZEVEDO; SANTOS (2008) mostra as diferenças e similaridades entre novas técnicas desenvolvidas, SEMMA (*Sample, Explore, Modify, Model, Assess*) e CRISP-DM (*CRoss-Industry Standard Process for Data Mining*), e a técnica mais tradicional na área, o KDD (*Knowledge Discovery in Databases*). Os autores mostram que existe uma grande similaridade entre as fases das técnicas KDD e SEMMA, enquanto a técnica CRISP-DM apresenta algumas etapas extras no início e fim do processo.

Figura 2.1 – Ilustração das etapas do KDD.



Fonte: (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996b).

Devido ao já estabelecimento da técnica de KDD na área e também por ser apropriada para o problema em questão, essa foi a técnica escolhida. O processo de KDD é definido como a maneira não-trivial, de identificar padrões válidos, novos, potencialmente úteis e compreensíveis nos dados em questão (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996b). Sua divisão ocorre em 5 etapas: seleção, pré-processamento, transformação, mineração de dados e avaliação (Figura 2.1). Dentre seus potenciais, desde o início, o

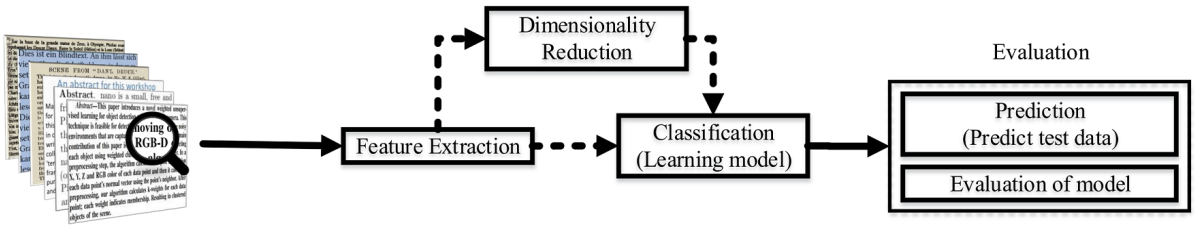
processo já visava e modelava, através de suas etapas, o processamento de linguagem natural desde o seu estágio mais simples até o entendimento da linguagem (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996a).

2.3 CLASSIFICAÇÃO DE TEXTOS E QUESTÕES

O PLN é uma área com um número crescente de trabalhos, que acompanha o crescimento da quantidade de dados textuais sendo produzidos. Além dos trabalhos abordados na sequência, é importante destacar a quantidade de materiais que abordam a classificação de texto e de questões em sites como Kaggle[1], Medium[2], Towards Data Science[3] e outros, que ajudam a disseminar o conhecimento de maneira mais simplificada e prática, através de casos reais de uso.

Dentro da área de PNL, um dos campos que se destaca é o de classificação e categorização de textos, o qual faz uso de algoritmos clássicos, além de uma diversidade de algoritmos desenvolvidos recentemente para tratamento de textos. Todos estes métodos servem tanto para extração de características e representação de texto, quanto para o processamento desses dados, através de classificadores.

Figura 2.2 – Etapas geralmente usadas em trabalhos de classificação de textos.



Fonte: (KOWSARI et al., 2019).

Nesse contexto, KOWSARI et al. (2019) traz uma revisão de literatura completa acerca do tema, indicando os escopos em que a classificação textual é aplicada: a nível de documento, a nível de parágrafo, a nível de frase e a nível de trecho de frase, sendo cada um dos níveis existentes uma porção do grau acima. Para o contexto atual, os classificadores selecionados estarão atuando em um escopo de parágrafo, visto que as questões de prova são compostas por diversas frases e serão analisadas individualmente, não como documento inteiro. Além desses escopos, a revisão traz técnicas utilizadas na literatura para cada uma das etapas ilustradas na Figura 2.2, as quais podem ser atreladas à etapas do KDD (Figura 2.1).

[1] https://www.kaggle.com/
[2] https://medium.com/
[3] https://towardsdatascience.com/

Seguindo o fluxo da Figura 2.2, após a seleção dos dados, a extração de atributos de textos é uma etapa fundamental no processo de classificação. É através dessa etapa que o formato textual se torna apropriado para uso como entrada em classificadores, ao mesmo tempo em se deseja preservar as características do texto, tanto em relação a própria palavra, quanto no que se refere ao contexto em que a palavra se encontra (palavras próximas). Dentre os métodos mostrados por KOWSARI et al. (2019), os métodos TF-IDF (*Term Frequency Inverse Document Frequency*) e Word2Vec são os que apresentam o melhor custo benefício, dado que apresentam poucas limitações e nenhuma delas relacionadas ao desempenho do algoritmo.

Quanto à redução de dimensionalidade, o próprio fluxo indica que é uma etapa opcional. Além da aplicação de redução para posterior classificação dos dados, a redução de dimensionalidade pode ser útil para visualização de *embeddings* de palavras. Dentre os algoritmos, MAATEN; HINTON (2008) introduzem o t-SNE (*t-distributed Stochastic Neighbor Embedding*) para visualização de dados em planos bidimensionais ou tridimensionais. Os autores comparam a técnica com outras existentes na literatura e mostram as vantagens da sua utilização em relação a esses outros métodos, tanto na teoria, quanto na prática, através de alguns exemplos com casos reais.

Dentre a diversidade de classificadores existentes, os algoritmos de *Deep Learning* apresentam uma variedade de arquiteturas, ou até mesmo uma combinação delas, já aplicada para a tarefa de classificação textual. Nesse contexto, até mesmo algoritmos normalmente utilizados para processamento de imagens, como a CNN (*Convolutional Neural Network*), já foram utilizados. LAI et al. (2015) apresenta essa arquitetura em uma variante bidirecional (recorrente) que realiza desde a extração de atributos e representação das palavras até a classificação da entrada, alcançando acurácia próxima ou superior a trabalhos anteriores para quatro *datasets* distintos.

Junto à arquitetura CNN, também é realizada a integração de uma rede LSTM (*Long Short-Term Memory*). ZHOU et al. (2015) indica que este tipo de arquitetura tenta solucionar o problema de dependência de dados entre iterações distantes em uma rede RNN (*Recurrent Neural Network*), visto que é uma memória com dados de iterações passadas, não apenas da última. Os autores conseguiram resultados melhores ou próximos a outros algoritmos comumente usados para análise de sentimentos e para a classificação de questões.

Apesar de utilizarem *Deep Learning*, LAI et al. (2015) e KOWSARI et al. (2018) citam os algoritmos NB (*Naive Bayes*) e SVM (*Support Vector Machine*) como classificadores populares e frequentemente utilizados para classificação de textos. De fato, KOWSARI et al. (2019), em sua revisão, corrobora essas afirmações e indica que ambos são algoritmos mais simples que as arquiteturas de *Deep Learning*, além de apresentarem menos limitações para resolução de tal problema.

Entrando em um escopo mais específico da classificação de textos, a classificação

automática de questões já foi estudada sob diversas perspectivas. A revisão de literatura proposta por SANGODIAH; MUNIANDY; HENG (2015), acerca da classificação automática de questões utilizando aprendizado de máquina, mostra que os trabalhos com foco em classificação no ambiente educacional costumam realizar a classificação em torno da taxonomia de Bloom. A taxonomia de Bloom modela, de forma hierárquica, o aprendizado cognitivo desde o conhecimento em fatos específicos até os níveis de análise, síntese e avaliação de conteúdos (BLOOM; KRATHWOHL; MASIA, 1984). Essa classificação está ligada ao nivelamento de dificuldade das questões da prova, onde é esperado que haja um equilíbrio para que a avaliação seja considerada adequada para testar o conhecimento.

Nesse contexto de classificação de questões usando a taxonomia de Bloom, JAYAKODI; BANDARA; PERERA (2015) usam algoritmos de similaridade entre palavras para comparação com dados da base WordNet[4] e classificação baseada em regras. Os autores focam na extração de verbos, visto que as categorias da taxonomia de Bloom podem ser bem definidas através desse tipo de palavra, junto ao contexto, pois alguns verbos estão em mais de uma categoria. Os resultados apresentam boa acurácia e o escopo é limitado a avaliações de um curso de engenharia. Usando outras estratégias, MOHAMMED; OMAR (2020) usam técnicas para extração de atributos das palavras, de maneira a prepará-las como entrada para algoritmos de aprendizado de máquina. O foco do trabalho é avaliar essas técnicas de extração de atributos, usando três algoritmos para classificação: KNN (*K-Nearest Neighbours*), LR (*Logistic Regression*) e SVM. É possível notar que a acurácia dos algoritmos varia, porém a relação com as técnicas de extração de atributos é semelhante.

Outro ponto importante para a classificação de questões é a definição de como o dado textual será usado como entrada em classificadores. No contexto de PLN existe um conjunto de técnicas, os modelos semânticos distribucionais, que transformam os textos em formato adequado para que esses textos possam ser interpretados pelos algoritmos de classificação e predição. Mais do que isso, essas técnicas usadas oferecem a capacidade de manter o significado semelhante ao do dado inicial e como consequência oferecem aos métodos de classificação uma maneira de interpretar o dado, sem que ele perca as características que o definem em seu contexto textual (LENCI, 2018).

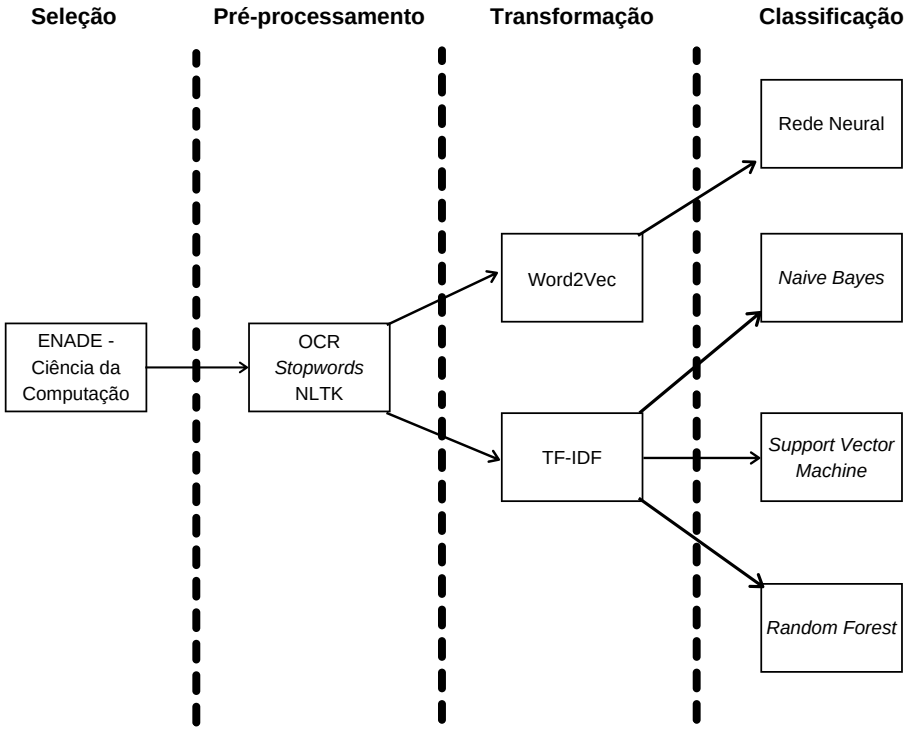
[4] https://wordnet.princeton.edu/

3 DESENVOLVIMENTO

Este trabalho segue uma metodologia de pesquisa exploratória descrita por THEODORSON; THEODORSON (1969) como um estudo preliminar. Esta metodologia permite ao investigador definir o seu problema e a sua hipótese com mais precisão, além de permitir a escolha de técnicas mais adequadas, decidir quais pontos necessitam de mais atenção e de alertá-lo para potenciais dificuldades.

Junto a essa metodologia de pesquisa, utilizou-se a técnica de KDD (Seção 2.2) como base para definição e execução das etapas de tratamento dos dados, desde a seleção de dados específicos, até a análise dos resultados obtidos após a execução de algoritmos de mineração de dados. Em cada uma das fases do processo de KDD, foram utilizadas bibliotecas da linguagem Python para execução de algoritmos responsáveis por tal etapa. As bibliotecas e algoritmos utilizados serão descritos nas seções específicas de cada uma dessas etapas. Todo o código implementado para execução dos algoritmos foi registrado com a utilização do recurso Jupyter Notebook, no ambiente Google Colaboratory[1], com execuções nas máquinas remotas que o ambiente fornece.

Figura 3.1 – Técnicas usadas nas etapas do trabalho.



As etapas e os principais algoritmos e técnicas utilizadas em cada uma delas estão ilustradas na Figura 3.1. Nas seções que seguem, os procedimentos realizados na Seção 3.1 e Seção 3.2 foram executadas no período inicial e seu resultado armazenado para

[1] https://colab.research.google.com/

posterior uso nas fases da Seção 3.3 e Seção 3.4, as quais são refeitas a cada novo método selecionado para utilização. A Seção 3.5 indica onde será mostrada a comparação entre os algoritmos usados.

3.1 DADOS E SELEÇÃO

Essa é a etapa primordial do KDD, que inclui a seleção de uma base de dados e, caso necessário, a escolha de dados específicos dentro dessa base, para posterior tratamento e aplicação. No estudo de caso, a base de dados escolhida foi a de provas do Enade aplicadas para alunos do curso de Bacharelado em Ciência da Computação. Mais especificamente, os dados utilizados são compostos por questões das provas de Computação do Enade, aplicadas nos anos de 2005, 2008 e 2011, e das provas do curso de Bacharelado em Ciência da Computação do Enade aplicadas nos anos de 2014 e 2017.

Nas 3 provas mais antigas, existem questões de formação geral, questões específicas comuns a todos os cursos envolvidos (Ciência da Computação, Engenharia da Computação e Sistemas de Informação) e questões específicas para cada curso envolvido no exame. Para essas provas, de um total de 220 questões, foram filtradas 90 questões correspondentes às categorias "*questões específicas comuns a todos os cursos*" e "*questões específicas para o curso de Ciência da Computação*". Nas 2 provas mais recentes, como a prova já é específica para o curso, foram ignoradas apenas as questões de formação geral, que estão presentes em todas as provas. Sendo assim, foram separadas 60 questões de um total de 80, para essas duas provas. Ao total foram selecionadas 150 questões de um total de 300 questões das 5 provas utilizadas como base de dados.

Para cada uma das questões específicas filtradas, existe uma classificação de acordo com uma lista de tópicos, onde esses tópicos se referem ao domínio ao qual pertence o conteúdo daquela questão. Uma questão pode ter mais de um tópico atrelado a ela, porém aqui a classificação será limitada ao tópico principal. Essa lista de tópicos pode ser visualizada na Tabela 3.1, que atrela cada um desses tópicos a um código identificador, de maneira a facilitar a utilização desses dados no restante do texto e em figuras.

Além da diferença de abrangência entre as três provas mais antigas e as duas mais recentes, é importante destacar que o conjunto de provas recentes apresenta uma classificação atribuída pelo INEP, enquanto o conjunto de provas antigas tem uma classificação realizada manualmente de acordo com a percepção de um profissional em relação ao conteúdo das questões e aos tópicos de classificação do conjunto mais recente.

Tabela 3.1 – Código de identificação para os tópicos de classificação das questões.

Código	Tópico
T0	Algoritmos e Estruturas de Dados
T1	Arquitetura de Computadores e Sistemas Operacionais
T2	Banco de Dados
T3	Compiladores
T4	Computação Gráfica e Processamento de Imagem
T5	Engenharia de Software e Interação Homem-Computador
T6	Fundamentos e Técnicas de Programação
T7	Inteligência Artificial e Computacional
T8	Lógica e Matemática Discreta
T9	Paradigmas de Linguagens de Programação
T10	Probabilidade e Estatística
T11	Redes de computadores
T12	Sistemas Digitais
T13	Sistemas Distribuídos
T14	Teoria da Computação
T15	Teoria dos Grafos
T16	Ética, computador e sociedade

3.2 PRÉ-PROCESSAMENTO

Nessa etapa do KDD costuma-se usar algoritmos para limpeza dos dados, considerando dados incompletos e ruídos. Previamente a essa limpeza, foi realizada a conversão das questões selecionadas na etapa anterior, pois os dados descritos na seção anterior encontram-se em recortes no formato de imagem, mais especificamente em formato PNG. A extração desses recortes foi feita para utilização em um trabalho anterior (CHARAO et al., 2020).

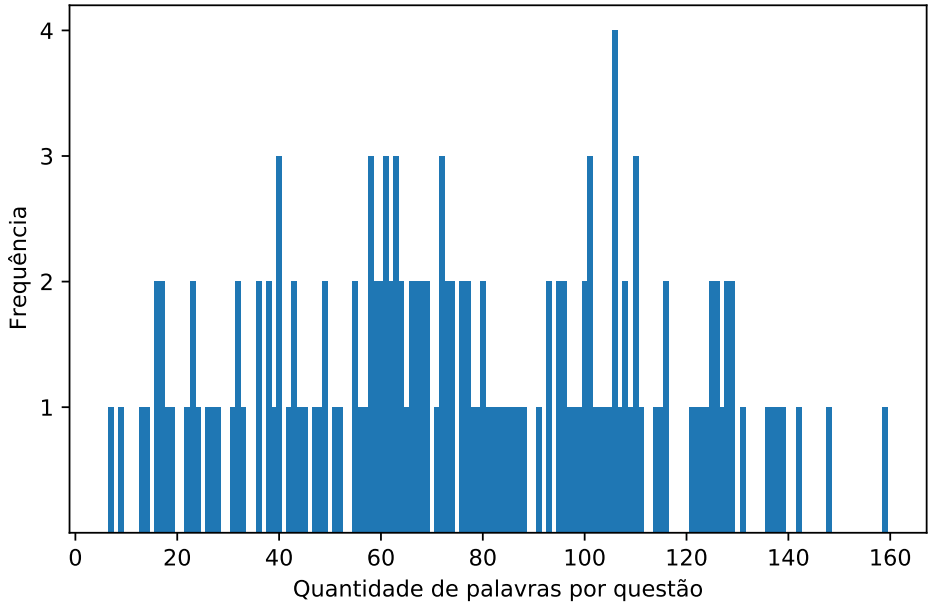
A partir disso, o processamento inicial envolveu a conversão do dado nesse formato para formato de texto (*string*, dentro da linguagem) e armazenamento em arquivos TXT para processamento posterior. Tal tarefa foi realizada utilizando a biblioteca `pytesseract`[2], que oferece um algoritmo de OCR, que reconhece e lê o texto de imagem para uma saída. O algoritmo foi executado para o conjunto completo de questões e o texto foi armazenado em arquivos.

Com os dados em formato textual, foi realizada a limpeza, através de uma filtragem para remoção da pontuação e de *stopwords*, visto que são caracteres que pouco agregam na classificação, já que comumente se encontram em todos os tipos de texto. Além disso, como a utilização do algoritmo de OCR não garante uma transformação completamente correta para o formato textual, foram utilizados vocabulários/corpus da biblioteca NLTK[3] (*Natural Language Toolkit*) para remoção de palavras que não estão contidas nesse voca-

[2] https://github.com/madmaze/pytesseract
[3] https://www.nltk.org/

bulário, ou seja, palavras, termos e símbolos sem significado algum.

Figura 3.2 – Quantidade de palavras por questão pré-processada.



A etapa anterior fez com que o número de palavras para as 150 questões do conjunto de dados ficasse distribuída de acordo com o gráfico mostrado na Figura 3.2. Como mostrado, o número de palavras varia de 7 a 159 palavras, com uma média ponderada de 76 palavras por questão.

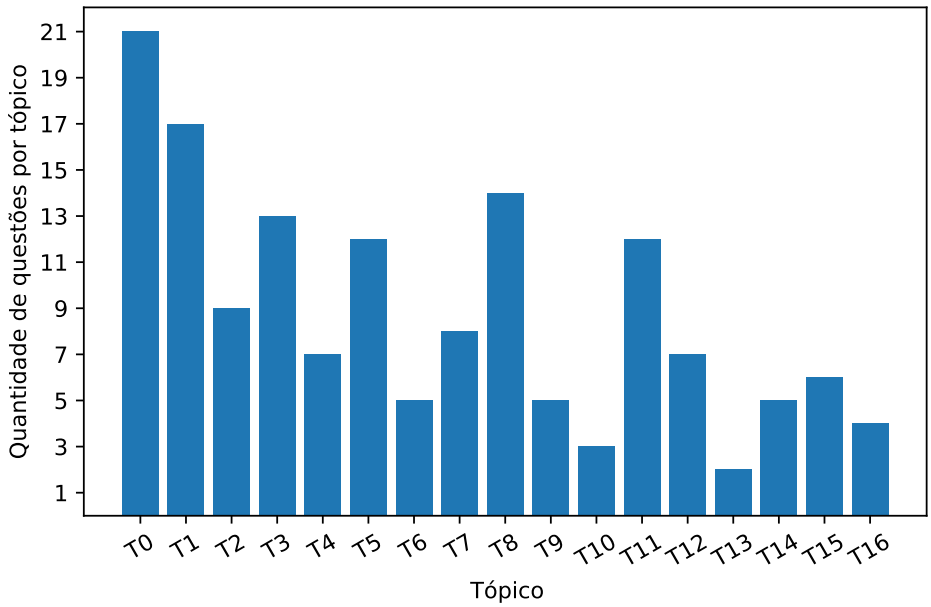
Além do pré-processamento do texto das questões, a classificação das questões também passa por essa etapa, de maneira a auxiliar na utilização de classificadores supervisionados. Para esses dados, foi realizada a conversão de todos os termos para caixa baixa. Esses tópicos foram então organizados em uma estrutura de dados com índices atrelados ao do conjunto de questões, de maneira a parear a questão com seu tópico de classificação.

Na sequência, considerando a atribuição de códigos aos tópicos de classificação da Tabela 3.1, é possível visualizar, na Figura 3.3, a distribuição das 150 questões do conjunto de dados entre esses tópicos, usando os códigos de identificação. O número de questões varia de 2 até 21 questões por tópico, havendo uma distribuição não uniforme entre eles, o que pode afetar a acurácia em relação a tópicos específicos.

3.3 TRANSFORMAÇÃO

A etapa de transformação inclui a conversão dos dados em formato de texto para um formato numérico, o qual é comumente aceito por classificadores. Dessa forma, houve uma seleção de técnicas que propõem realizar a transformação para o formato adequado,

Figura 3.3 – Quantidade de questões por tópico de classificação.



de acordo com os classificadores utilizados. As técnicas selecionadas foram Word2Vec e TF-IDF, que serão abordadas especificamente na Subseção 3.3.1 e Subseção 3.3.2, respectivamente.

3.3.1 Word2Vec

A técnica de *word embedding* conhecida popularmente como Word2Vec (MIKOLOV et al., 2013) realiza a transformação de palavras contidas em um determinado conjunto de texto, levando em consideração o contexto (através de uma janela de palavras próximas), para um formato numérico e vetorial. A base do algoritmo é uma rede neural que é implementada utilizando uma das possíveis arquiteturas existentes: CBOW (*Continuous Bag-of-Words*) e Skip-gram. Cada uma dessas arquiteturas trata de uma maneira diferente a questão da janela de palavras próximas, em relação ao termo em questão.

Para a execução desse algoritmo, dois parâmetros são de extrema importância: o tamanho do *embedding*, que é o tamanho do vetor de cada palavra, e o tamanho da janela, que indica a quantidade de palavras próximas ao termo que são levadas em consideração ao executar a rede neural para ajustar seus valores de *embedding*. KHATTAK et al. (2019) indica que o tamanho do *embedding* costuma ser um valor entre 50 e 500, modificado experimentalmente, e o tamanho da janela costuma ser de 5 ou 10 palavras. Uma tentativa de incluir bigramas no vocabulário de treinamento resultou majoritariamente no encontro de termos comuns entre as questões (”*pode_ser*”, ”*avalie_afirmações*”, ”*opção_correta*”, etc) e foi descartada para as execuções.

Para utilização do Word2Vec, foi escolhida a implementação da biblioteca `gensim`[4]. A biblioteca possibilita realizar o treinamento de um modelo e fornece recursos para mostrar a similaridade entre os termos treinados. Após os treinamentos, o recurso t-SNE[5], utilizado para redução de dimensionalidade, foi empregado para visualização dos *embeddings* em um espaço bidimensional. A técnica descrita nesta seção foi utilizada para preparar as entradas para o algoritmo de Redes Neurais, como será mostrado na Subseção 4.1.

3.3.2 TF-IDF

Outra técnica utilizada foi TF-IDF. Essa medida indica o quão concentrada a ocorrência de uma palavra está em relação a um conjunto de documentos, onde termos com um valor mais elevado são os que melhor caracterizam o tópico do documento (RAJARAMAN; ULLMAN, 2011). Essa caracterização de tópicos através de termos é uma abordagem que deve ser útil na classificação de questões, que seriam os documentos para o estudo de caso atual.

Para execução desse algoritmo, foram configurados 2 parâmetros: frequência mínima de cinco ocorrências para receber o atributo, além da consideração de unigramas e bigramas. A utilização de trigramas também poderia ser útil, vide termos como "*redes de computadores*" e "*engenharia de software*", mas como houve a filtragem de *stopwords*, a configuração para bigramas já é suficiente e também abrange termos como "*computação gráfica*", "*sistemas distribuídos*" e "*sistemas operacionais*".

Para utilização do TF-IDF, foi escolhida a implementação `TfidfVectorizer`[6] da biblioteca Scikit-learn. Essa implementação realiza a contagem de tokens em um conjunto de texto e aplicação do algoritmo propriamente dito. A técnica descrita nesta seção foi utilizada para preparar as entradas para os algoritmos *Naive Bayes*, *Support Vector Machine* e *Random Forest*, como será mostrado nas Seções 4.2, 4.3 e 4.4, respectivamente.

3.4 CLASSIFICAÇÃO

Para classificação automática das questões foram utilizados quatro métodos supervisionados: Redes Neurais, *Naive Bayes*, *Support Vector Machine* e *Random Forest*. O primeiro escolhido por utilização prévia em outro contexto, apresentado na Subseção 3.4.1, enquanto os outros foram escolhidos com base em resultados apresentados anteriormente na literatura e são descritos nas Subseções 3.4.2, 3.4.3 e 3.4.4, respectivamente.

[4] https://radimrehurek.com/gensim/
[5] https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html
[6] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

3.4.1 Redes Neurais

O algoritmo de Rede Neural, também conhecido como Rede Neural Artificial, é um modelo preditivo motivado pela forma como o cérebro funciona (GRUS, 2019). Isso acontece dado que o modelo é composto por neurônios conectados, em camadas, os quais se alimentam através da saída de outros. Atualmente, existem diversas variantes desse modelo, as quais podem ser aplicadas para os mais diversos problemas na área de aprendizado de máquina. Para o estudo de caso em questão, o modelo selecionado foi uma rede neural simples com uma camada de entrada, uma camada escondida e uma camada de saída, que será treinada de maneira supervisionada.

Em relação a escolha de uma única camada escondida (camada intermediária), HEATON (2008) indica a maioria dos problemas aplicados em redes neurais podem ser solucionados com uma única camada escondida, pois esse fator faz com que seja possível aproximar arbitrariamente qualquer função contínua de um espaço finito para outro. Além disso, a publicação também traz indicações para escolha do número de neurônios presente nessa camada, fator que será mostrado na Seção 4.1.

Em relação ao treinamento supervisionado, ele será aplicado porque há a disponibilidade da classificação correta das questões, o que possibilita calcular o erro entre a saída da rede e a resposta correta. Esse erro será usado no processo conhecido por *back-propagation*, que tem como objetivo ajustar os pesos dos neurônios da rede, de maneira a fazer com que o erro diminua nas próximas execuções. Esse erro é calculado através da função de entropia cruzada, uma função comumente utilizada para o cálculo do erro em redes neurais, mais notavelmente em problemas de classificação com múltiplas classes (GORDON-RODRIGUEZ et al., 2020), como o atual.

Outros detalhes técnicos da rede incluem a utilização da função sigmóide na camada escondida e da função softmax na camada de saída. A função sigmóide é uma função que transforma a entrada em um intervalo entre 0 e 1, e é amplamente usada por ser uma função não-linear, enquanto a função softmax é uma combinação de múltiplas funções sigmóide utilizada para problemas de classificação com múltiplas classes, para expressar probabilidade da entrada pertencer a alguma das classes na camada de saída (SHARMA, 2017). Para utilização da função softmax na camada de saída, esta foi adaptada para o formato *one-hot encoding*.

Considerando a modelagem descrita, foi utilizada a biblioteca TensorFlow[7] para compilação e execução da rede. Os parâmetros ajustados ao longo das execuções foram a quantidade de neurônios na camada escondida, de acordo com o tamanho da entrada (definido na execução dos algoritmos de *embedding*), e o tamanho do lote de entrada.

[7] https://www.tensorflow.org/

3.4.2 Naive Bayes

O algoritmo *Naive Bayes* é um método probabilístico que faz utilização do Teorema de Bayes para processamento dos resultados. A chave para esse modelo é fazer a suposição de que as presenças, ou ausências, de determinado parâmetro de entrada sejam independentes umas das outras (GRUS, 2019).

Duas variantes desse classificador são comumente usadas na tarefa de classificação de textos: *Bernoulli Naive Bayes* e *Multinomial Naive Bayes*. ABBAS et al. (2019) descreve que a variante multinomial é uma melhoria da variante Bernoulli e sua utilização é dominante em trabalhos recentes devido à sua eficiência. SINGH et al. (2019) mostram que o classificador multinomial tem um desempenho levemente melhor, principalmente ao utilizar um conjunto de dados consideravelmente pequeno (com 312 instâncias, no caso em questão).

Sendo assim, foi utilizada a biblioteca Scikit-learn para modelagem e execução do algoritmo `MultinomialNB`[8], o qual é implementado para dados distribuídos de maneira multinomial e é a variante comumente usada para dados textuais, como descrito pela própria biblioteca do modelo.

3.4.3 Support Vector Machine

O algoritmo *Support Vector Machine* foi apresentado em 1995, inicialmente como um classificador binário, apenas para duas classes. A ideia do algoritmo é mapear um vetor de valores de entrada para um espaço com diversas dimensões, onde é criada uma superfície de decisão generalizável (CORTES; VAPNIK, 1995).

Para a classificação em múltiplas classes com esse algoritmo, já foram desenvolvidas diversas técnicas, incluindo algumas que realizam a combinação de diversas classificações binárias para chegar ao resultado final. Na comparação descrita por HSU; LIN (2002) são descritos alguns desses métodos, entre eles as clássicas implementações "*one-vs-one*" e "*one-vs-all*", ambas testadas de maneira preliminar com os dados em questão, onde a implementação "*one-vs-all*" mostrou uma melhor acurácia.

Dessa forma, apesar dos testes com a implementação `SVC`[9], da biblioteca Scikit-learn, foi utilizado o classificador `LinearSVC`[10], da mesma biblioteca, com parâmetros padrão para modelagem e execução do algoritmo.

[8] https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html
[9] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[10] https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

3.4.4 Random Forest

O algoritmo *Random Forest* também foi apresentado em 1995, como um classificador baseado em árvores de decisão. A proposta do algoritmo é de gerar diversas árvores de decisão em partes aleatórias do espaço de atributos, de maneira a utilizar essas diversas árvores para classificação e superar a limitação apresentada por uma única árvore de decisão, a qual não pode crescer indefinidamente e acabar resultando em sobreajuste para os dados de entrada (HO, 1995).

Para o caso em questão, foi utilizada a biblioteca Scikit-learn para modelagem e execução do algoritmo `RandomForestClassifier`[11]. Duas variáveis foram alteradas a partir do ponto inicial, buscando melhoria na acurácia: a quantidade de árvores de decisão e a profundidade máxima de cada uma dessas árvores. Os fatores estão diretamente atrelados ao desempenho, em tempo de execução, do classificador, porém causaram pouca mudança na acurácia, como será descrito na Seção 4.4.

3.5 ANÁLISE

A parte de análise dos resultados será discutida na Seção 4.5. O objetivo principal da análise é comparar os algoritmos e técnicas utilizadas, além de trazer particularidades e detalhes observados em cada um deles, seja durante sua execução ou na visualização dos resultados, através dos recursos gráficos utilizados.

[11] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

4 RESULTADOS

O capítulo de resultados será dividido de acordo com os algoritmos de transformação e de classificação utilizados, sendo que cada seção apresenta uma combinação de dois algoritmos. Para cada combinação de algoritmos, foram criados 3 subcasos. Na Seção 4.1, que traz os resultados da utilização em conjunto dos algoritmos Word2Vec e Redes Neurais, os subcasos foram criados de acordo com o conjunto de dados utilizado para treinamento do *embedding*, seguindo a lógica abaixo:

- Subcaso 1: treinamento do *embedding* somente com o texto das questões.
- Subcaso 2: treinamento do *embedding* com *corpus* de artigos científicos.
- Subcaso 3: treinamento do *embedding* com *corpus* de livros de computação.

Nas seções restantes (Seção 4.2, Seção 4.3 e Seção 4.4), que trazem a combinação do algoritmo TF-IDF com os classificadores *Naive Bayes*, *Support Vector Machine* e *Random Forest*, respectivamente, a divisão dos subcasos é baseada na divisão de dados utilizada para treinamento e validação. A organização dos dados é listada abaixo:

- Subcaso 1:
  - Dados de treinamento: questões de 2014 e 2017 (40%).
  - Dados de validação: questões de 2005, 2008 e 2011 (60%).
- Subcaso 2:
  - Dados de treinamento: questões de 2005, 2008 e 2011 (60%).
  - Dados de validação: questões de 2014 e 2017 (40%).
- Subcaso 3:
  - 80% de todas as questões (escolhidas aleatoriamente).
  - 20% de todas as questões (escolhidas aleatoriamente).

Os dois primeiros casos de teste foram criados visando a diferença existente, citada na Seção 3.1, entre as provas de 2005, 2008 e 2011, e as provas de 2014 e 2017. Já o terceiro subcaso foi usado para que houvesse uma variante com uma quantidade maior de questões no treinamento do modelo.

4.1 WORD2VEC E REDES NEURAIS

Para o classificador Redes Neurais, houve a utilização do algoritmo Word2Vec para a transformação das entradas. Ao longo das primeiras execuções, foram criados três subcasos de teste, apresentados acima, que diferem em relação ao conjunto de dados utilizado para treinamento do *embedding*. Esse fato ocorreu devido a necessidade da exploração de uma quantidade maior de dados para fazer com que o *embedding* trouxesse uma representação mais efetiva e segura para a execução do classificador.

De maneira geral, o algoritmo de Word2Vec foi configurado 2 vezes em cada subcaso, para formar *embeddings* de tamanho 50 e de tamanho 300. A rede foi modelada com uma camada escondida e a quantidade de neurônios nessa camada foi inicialmente guiada por orientações de HEATON (2008). Assim, com valores iniciais de 32 neurônios (*embedding* de tamanho 50) e 256 neurônios (*embedding* de tamanho 300), foram testados valores ao redor, porém a variação na acurácia foi pequena. Em relação ao tamanho do lote, o valor inicial foi de 32 entradas, porém houve uma melhora na acurácia ao reduzir esse valor para 8, em todos os subcasos.

Após o treinamento do *embedding*, foi calculado o vetor médio para cada questão, para utilização como entrada da rede neural. Na sequência, a modelagem e compilação da rede, para que houvesse o treinamento e validação do modelo. Para cada um dos subcasos serão descritos mais especificamente quais dados foram utilizados e a frequência mínima de ocorrências da palavra para ser incluída no treinamento. Além disso, será mostrada a distribuição das cem palavras mais frequentes e dos termos pertencentes aos nomes dos tópicos, observados na Tabela 3.1. No entanto, essa visualização visa observar a escala de localização das palavras e sua distribuição, de maneira geral, pois mostrar os dados específicos nesta visualização é um processo mais delicado.

4.1.1 Subcaso 1: *Embedding* com texto das questões

Neste subcaso, apenas o texto das 150 questões selecionadas foi utilizado para formação dos *embeddings*. Por ser um subcaso com poucos dados, palavras com três ocorrências mínimas em todo o texto foram consideradas para o treinamento. Com essa configuração, foi obtido um vocabulário com 1026 palavras ao total. Na Figura 4.1 e Figura 4.2 podemos observar a distribuição entre alguns desses termos.

De maneira geral, é possível observar que as palavras encontram-se espalhadas nesse espaço bidimensional e que há um grupo de palavras, na parte superior de cada uma das imagens, onde concentram-se boa parte de termos computacionais. Dentre os termos que se encontram nesse grupo estão: ”*arquitetura*”, ”*linguagens*”, ”*gráfica*”, ”*processamento*”, ”*probabilidade*”, entre outros, que são os termos que nomeiam os tópicos de

Figura 4.1 – *Embedding* de 50 posições (subcaso 1).

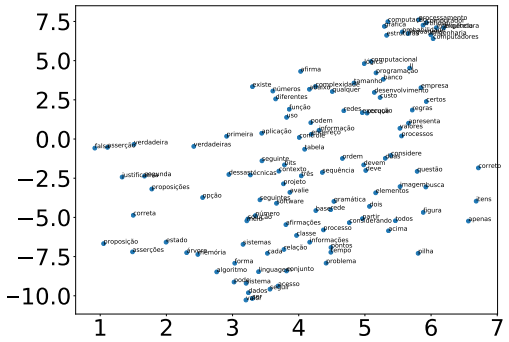
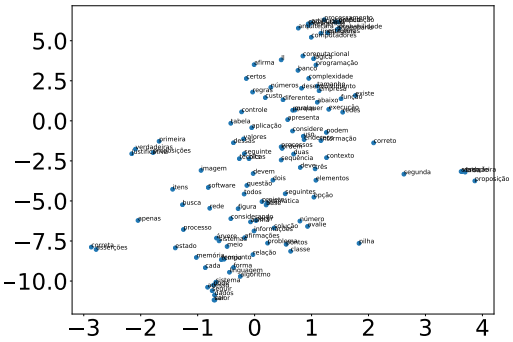


Figura 4.2 – *Embedding* de 300 posições (subcaso 1).



classificação.

Figura 4.3 – Erro e acurácia - *Embedding* de 50 posições (subcaso 1).

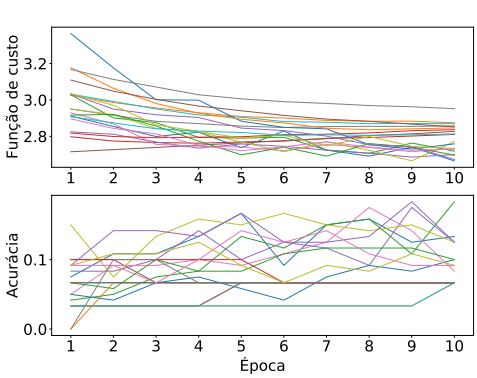
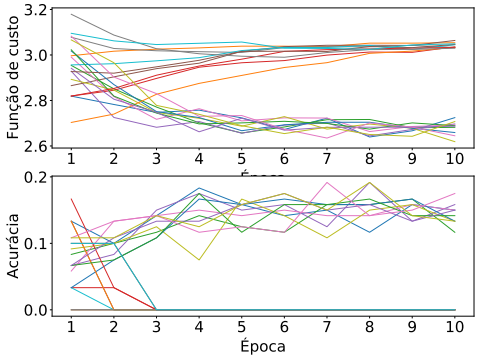


Figura 4.4 – Erro e acurácia - *Embedding* de 300 posições (subcaso 1).



Com relação ao treinamento e validação do classificador de questões através do algoritmo de Redes Neurais, é possível observar o erro, na parte superior da Figura 4.3 e Figura 4.4, e a acurácia na parte inferior. Nos gráficos acima, a diminuição do erro é lenta e pouco significante, ou inexistente, dado que em partes o erro é crescente. Isso ocasiona em uma acurácia abaixo de 20% para a maioria dos casos, apresentando um comportamento aleatório e sem perspectiva de melhora ao longo do tempo.

4.1.2 Subcaso 2: *Embedding* com *corpus* de teses de computação

Neste subcaso, o texto das 150 questões selecionadas foi utilizado para formação dos *embeddings*, junto a um *corpus* de textos científicos em português do Brasil, no do-

mínio da computação, conhecido como CorpusTCC (PARDO; NUNES, 2003). O subcaso seguiu a regra de palavras com três ocorrências mínimas em todo o texto para consideração no treinamento. Com essa configuração, foi obtido um vocabulário com 2696 palavras ao total, mais que o dobro do anterior e com os mesmos parâmetros de treinamento.

Figura 4.5 – *Embedding* de 50 posições (subcaso 2).

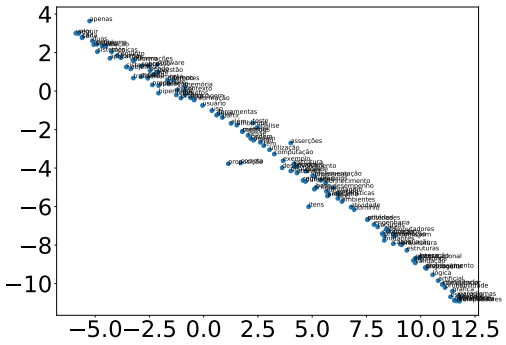
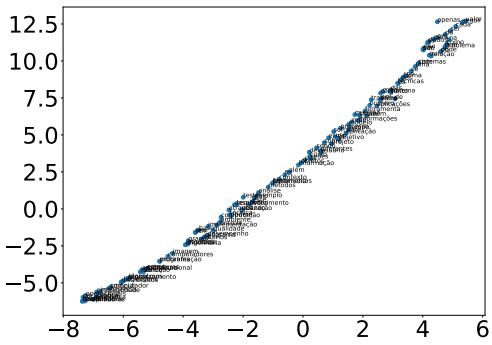


Figura 4.6 – *Embedding* de 300 posições (subcaso 2).



Na Figura 4.5 e Figura 4.6 podemos observar a distribuição entre alguns desses termos, que apresenta um comportamento diferente do que foi visto na subseção anterior. A visualização dos *embeddings* no plano bidimensional, para este subcaso, demonstra um comportamento diferente em relação aos outros subcasos, pois o fato de haver uma similaridade maior entre algumas palavras, elas ficaram próximas as outras formando uma linha de similaridade. O comportamento para esse conjunto de dados ocasionou na modelagem de outro subcaso, pois essa proximidade entre as palavras pode acabar afetando negativamente a classificação das questões, através dos *embeddings* treinados.

Figura 4.7 – Erro e acurácia - *Embedding* de 50 posições (subcaso 2).

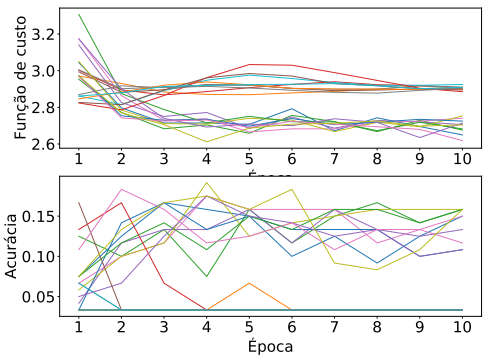
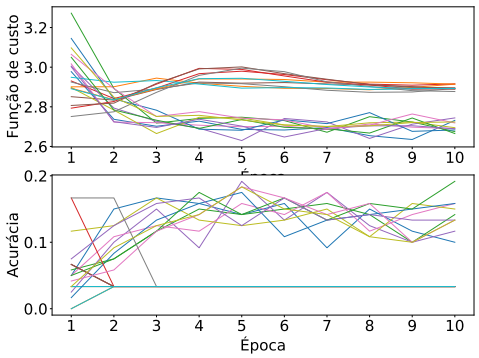


Figura 4.8 – Erro e acurácia - *Embedding* de 300 posições (subcaso 2).



Para este subcaso, dado o comportamento diferenciado observado na visualização

dos termos do *embedding*, os resultados para erro apresentaram um comportamento de estabilidade ao longo de todas as épocas (gráficos de cima das Figuras 4.7 e 4.8), assim como se manteve um comportamento aleatório para os valores de acurácia observados.

4.1.3 Subcaso 3: *Embedding* com texto de livros de computação

Neste subcaso, o texto das 150 questões selecionadas foi utilizado para formação dos *embeddings*, junto a um *corpus* personalizado de livros de computação. A lista com os 23 livros selecionados e seus respectivos tópicos está no Apêndice A. Para esse subcaso, foram consideradas 10 ocorrências mínimas por palavra para ser considerada no treinamento, devido a maior quantidade de dados. Com essa configuração, foi obtido um vocabulário com 18527 palavras ao total. Na Figura 4.9 e Figura 4.10 podemos observar a distribuição entre alguns desses termos e alguns agrupamentos, porém com algumas distinções do que foi observado na Subseção 4.1.1.

Figura 4.9 – *Embedding* de 50 posições (subcaso 3).

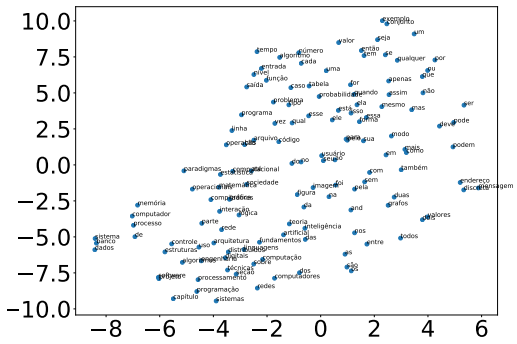
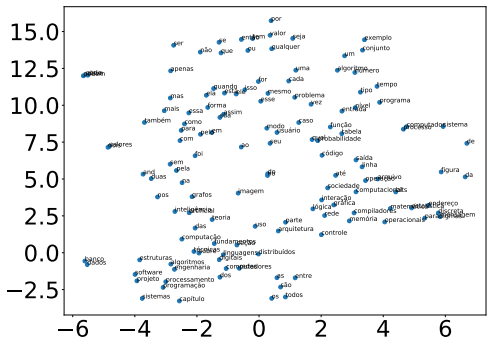


Figura 4.10 – *Embedding* de 300 posições (subcaso 3).



A visualização dos *embeddings* no plano bidimensional voltou a ter um aspecto semelhante ao do subcaso da Subseção 4.1.1, onde há um espalhamento das palavras, com alguns grupos formados. A diferença é que neste subcaso, os grupos de palavras parecem mais especializados, o que deve ser consequência do tamanho do vocabulário alcançado com os dados de treinamento, possibilitando maximizar a semelhança entre termos que comumente aparecem juntos.

Para este subcaso, é possível observar uma diminuição mais significativa no erro, na parte superior da Figura 4.11 e Figura 4.12, logo nas primeiras épocas de execução. Em relação à acurácia, ela ainda é baixa, mas apresenta um comportamento semelhante de subir lentamente, para as diversas execuções realizadas, ainda assim ficando na faixa entre 10% e 20%.

Figura 4.11 – Erro e acurácia - *Embedding* de 50 posições (subcaso 3).

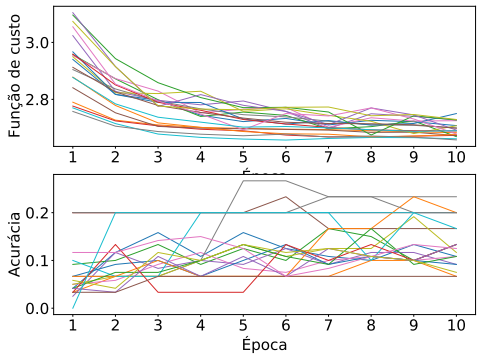
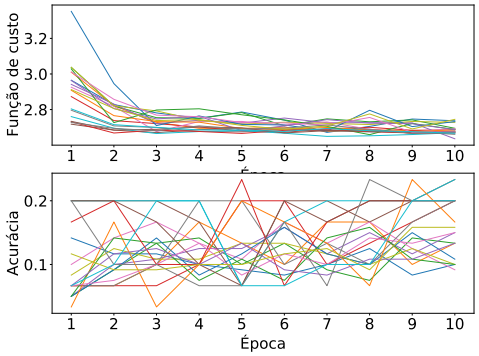


Figura 4.12 – Erro e acurácia - *Embedding* de 300 posições (subcaso 3).



4.2 TF-IDF E NAIVE BAYES

Para o classificador Naive Bayes, houve a utilização do algoritmo TF-IDF para transformação das entradas. Na sequência, foram executados os três subcasos de teste criados e armazenado os dados de acurácia geral e por tópico de classificação.

Figura 4.13 – Acertos e erros por tópico (NB - subcaso 1).

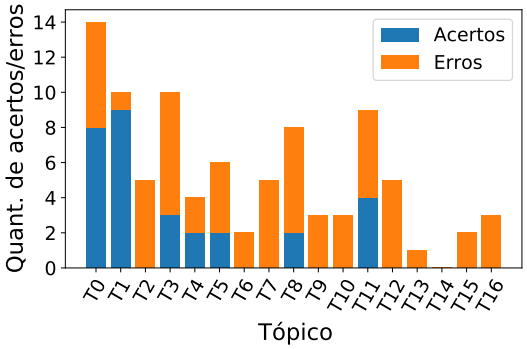
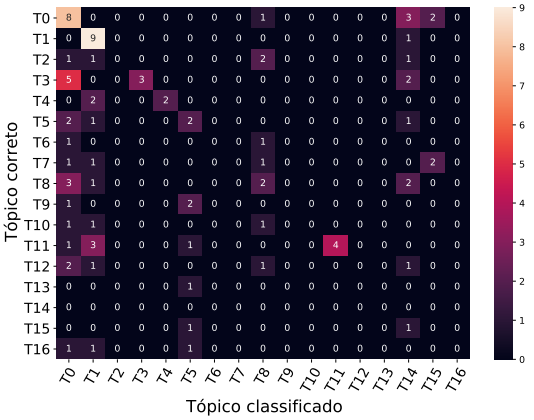


Figura 4.14 – Matriz de confusão (NB - subcaso 1).



Em relação ao subcaso 1, na Figura 4.13 é mostrada a quantidade de acertos e erros por tópico de classificação, sendo que os tópicos T0 e T1, que possuem as maiores quantidades de questões, apresentam as melhores acurácias. Para evidenciar os erros de classificação, é mostrada uma matriz de confusão, na Figura 4.14. A matriz indica que os tópicos que mais receberam indicações de classificação foram os tópicos T0, T1, T5, T8 e T14, sendo os dois primeiros com uma acurácia relativamente boa, enquanto nos outros a maioria das questões foi classificada erroneamente nestes tópicos.

Figura 4.15 – Acertos e erros por tópico (NB - subcaso 2).

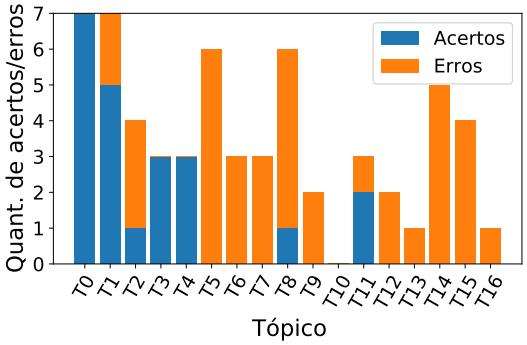
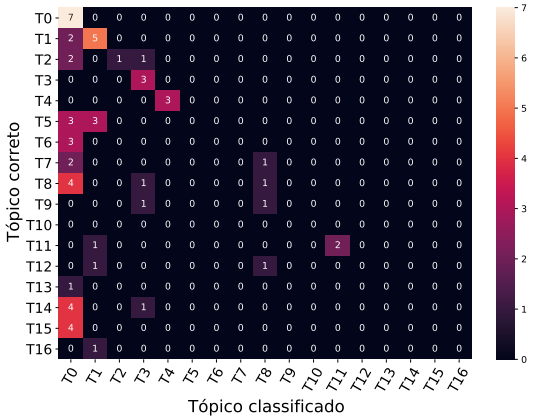


Figura 4.16 – Matriz de confusão (NB - subcaso 2).



Para o subcaso 2, a Figura 4.15 mostra a quantidade de acertos e erros por tópico de classificação, sendo que existem mais tópicos com uma acurácia boa (T0, T1, T3, T4 e T11), porém essa boa acurácia se concentra somente nesses casos, enquanto nos outros é zero ou próxima a este valor. A matriz de confusão, na Figura 4.14, indica uma concentração de questões classificadas erroneamente nos tópicos T0 e T1, mais uma vez evidenciando uma possível influencia da quantidade de questões que esses tópicos apresentam, em relação aos outros, na execução do classificador.

Figura 4.17 – Acertos e erros por tópico (NB - subcaso 3).

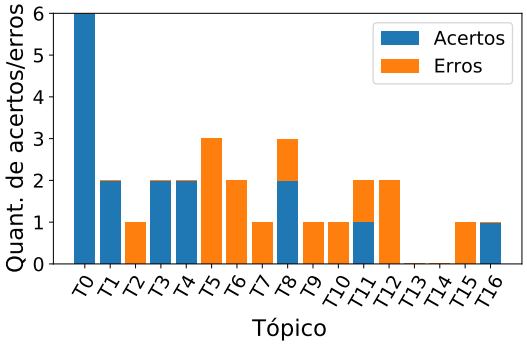
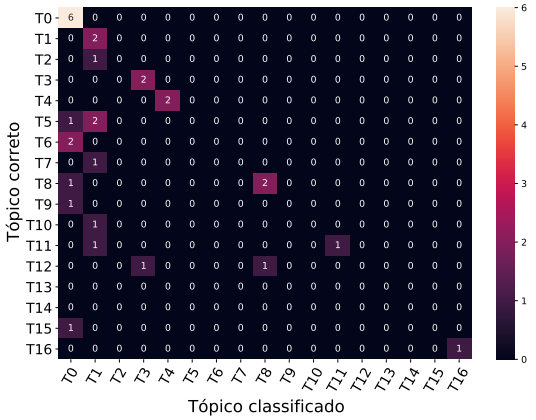


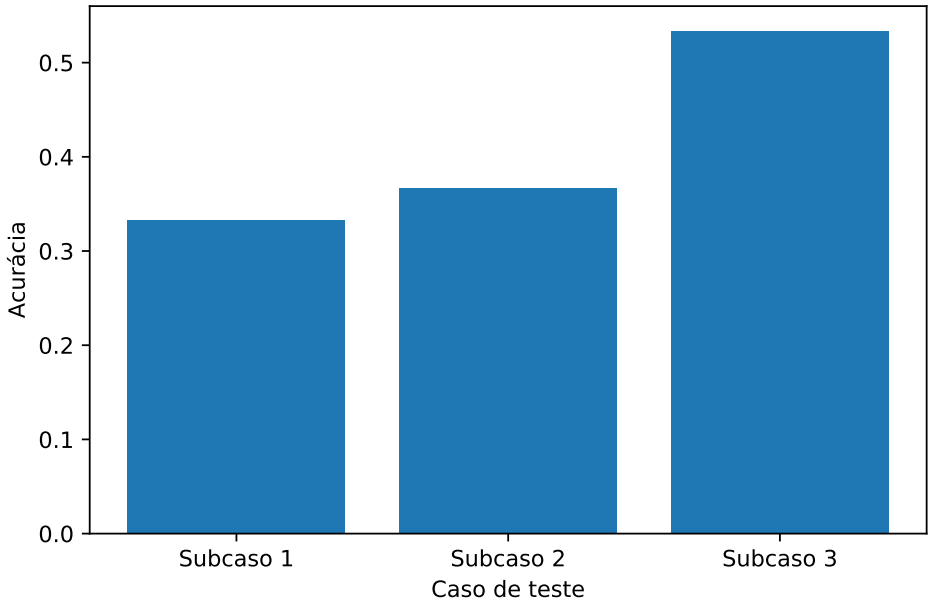
Figura 4.18 – Matriz de confusão (NB - subcaso 3).



No subcaso 3, que apresenta melhor acurácia em relação aos anteriores, houve uma boa distribuição de tópicos para classificação e que seguiu evidenciando uma melhor acurácia em tópicos específicos, dado a quantidade de acertos e erros por tópico, mostrada na Figura 4.17. Na matriz de confusão da Figura 4.18, segue o número de questões

classificadas erroneamente nos tópicos T0 e T1, porém em menor quantidade devido ao número de questões separado para validação neste subcaso.

Figura 4.19 – Acurácia por subcaso de teste (NB).



Os resultados de acurácia, para cada um dos subcasos, pode ser observado na Figura 4.19 e mostram um aumento na acurácia de acordo com o aumento do número de questões separadas para treinamento do modelo. A indicação do gráfico é de que com 60, 90 e 120 questões utilizadas para treinamento, para os casos 1, 2 e 3, respectivamente, há uma acurácia de 33%, 37% e 53%.

4.3 TF-IDF E SUPPORT VECTOR MACHINE

Para o classificador *Support Vector Machine*, também houve a utilização do algoritmo TF-IDF para a transformação das entradas. Na sequência, foram executados os três subcasos de teste criados e armazenados os dados de acurácia geral e por tópico de classificação.

Para o subcaso 1, utilizando SVM, a Figura 4.20 mostra que este é o subcaso que alcançou a maior quantidade de tópicos com acertos, conseguindo pelo menos uma questão classificada corretamente em dez tópicos. Em relação a matriz de confusão, na Figura 4.21, a maioria das questões classificadas no tópico errado se encontram nos tópicos T0, T14 e 15. Ainda que T14 (Teoria da Computação) e T15 (Teoria dos Grafos) representem uma quantidade baixa no total de questões selecionado, ambos apresentam alguma relação com o tópico T0 (Algoritmos e Estruturas de Dados) e acabaram herdando a classificação deste tópico em três questões, cada.

Figura 4.20 – Acertos e erros por tópico (SVM - subcaso 1).

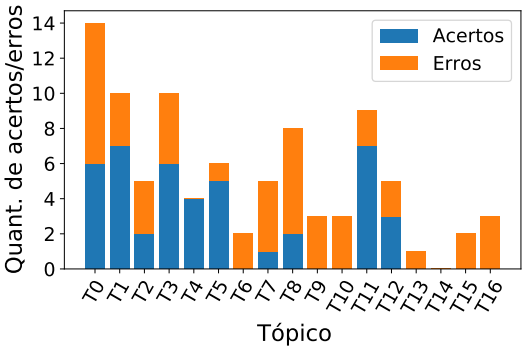


Figura 4.21 – Matriz de confusão (SVM - subcaso 1).

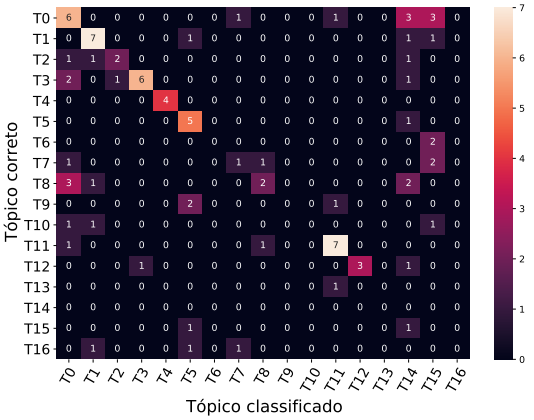


Figura 4.22 – Acertos e erros por tópico (SVM - subcaso 2).

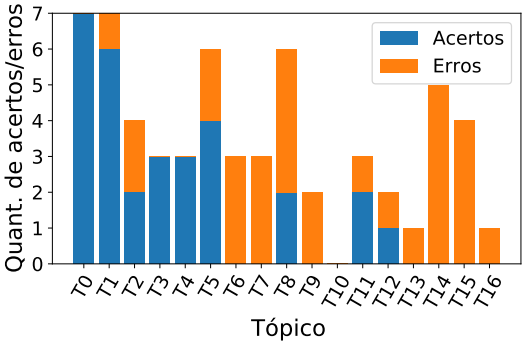
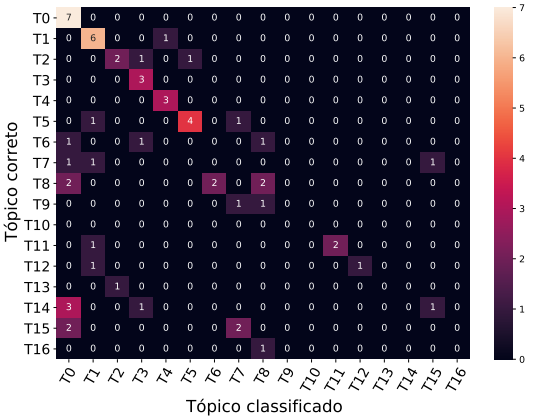


Figura 4.23 – Matriz de confusão (SVM - subcaso 2).



Para o subcaso 2, o comportamento de boa acurácia para os tópicos T0, T1, T3 e T4, visto na seção anterior, voltou a aparecer (Figura 4.22). Além desses tópicos, houve uma quantidade de acertos relevante para o tópico T5. A matriz de confusão, apresentada na Figura 4.23, mostra que a maioria das questões classificadas no tópico errado encontra-se no tópico T0, comportamento observado em boa parte dos casos e esperado devido ao grande número de questões pertencentes a este tópico, além de seu escopo ser algo presente em outros campos (Algoritmos e Estrutura de Dados).

No subcaso 3, o qual tem a melhor acurácia geral, a Figura 4.24 mostra também que esse subcaso possui a maior quantidade de tópicos com 100% de acurácia (T1, T3, T4, T7, T11, T12 e T16), enquanto os outros 2 subtópicos com acertos (T0 e T8) também apresentam boa acurácia. Para os outros restaram apenas erros ou sequer havia questões selecionadas para validação. A matriz de confusão, na Figura 4.25, mostra que os erros de

Figura 4.24 – Acertos e erros por tópico (SVM - subcaso 3).

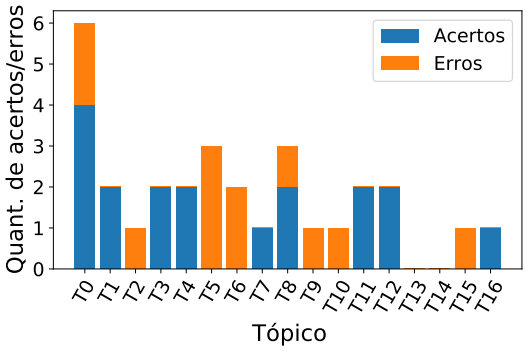
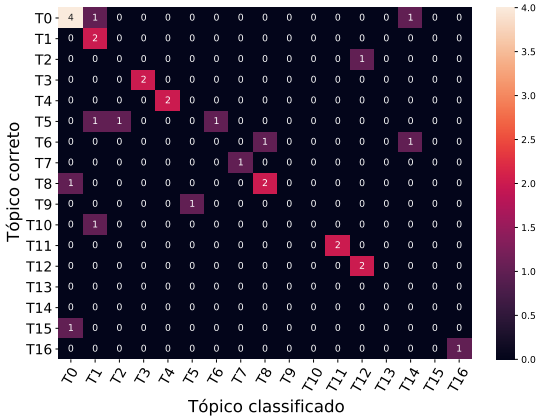
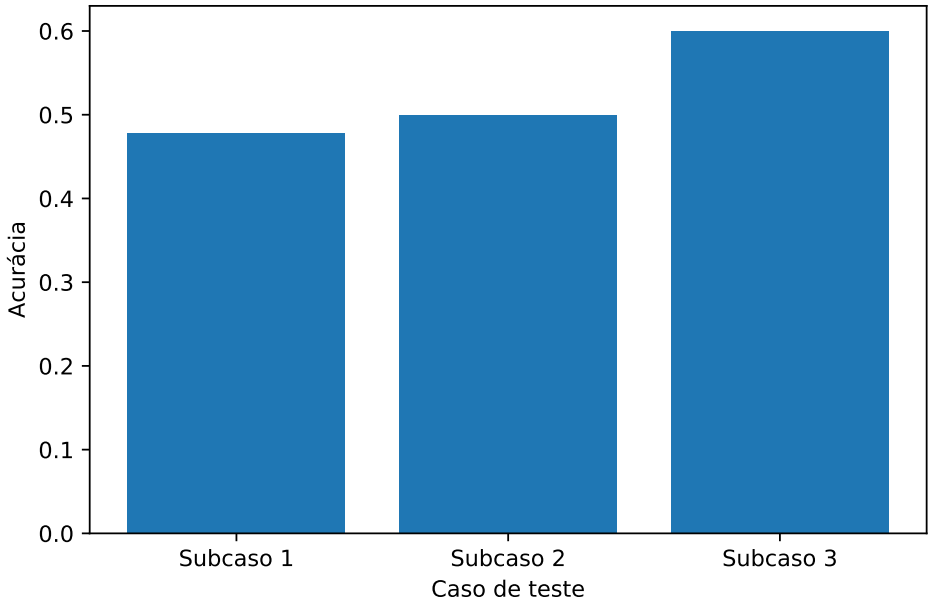


Figura 4.25 – Matriz de confusão (SVM - subcaso 3).



classificação foram bem distribuídos, sem concentração alguma em outro tópico ou algum tipo de padrão entre 2 ou mais tópicos.

Figura 4.26 – Acurácia por caso de teste (SVM).



Os resultados de acurácia, para cada um dos subcasos, pode ser observado na Figura 4.26. Assim como na Seção 4.2, a figura indica um aumento na acurácia de acordo com o aumento do número de questões separadas para treinamento do modelo. A indicação do gráfico é de que com 60, 90 e 120 questões utilizadas para treinamento, para os subcasos 1, 2 e 3, respectivamente, há uma acurácia de 48%, 50% e 60%. Porém, nesta seção, os valores são maiores para todos os subcasos, sendo essas as melhores acurácias para os três subcasos criados através da separação de dados, usados para execução

dos algoritmos NB, SVM e RF (*Random Forest*). Os resultados também superam qualquer acurácia alcançada com o algoritmo de Redes Neurais.

4.4 TF-IDF E RANDOM FOREST

Por fim, para o classificador *Random Forest*, também houve a utilização do algoritmo TF-IDF para a transformação das entradas. Na sequência, foram executados os três subcasos de teste criados e armazenados os dados de acurácia geral e por tópico de classificação.

Testes preliminares com um número reduzido de árvores (50 árvores) e com limitação da profundidade máxima da árvore (profundidade 5) eram executados rapidamente, porém apresentaram acurácia baixa, em torno de 20% para todos os subcasos. A alteração da quantidade de árvores para 500 e a eliminação do parâmetro de limitação da profundidade ocasionaram nos resultados mostrados na sequência, que apresentaram baixa elevação na acurácia (ainda é inferior a 40%) e elevaram o tempo de execução do algoritmo. Essa elevação no tempo de execução deve estar atrelada a formação de uma quantidade maior de árvores e com maior profundidade.

Figura 4.27 – Acertos e erros por tópico (RF - subcaso 1).

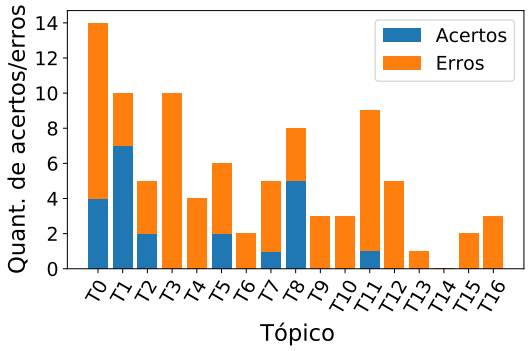
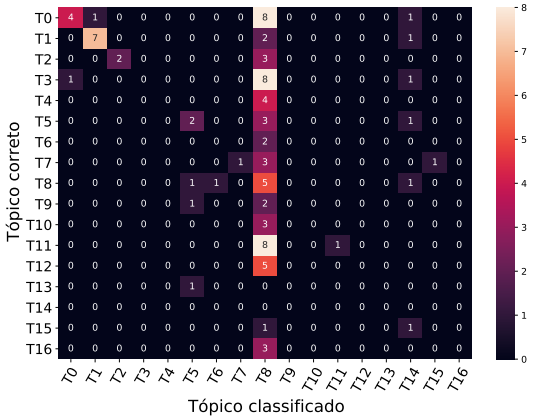


Figura 4.28 – Matriz de confusão (RF - subcaso 1).



Os resultados para o algoritmo RF foram bastante semelhantes para os três subcasos e podem ser comentados de maneira geral. Nas Figuras 4.27, 4.29 e 4.31, os gráficos de barras empilhadas indicam uma concentração maior de acertos nos tópicos T0, T1 e T8, sendo que no subcaso 2 (Figura 4.29) outros tópicos apresentam uma quantia significativa de acertos, devido ao maior número de questões para validação.

Em relação às matrizes de confusão, mostradas nas Figuras 4.28, 4.30 e 4.32, há indicação de um comportamento diferente das seções anteriores e um tanto inesperado,

Figura 4.29 – Acertos e erros por tópico (RF - subcaso 2).

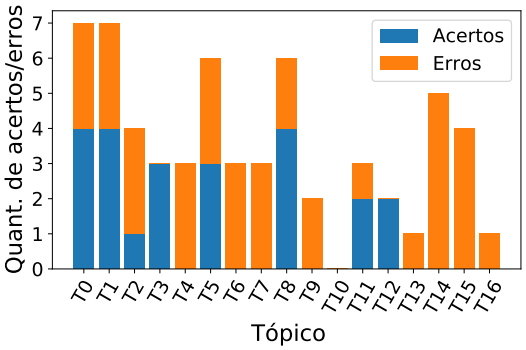


Figura 4.30 – Matriz de confusão (RF - subcaso 2).

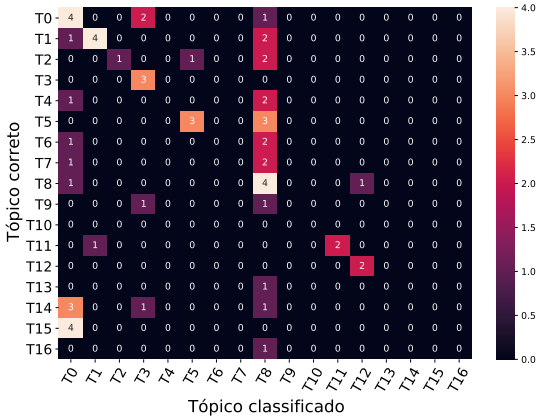


Figura 4.31 – Acertos e erros por tópico (RF - subcaso 3).

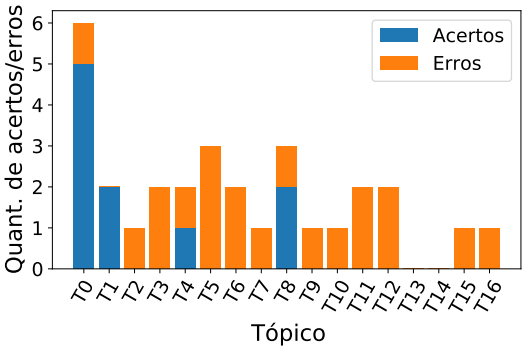
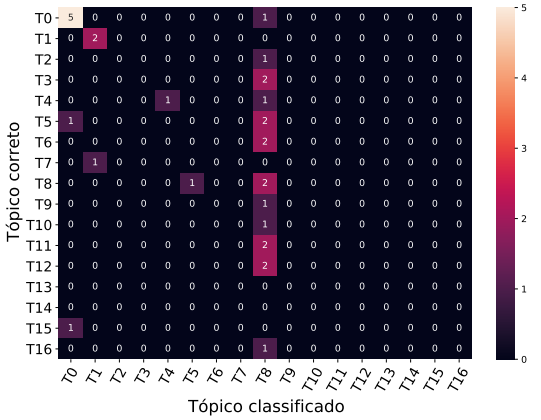


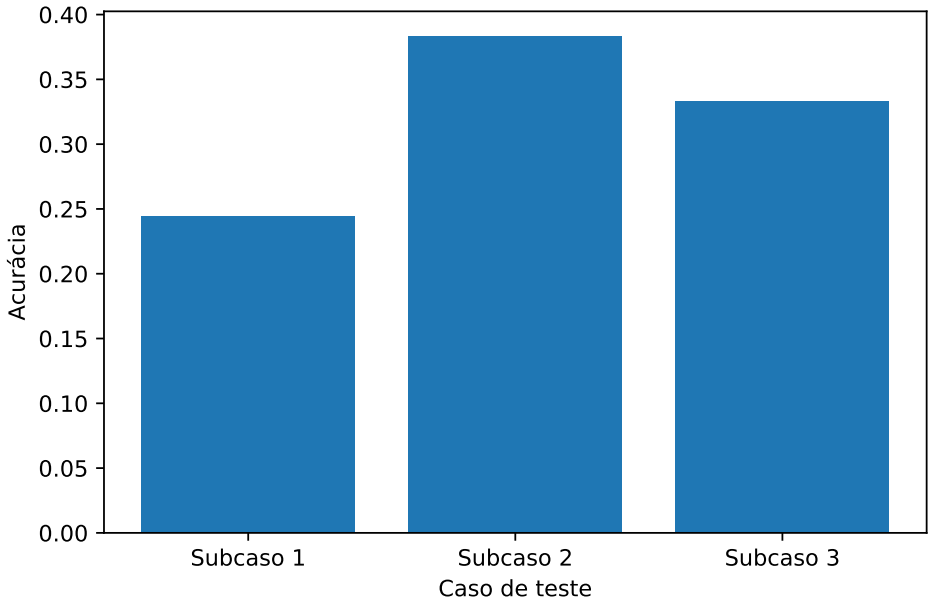
Figura 4.32 – Matriz de confusão (RF - subcaso 3).



dado que a maior concentração de questões classificadas no tópico errado concentram-se no tópico T8 (Lógica e Matemática Discreta). Em contraste com os subcasos da Seção 4.2 e Seção 4.3, os 3 subcasos executando RF realmente apresentaram uma melhor acurácia para o T8 por si só, porém isto pode estar atrelado ao fato do algoritmo ter classificado erroneamente boa parte as outras questões para este tópico.

Com relação a acurácia geral dos três subcasos, mostrada na Figura 4.33, foi observado um comportamento distinto em relação aos dois algoritmos anteriores. A melhor acurácia foi observada para o subcaso 2, que apresenta uma separação de dados de 60% e 40% para treinamento e validação, respectivamente. Desta vez, o aumento no número de questões separadas para treinamento no subcaso 3 resultou em uma acurácia menor (33%) em comparação ao subcaso 2 (38%) e subcaso 1 (24%). Em geral, resultados piores do que os vistos nas duas seções anteriores.

Figura 4.33 – Acurácia por caso de teste (RF).



4.5 VISÃO GERAL

Nesta seção serão tratadas algumas especificidades de algumas técnicas/algoritmos, além de breves comparações em relação ao que foi mostrado nas seções anteriores deste capítulo.

Com relação ao *embedding* e a visualização dele, é importante destacar a diferença entre os casos da Subseção 4.1.1 e Subseção 4.1.3, que inicia pela área do gráfico, dado que o vocabulário maior de palavras existente para o subcaso 3 faz com que a área de visualização seja maior, devido ao espalhamento das palavras. Outra diferença é que, enquanto nas Figuras 4.1 e 4.2 as palavras pertencentes aos tópicos de classificação encontram-se extremamente agrupadas, nas Figuras 4.9 e 4.10 esse grupos estão mais espalhados e especializados.

Outro ponto em relação ao *embedding* destas duas subseções é a diferença de especialização, devido a quantidade de dados utilizada para treinamento. No caso da Subseção 4.1.1, existem poucos grupos que incluem termos de computação, em geral (Figura 4.34). Por outro lado, no caso da Subseção 4.1.3, grupos como os dois mostrados na Figura 4.35 (atrelados a "*banco de dados*" e "*projeto de software*"), se encontram espalhados pelo espaço gerado.

Para a acurácia geral, a Tabela 4.1 traz os dados comparativos. Importante destacar que os subcasos para o algoritmo RN (Redes Neurais) são organizados de maneira distinta dos restantes. Para este caso, a adição de mais dados ao *embedding* não trouxe necessariamente um resultado melhor, principalmente para o subcaso 2, o que pode ser justificado pela não especialização dos dados usados em relação aos tópicos de classifica-

Figura 4.34 – Porção da visualização do *embedding* (subcaso 1).

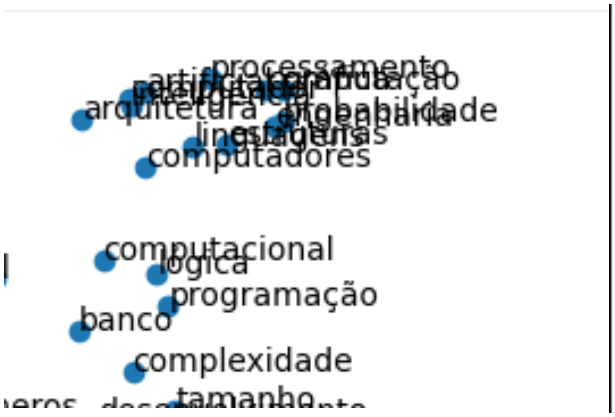


Figura 4.35 – Porção da visualização do *embedding* (subcaso 3).

ção. Já no subcaso 3, foram buscados livros diretamente relacionados aos tópicos, o que afetou a visualização do *embedding*, porém não ocasionou em aumento na acurácia.

Tabela 4.1 – Acurácia geral por subcaso de teste.

Algoritmo	Subcaso 1	Subcaso 2	Subcaso 3
RN*	20%	18%	27%
NB	33%	37%	53%
SVM	48%	50%	60%
RF	24%	38%	33%

Para os outros algoritmos, os melhores resultados foram atingidos com o algoritmo SVM. Além disso, o desempenho dos algoritmos RN e RF, no quesito de consumo de recursos computacionais, é pior em relação aos outros dois. A demora na execução da obtenção de resultados está atrelada também ao treinamento do *embedding*, via algoritmo Word2Vec, enquanto a demora na execução para o algoritmo RF foi descrita anteriormente, na Seção 4.4, e está atrelada à parametrização do algoritmo.

Outra informação importante é que as matrizes de confusão, mostradas para os algoritmos NB, SVM e RF não mostraram nenhum tipo de troca de classificação explícita entre dois tópicos de classificação. O que foi indicado é um comportamento esperado, onde existem muitas questões classificadas erroneamente em tópicos com uma maior quantidade de questões (maior quantidade de dados treinados) e um caso atípico no classificador RF, onde o algoritmo classificou majoritariamente no tópico T8, que tem uma quantidade intermediária de questões.

5 CONCLUSÃO

O PLN é uma área em crescimento que busca manipular e interpretar os dados ligados às linguagens usadas por humanos, seja em formato textual ou auditivo. No campo de dados textuais, há uma diversidade de problemas a serem explorados, entre eles a classificação e categorização de trechos de texto, dentro do contexto em que estes dados estão inseridos.

Os avanços nessa área indicam um futuro promissor para as necessidades no campo de classificação de texto. Porém, nesse contexto, muitos dados ainda seguem inexplorados, por mais que sua classificação possa contribuir para a avaliação desses algoritmos, além de favorecer o entendimento de alguns fatores relacionados a estes dados textuais.

O trabalho apresentou a aplicação de técnicas para processamento de dados textuais seguindo um fluxo de etapas proposto pela metodologia de KDD. Desde a utilização de técnicas frequentemente usadas para pré-processamento textual, junto a exploração de métodos para transformação e extração de atributos, além de algoritmos de classificação comumente usados, porém aplicados em um novo contexto: as questões das provas voltadas para o curso de Ciência da Computação do Enade.

Dentre os classificadores selecionados, a utilização de um modelo de Redes Neurais simples não foi eficiente ao realizar a categorização das questões. Apesar das tentativas de melhoria no treinamento do *embedding* (etapa anterior à classificação), com a utilização de dados extras e a mudança de comportamento desses vetores observada no plano 2D, a acurácia permaneceu baixa para todos os subcasos executados com esse algoritmo.

Por outro lado, técnicas clássicas na literatura, os algoritmos SVM, NB e RF obtiveram resultados intermediários de acurácia, porém se mostraram técnicas simples de serem aplicadas, principalmente ao considerar a extração de atributos realizada pelo algoritmo TF-IDF, em etapa prévia, que também é um algoritmo de simples e rápida execução. A sua acurácia intermediária pode estar atrelada a baixa quantidade de dados e, principalmente, a distribuição não-uniforme das questões acerca dos tópicos de classificação.

Ainda assim, por ser um trabalho exploratório, espera-se que o trabalho contribua para análises mais profundas dos algoritmos utilizados e dos possíveis gargalos, no escopo de acurácia, que estes classificadores sofreram considerando os dados em questão. Além disso, fica aberta a possibilidade de exploração de outros algoritmos e técnicas para classificação desses dados, que possam resultar em maior acurácia no processo de categorização.

REFERÊNCIAS BIBLIOGRÁFICAS

ABBAS, M. et al. Multinomial naive bayes classification model for sentiment analysis. **IJCSNS**, v. 19, n. 3, p. 62, 2019.

AZEVEDO, A.; SANTOS, M. Kdd, semma and crisp-dm: A parallel overview. In: . [S.l.: s.n.], 2008. p. 182–185.

BELLINGER, G.; CASTRO, D.; MILLS, A. **Data, information, knowledge, and wisdom**. 2004.

BLOOM, B. S.; KRATHWOHL, D. R.; MASIA, B. B. Bloom taxonomy of educational objectives. In: **Allyn and Bacon**. [S.l.]: Pearson Education, 1984.

CHARAO, A. et al. Explorando resultados por questão no enade em ciência da computação para subsidiar revisão de projeto pedagógico de curso. In: **Anais do XXVIII Workshop sobre Educação em Computação**. Porto Alegre, RS, Brasil: SBC, 2020. p. 16–20. ISSN 2595-6175. Disponível em: <https://sol.sbc.org.br/index.php/wei/article/view/11121>.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. **AI magazine**, v. 17, n. 3, p. 37–37, 1996.

____. The kdd process for extracting useful knowledge from volumes of data. **Communications of the ACM**, ACM New York, NY, USA, v. 39, n. 11, p. 27–34, 1996.

GORDON-RODRIGUEZ, E. et al. Uses and abuses of the cross-entropy loss: case studies in modern deep learning. **arXiv preprint arXiv:2011.05231**, 2020.

GRUS, J. **Data Science do zero: Primeiras regras com o Python**. [S.l.]: Alta Books, 2019.

HEATON, J. **Introduction to Neural Networks for Java, 2nd Edition**. 2nd. ed. [S.l.]: Heaton Research, Inc., 2008. ISBN 1604390085.

HO, T. K. Random decision forests. In: IEEE. **Proceedings of 3rd international conference on document analysis and recognition**. [S.l.], 1995. v. 1, p. 278–282.

HSU, C.-W.; LIN, C.-J. A comparison of methods for multiclass support vector machines. **IEEE transactions on Neural Networks**, IEEE, v. 13, n. 2, p. 415–425, 2002.

JAYAKODI, K.; BANDARA, M.; PERERA, I. An automatic classifier for exam questions in engineering: A process for bloom's taxonomy. In: **2015 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)**. [S.l.: s.n.], 2015. p. 195–202.

KHATTAK, F. K. et al. A survey of word embeddings for clinical text. **Journal of Biomedical Informatics: X**, v. 4, p. 100057, 2019. ISSN 2590-177X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2590177X19300563>.

KOWSARI, K. et al. Rmdl: Random multimodel deep learning for classification. In: **Proceedings of the 2nd International Conference on Information System and Data Mining**. [S.l.: s.n.], 2018. p. 19–28.

____. Text classification algorithms: A survey. **Information**, v. 10, n. 4, 2019. ISSN 2078-2489. Disponível em: <https://www.mdpi.com/2078-2489/10/4/150>.

LAI, S. et al. Recurrent convolutional neural networks for text classification. In: **Proceedings of the AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2015. v. 29, n. 1.

LENCI, A. Distributional models of word meaning. **Annual review of Linguistics**, Annual Reviews, v. 4, p. 151–171, 2018.

LIMA, P. d. S. N. et al. Sysenade - análise das questões de provas do enade organizadas pelos temas abordados. **Anais dos Workshops do Congresso Brasileiro de Informática na Educação**, v. 7, n. 1, 2018. Disponível em: <https://br-ie.org/pub/index.php/wcbie/article/view/8267>.

____. Análise de dados do enade e enem: uma revisão sistemática da literatura. **Avaliação: Revista da Avaliação da Educação Superior (Campinas)**, scielo, v. 24, p. 89 – 107, 05 2019. ISSN 1414-4077. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1414-40772019000100089&nrm=iso>.

MAATEN, L. Van der; HINTON, G. Visualizing data using t-sne. **Journal of machine learning research**, v. 9, n. 11, 2008.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

MOHAMMED, M.; OMAR, N. Question classification based on blooms taxonomy cognitive domain using modified tf-idf and word2vec. **PLOS ONE**, Public Library of Science, v. 15, n. 3, p. 1–21, 03 2020. Disponível em: <https://doi.org/10.1371/journal.pone.0230442>.

PARDO, T. A. S.; NUNES, M. d. G. V. **A Construção de um Corpus de Textos Científicos em Português do Brasil e sua Marcação Retórica**. [S.l.], 2003.

PROVOST, F.; FAWCETT, T. Data science and its relationship to big data and data-driven decision making. **Big data**, Mary Ann Liebert, Inc. 140 Huguenot Street, 3rd Floor New Rochelle, NY 10801 USA, v. 1, n. 1, p. 51–59, 2013.

QEDU. **O que são microdados? | Academia QEdu**. 2020. <https://academia.qedu.org.br/glossario/o-que-sao-microdados/>. Accessed: 2020-10-24.

QIU, X. et al. Pre-trained models for natural language processing: A survey. **arXiv preprint arXiv:2003.08271**, 2020.

RAJARAMAN, A.; ULLMAN, J. D. **Mining of Massive Datasets**. USA: Cambridge University Press, 2011. ISBN 1107015359.

SANGODIAH, A.; MUNIANDY, M.; HENG, L. Question classification using statistical approach: A complete review. **Journal of Theoretical and Applied Information Technology**, v. 71, p. 386–395, 01 2015.

SHARMA, S. Activation functions in neural networks. **Towards Data Science**, v. 6, 2017.

SILVA, V. A.; BITTENCOURT, I. I.; MALDONADO, J. C. Automatic question classifiers: A systematic review. **IEEE Transactions on Learning Technologies**, v. 12, n. 4, p. 485–502, 2019.

SINGH, G. et al. Comparison between multinomial and bernoulli naive bayes for text classification. In: IEEE. **2019 International Conference on Automation, Computational and Technology Management (ICACTM)**. [S.l.], 2019. p. 593–596.

THEODORSON, G. A.; THEODORSON, A. G. A modern dictionary of sociology. 1969.

TSUMURA, M. et al. **Obtaining data from unstructured data for a structured data collection**. [S.l.]: Google Patents, 2016. US Patent 9,299,041.

TUFFÉRY, S. **Data mining and statistics for decision making**. 1st. ed. [S.l.]: Wiley Publishing, 2011. ISBN 0470688297.

ZHOU, C. et al. A c-lstm neural network for text classification. **arXiv preprint arXiv:1511.08630**, 2015.

APÊNDICE A – LISTA DE LIVROS UTILIZADOS PARA TREINAMENTO DO *EMBEDDING* E CÓDIGO DO TÓPICO RELACIONADO

- Cormen, T. H. et al. Algoritmos: Teoria e Prática. (T0)
- Goodrich, M. T. et al. Projeto de Algoritmos: Fundamentos, análise e exemplos da internet. (T0)
- Stallings, W. Arquitetura e Organização de Computadores. (T1)
- Tanenbaum, A. S. et al. Sistemas Operacionais: Projeto e Implementação. (T1)
- Machado, F. N. R. Banco de Dados: Projeto e Implementação. (T2)
- Date, C. J. Introdução a Sistemas de Bancos de Dados. (T2)
- Alexandre, E. de S. M. Introdução aos Compiladores. (T3)
- Saúde, A. V. Computação Gráfica e Processamento de Imagens. (T4)
- Filho, O. M. et al. Processamento Digital de Imagens. (T4)
- de Morais, E. M. et al. Interação humano-computador. (T5)
- Pressman, R. S. Engenharia de Software: Uma Abordagem Profissional. (T5)
- Lima, J. M. C. Informática na Sociedade e Ética. (T6)
- Aguilar, L. J. Fundamentos de Programação: Algoritmos, estruturas de dados e objetos. (T7)
- de Oliveira, R. F. Inteligência Artificial. (T8)
- Gomide, A. et al. Elementos de Matemática Discreta para Computação. (T9)
- Tucker, A. B. et al. Linguagens de Programação: Princípios e Paradigmas. (T10)
- Departamento de Estatística UFPB. Cálculo das Probabilidades e Esatística I. (T11)
- Forouzan, B. A. Comunicação de Dados e Redes de Computadores. (T12)
- Tanenbaum, A. S. et al. Redes de Computadores. (T12)
- Floyd, T. Sistemas Digitais: Fundamentos e Aplicações. (T13)
- Coulouris, G. et al. Sistemas Distribuídos: Conceito e Projeto. (T14)
- Sipser, M. Uma Introdução à Teoria da Computação. (T15)
- Feofiloff, P. et al. Uma Introdução Sucinta à Teoria dos Grafos. (T16)