Desenvolvimento de um Modelo de Inteligência Artificial com Scikit-Learn para Classificação de Atividades Complementares de Curso

Pedro Henrique Southier[1], **Árton Pereira Dorneles**[1]

[1]Curso de Bacharelado em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia Farroupilha (IFFar) – Frederico Westphalen – RS – Brasil

phsouthier@hotmail.com, arton.dorneles@iffarroupilha.edu.br

Abstract. *This work presents the development and evaluation of an artificial intelligence model for the automatic classification of Complementary Course Activity certificates. The approach employs supervised learning algorithms implemented in Python using the Scikit-learn library, evaluated on a pre-labeled corpus of certificates from the Bachelor's degree program in Computer Science at IFFar/FW. The experiments indicate accuracy above 70% for all models, with the Neural Network achieving 82.54%, demonstrating the potential of the solution as a tool to support academic management.*

Resumo. *Este trabalho apresenta o desenvolvimento e a avaliação de um modelo de inteligência artificial para a classificação automática de certificados de Atividades Complementares de Curso. A abordagem utiliza algoritmos de aprendizado supervisionado implementados em Python com a biblioteca Scikit-learn, avaliados a partir de um corpus pré-rotulado de certificados do curso de Bacharelado em Ciência da Computação do IFFar/FW. Os experimentos indicam acurácia superior a 70% para todos os modelos, com destaque para a Rede Neural, que alcançou 82,54%, demonstrando o potencial da solução como ferramenta de apoio à gestão acadêmica.*

1. Introdução

Com o avanço da transformação digital nos últimos anos, a quantidade de documentos digitais disponíveis em formatos como PDF cresce de forma acelerada em ambientes acadêmicos, administrativos e corporativos. Nesse contexto, a identificação e classificação manual das informações contidas nesses documentos torna-se uma tarefa cada vez mais trabalhosa, demandando tempo, esforço e estando sujeita a erros e inconsistências. Além disso, a tarefa apresenta maior complexidade quando os documentos possuem diferentes formatos, layouts ou níveis variados de qualidade na digitalização, uma vez que isso aumenta o risco de falhas na identificação.

O processo de extração de dados de documentos envolve a identificação e captura de informações relevantes para cada aplicação, tais como nomes, datas, instituições emissoras, descrições de atividades e outros dados pertinentes. Esses elementos são essenciais para o correto registro, organização e validação das informações em sistemas automatizados, possibilitando, entre outras possíveis aplicações, a categorização de documentos, a verificação da autenticidade, a alimentação de bancos de dados e o suporte a processos de tomada de decisão.

Considerando esse cenário, torna-se evidente a necessidade de soluções automáticas que possam extrair e classificar informações com maior agilidade e precisão. Nesse contexto, técnicas de inteligência artificial podem ser utilizadas para reduzir erros humanos, a otimização dos fluxos de trabalho e a melhoria na padronização dos resultados. Entre essas técnicas, destacam-se os métodos baseados em aprendizado de máquina e processamento de linguagem natural, que são capazes de identificar padrões, interpretar o conteúdo dos documentos e extrair dados relevantes de forma consistente.

De modo geral, essas soluções utilizam algoritmos tradicionais de aprendizado supervisionado, como Regressão Logística, Máquinas de Vetores de Suporte (SVM), Naive Bayes, *k-Nearest Neighbors* (k-NN), *Random Forest* e Redes Neurais (MLP) (HASTIE et al., 2009). Além desses métodos, há também arquiteturas baseadas em redes neurais profundas, incluindo Redes Neurais Convolucionais (CNNs) e Redes Neurais Recorrentes (RNNs), particularmente eficazes para o processamento de dados mais complexos e sequenciais (GOODFELLOW et al., 2016). Todos esses métodos exigem o treinamento de modelos com dados previamente rotulados, permitindo que eles aprendam a identificar padrões e a associar entradas a categorias específicas, buscando maximizar a precisão das classificações. A eficácia dessas soluções, por sua vez, depende diretamente da qualidade e representatividade dos dados usados no treinamento, bem como de uma escolha adequada de parâmetros dos algoritmos.

No contexto acadêmico brasileiro, os certificados de atividades complementares de curso desempenham um papel relevante, pois são utilizados para comprovar a participação de estudantes em eventos, cursos, projetos e outras atividades extracurriculares exigidas pelos currículos de cursos superiores. Esses certificados são frequentemente disponibilizados em formato digital e como documentos PDF, e precisam ser analisados e validados pelas instituições de ensino. A realização manual desse processo pode se tornar onerosa e suscetível a inconsistências quando envolve um grande volume de documentos.

Diante dessa realidade, este trabalho propõe o desenvolvimento de um modelo de inteligência artificial para a classificação automática de certificados de Atividades Complementares de Curso, baseado em algoritmos de aprendizado supervisionado implementados em Python por meio da biblioteca Scikit-Learn. A proposta visa apoiar sistemas de gestão acadêmica, contribuindo para a redução do esforço manual e para a melhoria da padronização e confiabilidade da análise de certificados.

Dessa forma, o objetivo deste trabalho é desenvolver e avaliar um modelo de classificação automática de certificados de Atividades Complementares de Curso, a partir de documentos digitais em formato PDF, utilizando algoritmos de aprendizado supervisionado. Para atingir esse objetivo, são avaliados diferentes algoritmos por meio de conjuntos de experimentos computacionais, empregando um corpus composto por certificados reais, previamente rotulados, arquivados no curso de Bacharelado em Ciência da Computação do IFFar, Campus Frederico Westphalen.

O restante deste trabalho está organizado como segue. Na Seção 2 são apresentados os fundamentos teóricos e trabalhos relacionados. Na Seção 3 é apresentada a metodologia do trabalho, bem como a delimitação de escopo da pesquisa. A Seção 4 apresenta os experimentos computacionais e principais resultados da implementação. Finalmente, na Seção 5, são apresentadas considerações finais e alternativas de trabalhos futuros.

2. Referencial Teórico

Nesta seção, são expostos os conceitos fundamentais e as tecnologias essenciais para a compreensão do estudo, além de serem discutidos trabalhos relacionados.

2.1. Atividades Complementares de Curso no IFFar/FW

As Atividades Complementares de Curso (ACCs) correspondem a um conjunto de práticas extracurriculares que vão além da matriz curricular obrigatória das disciplinas do curso. Tais atividades são frequentemente exigidas como componente obrigatório para a integralização do curso superior, contribuindo para a formação acadêmica do estudante, conforme diretrizes adotadas por diversas instituições de ensino no Brasil.

Particularmente, nos cursos de graduação oferecidos pelo Instituto Federal Farroupilha (IFFar), Campus Frederico Westphalen, estão estabelecidas múltiplas categorias de Atividades Complementares de Curso (ACCs), que abrangem diferentes dimensões da formação acadêmica, como ensino, pesquisa, extensão, estágios, dentre outras. Especificamente, no curso de Ciência da Computação, existem 20 categorias distintas, cada uma com limites máximos de carga horária estabelecidos no Projeto Pedagógico do Curso (PPC), de modo a incentivar o estudante a diversificar suas experiências formativas. Além disso, o PPC define uma carga horária total mínima obrigatória de 320 horas de ACCs para a integralização do curso, sendo este um requisito essencial para a colação de grau (INSTITUTO FEDERAL FARROUPILHA, 2022).

Nesse contexto, a gestão dessas atividades pode se tornar um processo complexo e trabalhoso tanto para os alunos quanto para os coordenadores, especialmente quando realizada de forma manual. Com o objetivo de tornar esse processo mais eficiente no IFFar, campus Frederico Westphalen, SOUTHIER e DORNELES (2022) desenvolveram o Sistema Integrado de Validação de Atividades Complementares (SIVAC), um sistema web baseado no framework Django que possibilita a submissão digital de certificados pelos estudantes e a validação ágil por parte dos coordenadores, promovendo uma administração mais organizada e eficaz das ACCs. A partir desse projeto, os cursos de Bacharelado em Ciência da Computação e Técnico em Informática Integrado do IFFar/FW constituíram uma base com cerca de 1000 certificados enviados ao SIVAC e classificados manualmente em categorias distintas. Essa base representa o conjunto total de dados disponíveis, a partir do qual foi selecionado o corpus específico que será utilizado como fonte de dados para o desenvolvimento do presente trabalho.

Em complemento a esse sistema, KIRSCH e DORNELES (2023), utilizaram a base de dados do SIVAC para desenvolver uma ferramenta em Python voltada ao reconhecimento de entidades nomeadas em certificados de ACCs. A avaliação da ferramenta foi realizada a partir do Corpus de Dados do SIVAC (CDS), composto por aproximadamente 530 certificados, a maioria em formato PDF e abrangendo diferentes tipos, como certificados de cursos, eventos, palestras e estágios, dos quais, após a aplicação de critérios de seleção, apenas 430 certificados foram utilizados nos experimentos. Os resultados indicaram que a ferramenta apresentou desempenho satisfatório na extração de entidades como nomes, títulos, períodos e carga horária, evidenciando o interesse e a viabilidade de abordagens automatizadas no processamento de certificados.

2.2. Fundamentos de Inteligência Artificial e Aprendizado de Máquina

A Inteligência Artificial (IA) é um campo da ciência da computação que estuda agentes capazes de perceber seu ambiente e agir racionalmente para atingir objetivos. A IA abrange diversas áreas, como percepção, raciocínio, representação do conhecimento, aprendizado, tomada de decisão e ação, sendo amplamente aplicada em tarefas específicas como diagnósticos médicos, jogos, robótica e direção autônoma de veículos. Dentre seus subcampos, destaca-se o aprendizado de máquina, que permite que sistemas melhorem seu desempenho a partir de dados, sem serem explicitamente programados para cada situação (RUSSELL; NORVIG, 2013).

O aprendizado de máquina pode ser classificado em três abordagens principais: supervisionado, não supervisionado e por reforço. No aprendizado supervisionado, o sistema aprende a partir de um conjunto de dados rotulados, isto é, exemplos em que as entradas e suas respectivas saídas são conhecidas. Já no não supervisionado, o objetivo é descobrir estruturas ou padrões ocultos em dados não rotulados. Por fim, o aprendizado por reforço é baseado na interação do agente com o ambiente, recebendo recompensas ou penalidades de acordo com suas ações (RUSSELL; NORVIG, 2013).

O desenvolvimento de um modelo de aprendizado supervisionado, frequentemente aplicado em tarefas de classificação, compreende as seguintes etapas principais: (i) coleta dos dados, em que os dados brutos são obtidos; (ii) pré-processamento, com limpeza, normalização e transformação dos dados; (iii) treinamento do modelo com os dados rotulados; (iv) avaliação do desempenho do modelo com métricas apropriadas; e (v) implantação, na qual o modelo é utilizado em um ambiente real.

Alguns exemplos de algoritmos de classificação utilizados em problemas supervisionados de aprendizagem de máquina incluem a Regressão Logística, amplamente utilizada como baseline em tarefas de classificação textual; as Árvores de Decisão, que oferecem interpretabilidade, embora possam sofrer com sobreajuste; o Random Forest, que combina múltiplas árvores de decisão; as Máquinas de Vetores de Suporte (SVM), frequentemente empregadas em dados de alta dimensionalidade, como textos; e o *k-Nearest Neighbors* (k-NN), um método baseado em instâncias, sensível à escolha da métrica de distância. Cada um desses algoritmos possui características que os tornam mais indicados conforme a natureza dos dados e os objetivos específicos do problema a ser resolvido.

A avaliação do desempenho de um modelo de classificação utiliza métricas como acurácia, precisão, revocação (*recall*) e medida F1 (F1-*score*). A acurácia representa a proporção de acertos sobre o total de previsões. A precisão representa a proporção de instâncias corretamente classificadas como positivas entre todas aquelas que o modelo classificou como positivas. O recall refere-se à proporção de instâncias positivas corretamente identificadas entre todas as instâncias que de fato pertencem à classe positiva. Já a medida F1 é a média harmônica entre precisão e revocação, sendo particularmente relevante em cenários com desequilíbrio entre as classes, pois proporciona uma avaliação mais equilibrada do desempenho do modelo (SOKOLOVA; LAPALME, 2009).

2.3. Python

Python é uma linguagem de programação de alto nível, interpretada e de propósito geral, amplamente reconhecida por sua sintaxe clara, concisa e intuitiva. Sua ênfase na legibi-

lidade do código e na redução do custo de manutenção a torna especialmente acessível tanto para iniciantes quanto para desenvolvedores experientes.

A linguagem também conta com uma extensa variedade de bibliotecas nativas e de terceiros, as quais oferecem suporte ao desenvolvimento dos mais diversos tipos de projetos. Nesse contexto, Python é amplamente utilizado em áreas como desenvolvimento web, automação de tarefas, análise e visualização de dados, inteligência artificial, aprendizado de máquina, computação científica, aplicações desktop, entre outras.

Além disso, por ser um software livre e de código aberto, seu código-fonte pode ser estudado, modificado e redistribuído livremente. Esse modelo promove o fortalecimento de uma comunidade global ativa e colaborativa, que contribui continuamente para a evolução da linguagem e a expansão de seu ecossistema (PYTHON SOFTWARE FOUNDATION, 2025).

2.4. Scikit-Learn

Scikit-learn é uma biblioteca de aprendizado de máquina desenvolvida em Python, amplamente utilizada para a implementação de algoritmos supervisionados e não supervisionados, como regressão, classificação, agrupamento, redução de dimensionalidade e seleção de características. Sua interface é simples e consistente, o que facilita a implementação, além de permitir a criação de modelos eficientes com poucas linhas de código.

Além de sua diversidade de algoritmos, a Scikit-learn também se destaca por oferecer ferramentas para avaliação e validação de modelos, como cross-validation, métricas de desempenho, ajuste de hiperparâmetros e pipelines para automação de fluxos de trabalho. Esses recursos tornam o processo de modelagem mais robusto e confiável, contribuindo para a reprodutibilidade dos experimentos e a padronização das etapas de treinamento e teste de modelos (SCIKIT-LEARN, 2025).

No contexto de tarefas de classificação supervisionada, a biblioteca reúne diferentes famílias de algoritmos, que variam desde modelos probabilísticos e lineares até métodos baseados em árvores, ensembles e redes neurais. Entre os classificadores tradicionalmente empregados em problemas de classificação textual, destacam-se neste estudo os métodos Naive Bayes, Arvores de Decisão, SVM Linear, Redes Neurais, Regressão Logística e *Random Forest*, selecionados por permitirem testar abordagens distintas entre si, variando desde modelos probabilísticos e lineares até métodos baseados em árvores, ensembles e redes neurais.

O Naive Bayes, implementado pela classe `MultinomialNB`, é um modelo probabilístico baseado no teorema de Bayes, que considera a frequência dos termos e assume independência entre as *features*. É rápido, simples e especialmente eficaz em classificação de textos devido ao bom desempenho em dados esparsos.

As Árvores de Decisão, implementada pela classe `DecisionTreeClassifier`, constrói um conjunto hierárquico de divisões no espaço de atributos, escolhendo em cada nó a feature que melhor separa as classes. É interpretável, mas pode sofrer sobreajuste em problemas de alta dimensionalidade, como aqueles derivados de vetores TF-IDF.

O SVM Linear, implementado via `LinearSVC`, determina hiperplanos lineares que maximizam a margem entre as diferentes classes. É bastante adequado para textos,

pois lida muito bem com uma grande quantidade de features e costuma oferecer excelente desempenho em classificações multiclasse.

As Redes Neurais, implementadas com `MLPClassifier`, fornecem uma rede neural *feed-forward* com camadas densas que ajusta pesos por retropropagação. É capaz de modelar relações não lineares entre os termos, embora demande maior tempo de treinamento e sensibilidade aos hiperparâmetros.

A Regressão Logística, acessada via `LogisticRegression`, é um método de classificação linear que estima probabilidades das classes. É muito eficiente em dados esparsos e de alta dimensionalidade, apresenta bom desempenho geral e permite regularização para evitar sobreajuste.

Finalmente, o método *Random Forest*, implementado via `RandomForestClassifier`, é um método de *ensemble* que combina diversas árvores de decisão treinadas sobre subconjuntos de dados e atributos. Reduz significativamente o risco de sobreajuste e melhora a generalização, embora cada árvore individual não seja ideal para dados altamente esparsos.

2.5. Trabalhos Relacionados

Nesta seção são discutidos trabalhos que abordam problemas semelhantes ao deste estudo, com foco na classificação automática de documentos por meio de técnicas de inteligência artificial, especialmente em contextos institucionais e administrativos.

CORREIA DA SILVA et al. (2018) destacam que o sistema judiciário brasileiro recebe diariamente um volume extremamente elevado de processos, muitos dos quais são encaminhados como um único arquivo PDF contendo múltiplos documentos distintos. Cada um desses documentos precisa ser analisado individualmente para ser associado a etiquetas relevantes e, posteriormente, encaminhado à equipe responsável. Nesse contexto, os autores coletaram uma base de 6.814 documentos do Supremo Tribunal Federal, manualmente rotulados em seis categorias, e aplicaram uma rede neural convolucional simples para automatizar a etapa de classificação. O modelo alcançou 90,35 % de acurácia, evidenciando a eficácia das redes neurais convolucionais na identificação de características relevantes em documentos digitalizados, mesmo diante de variações e imperfeições na qualidade dos textos.

De forma complementar, NOGUTI et al. (2020) ressaltam que, nos últimos anos, a aplicação de técnicas de Processamento de Linguagem Natural (PLN) em documentos jurídicos tem se intensificado. O estudo teve como foco a automatização da alocação de petições às suas respectivas áreas do Direito para reduzir custos e tempo de análise. Para isso, foi proposto o uso de técnicas de PLN para classificação textual, de modo a categorizar as descrições dos serviços prestados pelo Ministério Público do Estado do Paraná, enquadrando-as em uma das áreas do Direito abrangidas pela instituição. Foram comparadas diferentes abordagens de representação textual, incluindo matrizes termo-documento e embeddings, com ênfase no Word2Vec. Também foram avaliados três modelos de classificação: modelos lineares, árvores de decisão aprimoradas (*boosted trees*) e redes neurais. Os melhores resultados foram alcançados pela combinação de Word2Vec, treinado em um corpus específico do domínio jurídico, com redes neurais recorrentes do tipo LSTM, atingindo 90% de acurácia na classificação de dezoito categorias distintas.

No contexto da administração pública, CARVALHO (2024) propõe duas abordagens complementares para a classificação automática de documentos sensíveis, com o objetivo de aumentar a segurança e a transparência na gestão documental. A primeira é baseada em técnicas convencionais de PLN e expressões regulares, destacando-se o modelo BERT que atingiu 94% de acurácia na identificação de documentos sensíveis. A segunda abordagem recorre à recuperação de informação baseada em conteúdo visual (CBIR) para cenários onde a extração de texto não é viável, classificando-os a partir de características de imagem e alcançando 87% de acurácia.

Esses trabalhos evidenciam o potencial das técnicas de aprendizado de máquina e PLN aplicados na classificação documental em diferentes domínios institucionais, servindo como base metodológica para o desenvolvimento da solução proposta neste estudo.

3. Metodologia

Esta seção descreve os procedimentos metodológicos adotados no desenvolvimento da pesquisa, incluindo a definição do escopo dos certificados analisados, a conversão dos documentos em formato PDF para texto e o treinamento e a avaliação dos modelos de classificação supervisionada.

3.1. Delimitação do Escopo dos Certificados Analisados

Devido à variedade de formatos e estruturas presentes nos certificados reais que compõem o corpus utilizado nesta pesquisa, torna-se necessária a delimitação clara do escopo de análise. Desta forma, esse estudo está restrito à análise de certificados de atividades complementares arquivados no curso de Bacharelado em Ciência da Computação do Instituto Federal Farroupilha (IFFar), Campus Frederico Westphalen. O conjunto de dados é composto por certificados previamente rotulados em 20 categorias numeradas de 1 a 20, emitidos pelo próprio IFFar e por diversas outras instituições públicas e privadas.

Para esse estudo, os certificados pertencentes a categorias com menos de cinco registros submetidos (2, 3, 7, 8, 12, 13, 15, 16 e 17) foram excluídos da análise, por não fornecerem dados suficientes para o treinamento e avaliação dos modelos de classificação. Para as categorias referentes à publicação de artigos ou anais de eventos/periódicos (categorias 9, 10 e 11), foi necessária a união em uma única classe, uma vez que os certificados não apresentavam informações suficientes para distinguir entre artigos com e sem Qualis ou com diferentes níveis de Qualis.

Além disso, assume-se que cada certificado é representado por um arquivo no formato PDF e que as informações relevantes para a classificação encontram-se disponíveis nos 4.000 caracteres textuais iniciais extraídos do documento. Embora os certificados possam apresentar diferentes *layouts*, assume-se que todos contenham conteúdo textual reconhecível em algum dos objetos de texto presentes no arquivo PDF. Assim, são excluídos do escopo documentos que, após o processo de extração e limpeza, não produziram texto legível ou apresentaram apenas elementos incompatíveis com a análise, como imagens incorporadas sem camada textual.

3.2. Procedimento de Conversão dos Certificados PDF em Texto

Para possibilitar a extração de informações a partir dos certificados no formato PDF, tornou-se necessário convertê-los em texto. Para este processo, utilizou-se o utilitário

de linha de comando Pdftotext, fornecido pela biblioteca Poppler de renderização de arquivos PDF (POPPLER, 2025). Após a conversão com o utilitário, o texto extraído foi submetido a um processo de limpeza e normalização (pré-processamento), baseado na abordagem de KIRSCH e DORNELES (2023). A implementação original de KIRSCH consistia na remoção de quebras de linha, normalização de múltiplos espaços consecutivos e substituição de aspas duplas por simples. Além destes procedimentos, neste trabalho, o texto ainda foi convertido para letras minúsculas, foi feita a remoção de acentos e eliminação de caracteres especiais e números (mantendo apenas letras), resultando em uma versão contínua e padronizada do conteúdo textual, para servir como entrada para a etapa subsequente de aumento de dados.

3.3. Técnica de Aumento de Dados

Técnicas de aumento de dados (também conhecidas como *data augmentation*), ou criação artificial de dados de treino por meio de transformações, são amplamente estudadas na literatura e reconhecidas por melhorar a generalização de modelos de aprendizado de máquina. De acordo com Bayer et al., 2022, essa estratégia é útil para diversos objetivos, incluindo a regularização de modelos, a redução do esforço de rotulagem, a diminuição do uso de dados do mundo real, principalmente em domínios sensíveis à privacidade, e o balanceamento de conjuntos de dados desbalanceados.

Nos experimentos deste estudo, foram considerados dois cenários para avaliação dos modelos de classificação textual: um utilizando os certificados originais, sem aumento de dados, e outro aplicando data augmentation experimentalmente. Neste último, o objetivo foi garantir uniformidade entre as categorias e aumentar a robustez dos modelos, assegurando que cada categoria do conjunto final possuísse 156 documentos, correspondente à quantidade de registros da classe mais populosa no dataset original, mantendo equilíbrio entre as classes. A geração de novos registros foi realizada a partir do conjunto original de 292 certificados previamente rotulados, obtidos após a aplicação das delimitações apresentadas na Seção 3.1.

Para aumentar o número de registros das categorias com menor número de documentos, foram criados novos registros usando técnicas de tradução de idiomas e substituição de palavras, gerando novas variações de dados a partir dos registros existentes. Esse procedimento é detalhado a seguir.

Inicialmente, o texto de cada documento foi traduzido do português para o inglês utilizando a biblioteca Deep-Translator, uma biblioteca de código aberto que permite traduções automáticas entre múltiplos idiomas (NIDHALOFF, 2025), o que possibilitou, na etapa seguinte, a aplicação de sinônimos por meio da base lexical WordNet, um grande banco de dados que agrupa palavras em conjuntos de sinônimos e mapeia relações semânticas entre elas (MILLER, 1995), garantindo que palavras fossem substituídas por sinônimos aleatórios, mas de forma reprodutível usando uma semente fixa com valor 45.

Após a introdução das variações lexicais, os textos foram retraduzidos para o português, resultando em novos registros baseados nos certificados originais mas com pequenas alterações semânticas. Esse processo, realizado por meio da técnica de backtranslation (Sennrich et al., 2016), combinado com a substituição de sinônimos, permitiu gerar novos conteúdos que preservam as estruturas originais do documento, ao mesmo tempo em que introduzem variações lexicais.

Ao final, todos os registros foram padronizados e incorporados ao conjunto de dados, produzindo um conjunto equilibrado para o treinamento e avaliação de modelos de classificação. A Figura 1 ilustra o fluxo completo do processo de aumento de dados, destacando as etapas de tradução, substituição de sinônimos e retradução que introduzem variações textuais e lexicais para diversificar o conjunto de dados.

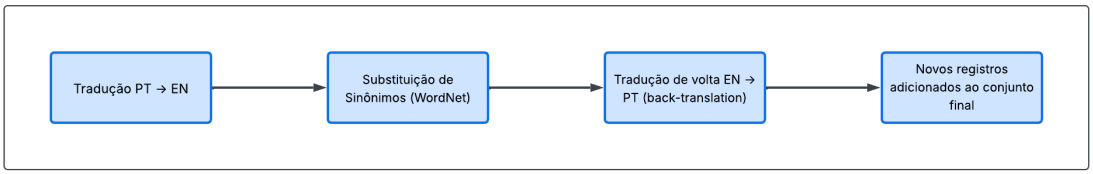

Figura 1. Fluxo do processo de aumento de dados aplicado aos certificados.

A Tabela 1 apresenta a distribuição das categorias de Atividades Complementares de Curso após a aplicação do aumento de dados. Essas categorias correspondem a classificação de documentos exigida pelo curso de Ciência da Computação do IFFar/FW. A coluna "Antes" apresenta a quantidade de registros originais após a etapa de delimitação do escopo do estudo. Já a coluna "Depois" apresenta o número de registros disponíveis após a aplicação da técnica de aumento de dados, totalizando 156 registros em cada categoria utilizada. As categorias sinalizadas com "-" não foram utilizadas nesse estudo.

Tabela 1. Distribuição das categorias de atividades complementares e registros antes e após o aumento de dados

Categoria	Atividades Complementares de Curso	Antes	Depois
1	Participação em Projetos de Pesquisa	5	156
2	Participação em Projetos de Ensino	0	-
3	Participação em Projetos de Extensão	0	-
4	Participação em eventos da área (semanas acadêmicas, ... , simpósios, fóruns, congressos, mostra, workshop)	83	156
5	Participação em cursos de extensão	8	156
6	Estágios curriculares não obrigatórios ou exercício profissional com vínculo empregatício, na área do curso	15	156
7	Estágios curriculares não obrigatórios em outras áreas	1	-
8	Monitorias na área	0	-
9	Publicação de artigo/resumo em anais de congressos, seminários, iniciação científica ou periódicos sem qualis	7	
10	Publicação de artigo/resumo em anais de congressos ou periódicos com qualis B3 ou inferior	0	156
11	Publicação de artigo/resumo em anais de congressos ou periódicos com qualis B2 ou superior	1	
12	Participação em serviço voluntário	1	-
13	Visitas técnicas (não previstas na carga horária das disciplinas da matriz curricular)	0	-
14	Participação em cursos da área	122	156
15	Disciplinas cursadas em outros cursos na área de formação do estudante	2	-
16	Participação em comissões organizadoras de eventos	1	-
17	Participação em entidades estudantis ou representação discente junto a órgãos colegiados da Instituição	2	-
18	Participação em atividades culturais/esportivas promovidas pela Instituição ou representando a Instituição	10	156
19	Participação em cursos de idiomas estrangeiros ou Libras	10	156
20	Outras	31	156

3.4. Avaliação do Modelo com Validação Cruzada

A avaliação de modelos de classificação é uma etapa essencial para mensurar sua capacidade de generalização e a confiabilidade das predições. Para esse fim, foi utilizada a técnica de validação cruzada *K-fold*, que permite analisar o desempenho do modelo em diferentes subconjuntos do conjunto de dados, evitando que os resultados dependam de uma única divisão entre treino e teste. Kohavi (1995) realizou um estudo comparativo de

técnicas de estimativa de acurácia e seleção de modelos, demonstrando que a validação cruzada estratificada, especialmente o esquema *K-fold*, apresenta desempenho consistente para avaliação de classificadores. Dessa forma, é possível obter uma estimativa robusta do desempenho do modelo e melhorar sua capacidade de generalização.

O procedimento envolve dividir o conjunto de dados em K partes aproximadamente iguais. Em cada iteração, uma dessas partes é utilizada como conjunto de teste, enquanto as demais formam o conjunto de treinamento. Esse processo é repetido até que todas as partes tenham sido utilizadas como teste, garantindo que cada amostra contribua para a avaliação do modelo.

Ao final de cada iteração são calculadas métricas de desempenho, como acurácia, precisão, *recall* e F1-*score*, para o *fold* correspondente. No fim do processo, a média das métricas fornece uma estimativa mais confiável da performance do modelo, refletindo sua capacidade de classificação em diferentes cenários e subconjuntos de dados.

3.5. Procedimento de Treinamento com Scikit-Learn

O treinamento dos modelos de classificação textual foi realizado utilizando a biblioteca Scikit-learn, seguindo um pipeline padronizado. Inicialmente, os textos previamente processados foram convertidos em representações numéricas por meio do método *Term Frequency–Inverse Document Frequency* (TF-IDF), que atribui pesos aos termos considerando a frequência do termo em um documento em relação à sua frequência no corpus, de forma a destacar palavras mais informativas (SALTON; BUCKLEY, 1988). Para capturar tanto informações léxicas individuais quanto combinações relevantes de palavras, foram considerados unigramas (palavras isoladas) e bigramas (pares consecutivos de palavras). A vetorização foi limitada às quatro mil palavras mais relevantes, e termos com frequência inferior a dois foram desconsiderados, resultando em uma matriz esparsa na qual cada documento é representado pelos pesos TF-IDF de seus termos.

A avaliação dos modelos foi conduzida por meio de validação cruzada estratificada com K = 5 partições, conforme Seção 3.4. Em cada iteração, aproximadamente 80% dos dados são utilizados para treinamento e 20% para validação, preservando a proporção das classes em todos os *folds*. A implementação computacional desse procedimento é apresentada na Figura 2. O algoritmo foi desenvolvido de forma genérica, recebendo como entrada o classificador a ser treinado, bem como os parâmetros necessários para a leitura do conjunto de dados e para o armazenamento dos modelos resultantes.

Conforme descrito nesta subseção, os textos são convertidos em representações vetoriais por meio da técnica TF-IDF, cujos parâmetros de vetorização são definidos previamente (linhas 13). Na sequência, é executada a validação cruzada estratificada com cinco partições (linha 21), na qual, a cada fold, uma nova instância do classificador é clonada e ajustada exclusivamente com os dados de treinamento correspondentes, garantindo a independência entre os modelos treinados (linhas 22–24). Em seguida, o modelo treinado é avaliado sobre o subconjunto de teste correspondente, sendo gerado um relatório detalhado de desempenho contendo métricas como precisão, revocação e F1-score (linhas 26–28). Ao final da execução do algoritmo de treinamento, os modelos obtidos em cada fold, bem como o vetor TF-IDF ajustado sobre todo o conjunto textual, são armazenados para utilização posterior na etapa de predição (linhas 30–31). Adicionalmente, os relatórios de avaliação gerados em cada fold são retornados pela função na forma de uma

estrutura de dados agregada, permitindo a análise sistemática do desempenho do classificador ao longo das diferentes partições da validação cruzada (linha 33).

A partir dos modelos treinados, obtém-se um conjunto de modelos independentes, cada um correspondente a um fold da validação cruzada estratificada. Com base nesses modelos, foi implementado um procedimento de combinação de predições por meio de votação majoritária, empregado no contexto experimental deste trabalho para a obtenção das predições finais, no qual cada classificador realiza uma predição independente para uma amostra de entrada, e a classe final é definida pela classe mais frequentemente indicada entre as predições individuais. Essa estratégia permite explorar o treinamento de diferentes subconjuntos dos dados, contribuindo para maior estabilidade das decisões e redução da sensibilidade a variações específicas das partições de treinamento.

```
1   import joblib
2   import pandas as pd
3   import os
4   from sklearn import clone
5   from sklearn.feature_extraction.text import TfidfVectorizer
6   from sklearn.model_selection import StratifiedKFold
7   from sklearn.metrics import classification_report
8
9   def train_model(csv_path, text_column, category_column, model, models_directory, SEED):
10
11      df = pd.read_csv(csv_path)
12
13      vectorizer = TfidfVectorizer(max_features=4000, ngram_range=(1,2), min_df=2)
14
15      X = vectorizer.fit_transform(df[text_column])
16      y = df[category_column]
17
18      trained_models = []
19      reports = []
20
21      for train_idx, test_idx in StratifiedKFold(n_splits=5, shuffle=True, random_state = SEED).split(X, y):
22          trained_model = clone(model)
23          trained_model.fit(X[train_idx], y.iloc[train_idx])
24          trained_models.append(trained_model)
25
26          y_pred = trained_model.predict(X[test_idx])
27          report = classification_report(y.iloc[test_idx], y_pred, output_dict=True)
28          reports.append(report)
29
30      joblib.dump(trained_models, os.path.join(models_directory, "fold_models.pkl"))
31      joblib.dump(vectorizer, os.path.join(models_directory, "tfidf_vectorizer.pkl"))
32
33      return reports
```

Figura 2. Implementação do procedimento de treinamento com validação cruzada estratificada.

Para garantir a reprodutibilidade dos resultados e permitir que outros pesquisadores explorem e utilizem a metodologia proposta, o código-fonte completo desenvolvido para este trabalho foi disponibilizado em um repositório público no *GitHub*. O repositório contém os principais módulos implementados para este estudo, incluindo extração de texto de certificados em PDF, procedimentos de aumento de dados por meio de back-translation e substituição de sinônimos, scripts de treinamento de modelos de classificação e scripts para predição de categorias em novos documentos. Também estão incluídos os arquivos de configuração e dependências necessários para reproduzir os experimentos. O código está licenciado sob a licença MIT e pode ser acessado em: `https://github.com/PHSouthier/acc-certificate-classification`.

4. Experimentos Computacionais e Resultados

Nesta seção são apresentados os experimentos computacionais realizados para avaliar o desempenho dos modelos de classificação de certificados de atividades complementares. Os experimentos foram organizados em dois grupos distintos: (i) experimentos principais, conduzidos exclusivamente com certificados reais, utilizados para avaliar a capacidade de generalização dos modelos; e (ii) experimentos complementares de caráter exploratório, realizados com dados aumentados artificialmente, com o objetivo de analisar o comportamento dos modelos em um cenário controlado de maior balanceamento entre classes.

Ressalta-se que o conjunto de dados original apresentava forte desbalanceamento entre as categorias, o que motivou a aplicação de técnicas de aumento de dados para criar um conjunto mais equilibrado e representativo. Os experimentos exploratórios não devem ser interpretados como estimativas de generalização, uma vez que o aumento de dados foi aplicado previamente à validação cruzada.

O objetivo dos experimentos foi responder às seguintes questões de pesquisa:

i) Qual a acurácia média obtida pelos modelos avaliados no conjunto de certificados reais?
ii) Entre os principais algoritmos de classificação avaliados, qual apresenta melhor desempenho para a categorização automática de certificados, considerando acurácia, precisão, *recall* e F1-*score*?

Os experimentos foram realizados em um MacBook Air 13-inch (2025), com chip Apple M4, 16 GB de memória RAM, rodando o sistema operacional macOS Tahoe 26.0.1. A implementação foi realizada em Python 3.13.7, utilizando a biblioteca Scikit-learn 1.7.2, Pandas 2.3.3, Joblib 1.5.2, Deep Translator 1.11.4, Poppler com o utilitário pdftotext 25.09.1 e a base lexical WordNet, acessada por meio do pacote NLTK 3.9.2, no ambiente de desenvolvimento Visual Studio Code.

Para avaliação dos modelos, foi utilizada a técnica de validação cruzada K-Fold com K = 5, dividindo o conjunto de dados em cinco subconjuntos aproximadamente iguais, garantindo proporção equivalente de cada categoria em cada fold, conforme detalhado na seção 3.4, e realizando o embaralhamento dos dados.

Os modelos avaliados incluíram Naive Bayes, Árvore de Decisão, *Support Vector Machine* (SVM), Rede Neural, Regressão Logística e *Random Forest*, todos instanciados utilizando as implementações padrão disponibilizadas pela biblioteca Scikit-learn. Não foram especificados limites máximos de iteração para os algoritmos iterativos, uma vez que todos os modelos apresentaram convergência com os valores padrão da biblioteca. Para garantir a reprodutibilidade dos experimentos, uma semente fixa (45) foi utilizada tanto no embaralhamento dos dados para a criação dos *folds* quanto na inicialização aleatória dos algoritmos que permitem esse parâmetro.

4.1. Resultados dos Experimentos com Certificados Reais

Os experimentos principais foram conduzidos utilizando exclusivamente um conjunto de 292 certificados reais previamente rotulados, arquivados no curso de Bacharelado em Ciência da Computação do IFFar/FW, após a aplicação das delimitações descritas na Seção 3.1.

A Tabela 2 apresenta a acurácia média dos modelos de classificação avaliados utilizando certificados reais, sem a aplicação de técnicas de data augmentation. As acurácias médias variam entre 70,88% e 82,54%, evidenciando diferenças de desempenho entre os algoritmos quando treinados exclusivamente com o conjunto de dados original. Convém observar que os valores da tabela estão arredondados; considerando maior precisão, a Rede Neural obteve 82,542%, enquanto a SVM alcançou 82,537%.

Em análise, os modelos Rede Neural e SVM alcançaram as maiores acurácias médias, com 82,542% e 82,537%, respectivamente. O *Random Forest* apresentou acurácia média de 78,08%, enquanto a Árvore de Decisão, a Regressão Logística e o Naive Bayes obtiveram 74,31%, 71,23% e 70,88%, respectivamente.

De forma geral, os resultados indicam que, entre os modelos avaliados, a Rede Neural se destacou, seguida de perto pelo SVM, apresentando as maiores acurácias médias na classificação dos certificados. Vale considerar que algumas categorias possuem quantidade reduzida de registros, o que pode influenciar a consistência das métricas por classe, sendo necessário considerar essa hipótese ao interpretar os resultados.

Tabela 2. Acurácia média dos modelos de classificação sem aumento de dados

Modelo	**Acurácia média**
Naive Bayes	0.7088
Árvore de Decisão	0.7431
SVM	0.8254
Rede Neural	0.8254
Regressão Logística	0.7123
Random Forest	0.7808

Com o objetivo de analisar esse desempenho de forma mais detalhada, a Tabela 3 apresenta os resultados obtidos pela Rede Neural ao longo dos folds da validação cruzada.

Tabela 3. Desempenho detalhado da Rede Neural sem aumento de dados

Categoria	Fold 1			Fold 2			Fold 3			Fold 4			Fold 5			Média		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	1.00	1.00	1.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.60	0.60	0.60
4	0.84	0.94	0.89	0.88	0.88	0.88	0.75	0.94	0.83	0.80	0.94	0.86	0.93	0.82	0.88	0.84	0.90	0.87
5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	0.60	1.00	0.75	1.00	0.33	0.50	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.92	0.87	0.85
9	1.00	0.50	0.67	0.50	1.00	0.67	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.50	0.67	0.90	0.80	0.80
14	0.79	0.92	0.85	0.88	0.92	0.90	0.96	0.88	0.92	0.91	0.83	0.87	0.82	0.96	0.88	0.87	0.90	0.88
18	0.00	0.00	0.00	1.00	0.50	0.67	1.00	1.00	1.00	0.50	0.50	0.50	1.00	1.00	1.00	0.70	0.60	0.63
19	1.00	0.50	0.67	0.67	1.00	0.80	0.00	0.00	0.00	0.33	0.50	0.40	1.00	1.00	1.00	0.60	0.60	0.57
20	1.00	0.57	0.73	0.62	0.83	0.71	0.60	0.50	0.55	1.00	0.83	0.91	0.71	0.83	0.77	0.79	0.71	0.73
Média	0.69	0.60	0.62	0.62	0.61	0.57	0.70	0.70	0.70	0.73	0.73	0.73	0.72	0.68	0.69	0.69	0.66	0.66
Acurácia		0.8136			0.8136			0.8103			0.8276			0.8621			0.8254	

Para cada categoria de Atividade Complementar, são reportados os valores de precisão (P), *recall* (R) e F1-*score* (F1) obtidos em cada um dos cinco folds. A coluna "Média" apresenta, para cada categoria, a média dessas métricas considerando todos os *folds*. Ao final da tabela, são apresentadas as médias de precisão, *recall* e F1-*score* calculadas para cada *fold* considerando todas as categorias, bem como a acurácia obtida em

cada fold, sendo que a coluna "Média" também reporta a média final dessas métricas e a acurácia média final do modelo. Essa análise permite avaliar não apenas a acurácia global, mas também a consistência do modelo em diferentes categorias de certificados.

Os resultados apresentados na Tabela 3 evidenciam elevada variabilidade no desempenho da Rede Neural entre os diferentes folds e categorias. Observam-se casos extremos em que determinadas classes apresentaram valores nulos de precisão, *recall* e F1-*score* em alguns *folds*, assim como situações em que essas métricas assumiram valores reduzidos, indicando que o modelo apresentou dificuldades em identificar corretamente instâncias dessas categorias em determinados particionamentos dos dados.

Essa variabilidade pode estar associada à baixa quantidade de registros disponíveis em algumas classes no conjunto original, o que pode resultar em particionamentos com número insuficiente de exemplos representativos durante o treinamento. Como consequência, o modelo tende a apresentar instabilidade nas métricas por classe, com quedas acentuadas de desempenho em categorias minoritárias.

Em síntese, os resultados obtidos demonstram que, embora a acurácia média global da Rede Neural sem aumento de dados seja relativamente elevada (82,54%), há considerável variabilidade nas métricas por categoria, com classes minoritárias, como 1, 5 e 18, apresentando precisão, *recall* e F1-*score* zerados ou muito baixos em alguns *folds*, enquanto categorias mais representativas, como 6 e 14, alcançam resultados consistentes e elevados. Essa análise evidencia que o modelo apresenta limitações na generalização para classes com poucos exemplos, reforçando a necessidade de estratégias como aumento de dados ou balanceamento de classes para garantir maior consistência. Ainda assim, os resultados evidenciam que o modelo consegue classificar corretamente a maioria das categorias mais representativas, mesmo que apresente instabilidade em classes minoritárias.

4.2. Resultados dos Experimentos Exploratórios com Aumentado de Dados

Com o objetivo de complementar a análise realizada com os certificados originais, uma segunda série de experimentos foi conduzida em um cenário experimental controlado, utilizando um conjunto de dados aumentado artificialmente, conforme descrito na Seção 3.3. O procedimento de aumento de dados permitiu uniformizar o número de exemplos por categoria, ampliando o conjunto de dados para um total de 1404 registros. Ressalta-se que, nesses experimentos, o aumento de dados foi aplicado antes da validação cruzada, o que inviabiliza a interpretação dos resultados como estimativas de generalização. Assim, os resultados apresentados nesta subseção possuem caráter estritamente exploratório.

Analisando os resultados demonstrados na Tabela 4, observa-se que a Rede Neural apresentou a maior acurácia média, com valor de 96.65%, seguida pela SVM com 96.44%. O modelo *Random Forest* também se destacou, atingindo 94.94% de acurácia média, evidenciando robustez frente a diferentes *folds* da validação cruzada. Em seguida, a Regressão Logística obteve 94.23% de acurácia média, enquanto o Naive Bayes e as Árvores de Decisão apresentaram desempenho relativamente inferior, com 89.17% e 89.03% respectivamente.

De forma geral, os resultados podem estar associados ao aumento do volume de dados e a uma distribuição mais uniforme entre as classes, sugerindo maior consistência no desempenho dos modelos avaliados.

Tabela 4. Acurácia média dos modelos de classificação com aumento de dados

Modelo	Acurácia média
Naive Bayes	0.8917
Árvore de Decisão	0.8903
SVM	0.9644
Rede Neural	0.9665
Regressão Logística	0.9423
Random Forest	0.9494

A seguir, a Tabela 5 apresenta os resultados da Rede Neural com data augmentation ao longo dos cinco *folds* da validação cruzada. São exibidas as métricas de precisão (P), *recall* (R) e F1-*score* (F1) para cada categoria, bem como a acurácia por *fold*, e as médias das métricas, permitindo uma análise detalhada do desempenho do modelo em um cenário com aumento de dados.

Tabela 5. Desempenho detalhado da Rede Neural com Aumento de Dados

Categoria	Fold 1			Fold 2			Fold 3			Fold 4			Fold 5			Média		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
4	0.96	0.84	0.90	0.88	0.91	0.89	0.91	0.94	0.92	0.97	0.90	0.93	1.00	0.97	0.98	0.94	0.91	0.92
5	0.97	0.97	0.97	0.97	0.97	0.97	1.00	0.97	0.98	0.97	0.97	0.97	0.97	1.00	0.98	0.98	0.98	0.97
6	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
9	0.97	1.00	0.98	1.00	1.00	1.00	0.97	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00	0.99	1.00	0.99
14	0.84	0.84	0.84	0.90	0.84	0.87	0.96	0.84	0.90	0.90	0.84	0.87	0.97	0.94	0.95	0.91	0.86	0.89
18	1.00	1.00	1.00	0.91	0.97	0.94	1.00	1.00	1.00	0.94	1.00	0.97	1.00	1.00	1.00	0.97	0.99	0.98
19	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.97	1.00	0.98	1.00	1.00	1.00	0.99	1.00	1.00
20	0.88	0.97	0.92	0.93	0.90	0.92	0.91	1.00	0.95	0.91	0.94	0.92	0.97	1.00	0.98	0.92	0.96	0.94
Média	0.96	0.96	0.96	0.95	0.95	0.95	0.97	0.97	0.97	0.96	0.96	0.96	0.99	0.99	0.99	0.97	0.97	0.97
Acurácia		0.9573			0.9537			0.9715			0.9609			0.9893			0.9665	

Os resultados da Tabela 5 mostram que a Rede Neural apresentou médias altas de precisão, *recall* e F1-score para a maioria das categorias, com métricas relativamente estáveis entre os diferentes folds. A acurácia por *fold* variou entre 95,37% e 98,93%, fornecendo uma visão detalhada do desempenho do modelo com aumento de dados.

Em termos de desempenho por classe, observa-se que diversas categorias apresentaram métricas praticamente perfeitas. Especificamente, as categorias 1 e 6 atingiram valores de precisão, recall e F1-*score* com 100% em todos os folds. Por outro lado, algumas categorias, como as de rótulos 4, 14 e 20, apresentaram métricas ligeiramente inferiores, especialmente em determinados folds, indicando variação entre as classes.

De forma geral, os resultados obtidos mostram que a Rede Neural apresentou métricas elevadas e relativamente estáveis ao longo dos diferentes *folds* da validação cruzada. A acurácia média do modelo foi de 96,65%, e as médias de precisão, *recall* e F1-*score* por categoria permaneceram próximas de 97%, indicando consistência das métricas no cenário com aumento de dados. Esses resultados levam à hipótese de que a uniformização das classes contribui para maior estabilidade nas métricas,

4.3. Considerações Finais dos Experimentos

Além dos resultados obtidos com os certificados reais, realizou-se uma análise complementar utilizando o conjunto de dados aumentado artificialmente. Essa avaliação experimental não tem caráter de generalização, mas permite investigar tendências sobre o efeito do balanceamento de classes na estabilidade das métricas da Rede Neural.

A comparação direta entre os resultados obtidos com e sem aplicação de data augmentation permite observar diferenças no desempenho da Rede Neural. Em geral, todas as métricas, precisão, *recall*, F1-*score* e acurácia, apresentam valores mais elevados no conjunto de dados aumentado, com menor variabilidade entre os *folds* e maior uniformidade nas categorias originalmente minoritárias. Enquanto a maior acurácia média no conjunto original foi de 82,54%, no conjunto com aplicação de data augmentation foi de 96,65%, indicando que a uniformização das classes tende a favorecer a estabilidade das métricas.

Cabe destacar que os resultados obtidos com aumento de dados neste estudo têm caráter estritamente experimental e não devem ser interpretados como estimativa de generalização do modelo, uma vez que o aumento de dados foi aplicado antes da divisão em *folds* na validação cruzada, de modo que o modelo pode ter sido avaliado em exemplos gerados artificialmente a partir do conjunto de treino. Essa abordagem apresenta limitações importantes: a similaridade entre exemplos originais e artificiais pode inflar as métricas, refletindo mais a consistência do conjunto do que o desempenho em dados inéditos. Além disso, se o aumento de dados não capturar adequadamente a variabilidade natural das classes, o modelo pode apresentar desempenho enviesado, especialmente em categorias originalmente minoritárias.

Finalmente, com base nos certificados reais, os resultados permitem responder diretamente às questões de pesquisa formuladas no início desta seção. Em relação à primeira questão, referente à acurácia média obtida pelos modelos avaliados, a Tabela 2 mostra que, os modelos alcançaram valores de acurácia entre 70,88% e 82,54%, sendo que a Rede Neural apresentou a maior acurácia média (82,542%), por uma diferença mínima em relação à obtida pela SVM (82,537%). No que diz respeito à segunda questão, que investiga qual algoritmo apresenta melhor desempenho na categorização automática de certificados, a análise detalhada apresentada na Tabela 3 mostra que a Rede Neural manteve valores elevados de acurácia média, em comparação aos outros modelos, apresentando desempenho consistente ao longo dos *folds* da validação cruzada.

5. Considerações Finais

Este trabalho apresentou uma abordagem para a classificação automática de certificados de Atividades Complementares de Curso, baseada em algoritmos de aprendizado supervisionado implementados em Python por meio da biblioteca Scikit-learn. Os experimentos foram conduzidos a partir de um conjunto de dados composto por 292 certificados arquivados no curso de Bacharelado em Ciência da Computação do IFFar/FW, após aplicação das delimitações apresentadas na Seção 3.1, avaliados por meio de validação cruzada estratificada K-Fold, permitindo uma análise consistente do desempenho dos modelos.

Os resultados obtidos com os certificados originais, sem aplicação de aumento de dados, indicam que os modelos *Support Vector Machine* (SVM) e Rede Neural se

destacaram na tarefa de categorização automática. Em particular, a Rede Neural destacou-se como o modelo mais eficaz, alcançando acurácia média de (82,542%), ligeiramente superior à SVM (82,537%). Além da acurácia global elevada, o modelo mostrou métricas relativamente consistentes de precisão, *recall* e F1-*score* ao longo dos diferentes *folds*, embora algumas categorias tenham apresentado métricas mais baixas em determinados *folds*, o que sugere que o desempenho pode variar de acordo com a representatividade das classes. Os demais modelos, *Random Forest*, Regressão Logística, Naive Bayes e Árvores de Decisão, apresentaram resultados inferiores, mas mantiveram desempenho consistente e resultados satisfatórios para aplicações de classificação em contexto acadêmico.

Os resultados obtidos com aumento de dados, embora de caráter exploratório, sugere que a uniformização das classes pode favorecer a estabilidade das métricas, resultando em maior acurácia média e consistência entre *folds*. No entanto, essa estratégia apresenta limitações, pois os dados sintéticos não correspondem a registros originais distintos e podem inflar métricas sem refletir desempenho real em dados inéditos.

Com base nos resultados alcançados, observa-se que a abordagem proposta possui elevado potencial de aplicação em sistemas de gestão de Atividades Complementares de Curso, como o SIVAC. A elevada acurácia obtida sugere que grande parte do processo de categorização pode ser automatizada, reduzindo significativamente o esforço manual na análise dos certificados, agilizando o trabalho dos coordenadores e facilitando o acompanhamento e a submissão das atividades por parte dos alunos, além de contribuir para maior padronização e confiabilidade no processo de validação.

Como trabalhos futuros, sugere-se: (i) ampliar a base de dados com a coleta de novos certificados reais; (ii) investigar combinações de algoritmos por meio de métodos ensemble, visando potencializar o desempenho dos classificadores; (iii) explorar técnicas mais avançadas de representação textual, como embeddings contextuais; (iv) aplicar a metodologia a diferentes cursos e instituições, ampliando a generalização da abordagem proposta; (v) avaliar o impacto de ajustes de hiperparâmetros nos modelos, buscando otimizar ainda mais o desempenho dos classificadores.

Referências

Bayer, M., Kaufhold, M.-A., and Reuter, C. (2022). A survey on data augmentation for text classification. *ACM Comput. Surv.*, 55(7).

Carvalho, R. R. (2024). Classificação de documentos da administração pública utilizando inteligência artificial. Dissertação (mestrado em ciência da computação), Instituto de Informática, Universidade Federal de Goiás, Goiânia.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, NY, 2 edition.

Instituto Federal Farroupilha (2022). Projeto pedagógico do curso de bacharelado em ciência da computação – campus frederico westphalen. `https://www.iffarroupilha.edu.br/projeto-pedag%C3%B3gico-de-curso/campus-frederico-westphalen`. Documento institucional. Acesso em: 13 jan. 2026.

Kirsch, B. G. and Árton Pereira Dorneles (2023). Desenvolvimento de uma ferramenta para reconhecimento de entidades nomeadas em certificados de atividades complementares de curso utilizando spacy. *Anais do Encontro Anual de Tecnologia da Informação*, 12(1):44.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, page 1137–1143, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Commun. ACM*, 38(11):39–41.

Nidhaloff (2025). deep-translator: A flexible free and unlimited python tool to translate between different languages in a simple way using multiple translators. https://pypi.org/project/deep-translator/. Acesso em: 15 dez. 2025.

Noguti, M. Y., Vellasques, E., and Oliveira, L. S. (2020). Legal document classification: An application to law area prediction of petitions to public prosecution service. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Poppler. Poppler: a pdf rendering library. `https://poppler.freedesktop.org`. Acesso em: 17 nov. 2025.

Python Software Foundation. Site oficial do python. `https://www.python.org/`. Acesso em: 20 mai. 2025.

Russell, S. J. and Norvig, P. (2013). *Inteligência Artificial*. Elsevier, Rio de Janeiro, Brasil, 3 edition.

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing Management*, 24(5):513–523.

Scikit-learn. Scikit-learn: Machine learning in python. `https://scikit-learn.org/stable/index.html`. Acesso em: 20 maio 2025.

Sennrich, R., Haddow, B., and Birch, A. (2016). Improving neural machine translation models with monolingual data. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 86–96.

Silva, N. C. d., Braz, F. A., de Campos, T. E., et al. (2018). Document type classification for brazil's supreme court using a convolutional neural network. In *10th International conference on forensic computer science and cyber law (ICoFCS), Sao Paulo, Brazil*, pages 7–11.

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing Management*, 45(4):427–437.

Southier, P. H. and Árton Pereira Dorneles (2022). Desenvolvimento de uma plataforma web em django para gerenciamento de atividades complementares de curso. *Anais do Encontro Anual de Tecnologia da Informação*, 11(1):28.