Estudo de técnicas de Aprendizado Profundo para prognóstico de pacientes com Covid-19

Willian Penteado[1], **Carolina Paula de Almeida**[1], **Sandra Mara Guse Scós Venske**[1]

[1]Departamento de Ciência da Computação – Universidade Estadual do Centro-Oeste (UNICENTRO)
Caixa Postal 730 – 85.015-430 – Guarapuava – PR – Brasil

57020140062@unicentro.edu.br, {carol,ssvenske}@unicentro.br

Abstract. *COVID-19 is an acute respiratory disease caused by the SARS-CoV-2 coronavirus, with a high potential for transmission and global impact. This study explores the use of Deep Learning (DL), a subfield of Machine Learning (ML), to develop a prognostic model for COVID-19 based on tabular data. Traditionally, classical ML methods are used for this type of problem, but recently, DL has gained interest in this context. The objective of this work was to study DL techniques, especially 1D Convolutional Neural Networks (CNNs), to create a prognostic model for COVID-19 patients. The choice to use CNN was due to the large amount of scientific work using this architecture and its challenging approach to working with tabular data. A pre-processed dataset from the proponent's undergraduate research, originating from the Brazilian Unified Health System (SUS), was used. The 1D CNN demonstrated the ability to outperformed the AdaBoost method in experiments. The contribution of this work is the study of DL behavior in problems with tabular data, specifically related to COVID-19.*

Resumo. *A Covid-19 é uma doença respiratória aguda causada pelo coronavírus SARS-CoV-2, com alto potencial de transmissão e impacto global. Este estudo explora o uso do Aprendizado Profundo (AP), uma subárea do Aprendizado de Máquina (AM), para desenvolver um modelo de prognóstico de Covid-19 com base em dados tabulares. Tradicionalmente, métodos clássicos de AM são usados para este tipo de problema, mas recentemente, o AP tem despertado interesse nesse contexto. O objetivo deste trabalho foi o estudo de técnicas de AP, especialmente Redes Neurais Convolucionais (CNNs) 1D, para criar um modelo de prognóstico para pacientes com Covid-19. A escolha pela utilização da CNN foi devido à grande quantidade de trabalhos científicos utilizando esta arquitetura e à sua abordagem desafiadora para trabalhar com dados tabulares. Foi utilizado um conjunto de dados pré-processado da Iniciação Científica do proponente, proveniente do SUS. A CNN-1D demonstrou capacidade de superar o método AdaBoost nos experimentos. A contribuição deste trabalho é o estudo do comportamento do AP em problemas com dados tabulares, especificamente relacionados à Covid-19.*

Palavras-chave – Inteligência Artificial, Aprendizado de Máquina, Redes Neurais Artificiais, Classificação.

1. Introdução

A Covid-19 é uma doença respiratória aguda resultante da infecção pelo coronavírus SARS-CoV-2. Ela possui um potencial grave, com alta capacidade de transmissão e é encontrada em todo o mundo [Saúde 2021]. Após dois anos de pandemia de Covid-19, em 2022 o comportamento social atingiu o patamar mais próximo da normalidade. Mesmo que o vírus continue circulando, a taxa de mortalidade caiu consideravelmente no Brasil (83,2% no primeiro semestre), a vacinação avançou. Assim, medidas de prevenção, como o uso de máscaras, deixaram de ser obrigatórias e novas descobertas sobre a doença foram feitas [Toledo 2022].

De acordo com [Ministério da Saúde 2023] o painel de casos de doença pelo coronavírus 2019 (Covid-19) no Brasil, fornecido pelo Ministério da Saúde, até 07/07/2023, o Brasil registrou mais de meio milhão (704.159) de óbitos decorrentes da Covid-19, além de um total de 37.682.660 casos confirmados em todo o país desde o início da pandemia.

Ao longo do tempo, hospitais e governos têm armazenado uma enorme quantidade de dados relacionados a tratamento de doenças e diagnósticos clínicos. Diante disso, surge a necessidade e oportunidade de explorar esses dados visando contribuir para o avanço da saúde pública. Nesse contexto, o Aprendizado de Máquina (AM) tem se destacado de maneira significativa na área da saúde [Batista and Chiavegatto 2019].

O AM é uma subárea da Inteligência Artificial (IA) que pode ser definido como "a ciência (e a arte) da programação de computadores para que eles possam aprender com os dados" [Géron 2019]. Neste contexto, o conceito de Aprendizado Profundo (AP) pode ser relacionado, pois é uma subárea do AM. O AP se destaca do AM pela sua análise aprofundada por meio de camadas interconectadas. Sua base fundamental são as Redes Neurais Artificiais (RNAs) que são modelos matemáticos inspirados biologicamente no funcionamento do cérebro humano [Mueller 2020]. Dentro do AP, as Redes Neurais Convolucionais (CNNs), são um tipo de arquitetura de rede neural que apresenta uma estrutura semelhante a uma grade. Elas são adequadas para o processamento de dados que podem ser representados como matrizes, séries temporais e imagens [Goodfellow et al. 2016].

Tradicionalmente, os métodos clássicos de AM têm sido amplamente empregados em aplicações de dados tabulares devido ao seu desempenho superior [Chen and Guestrin 2016, Tarwidi et al. 2023]. No entanto, recentemente, tem havido um crescente interesse em explorar o uso do AP em dados tabulares [Shwartz-Ziv and Armon 2022].

O objetivo deste trabalho foi o estudo de técnicas de AP, mais especificamente CNN considerando uma dimensão (1D), para processamento de dados tabulares, visando o desenvolvimento de um modelo de prognóstico para pacientes com Covid-19. O *framework* utilizado para a codificação foi o *PyTorch*[1].

A escolha do uso da CNN foi devido à grande quantidade de trabalhos científicos utilizando esta arquitetura e sua abordagem desafiadora ao trabalhar com dados tabulares.

Em um trabalho anterior, realizado como parte de um projeto de Iniciação Científica (IC) dos autores deste estudo, foi conduzida uma análise abrangente de pré-processamento em um conjunto de dados tabular de acesso público proveniente do Sistema Único de Saúde (SUS) do Ministério da Saúde. Portanto, os conjuntos de dados utilizados neste trabalho são vertentes deste conjunto e já foram pré-processados como objetivo desta IC.

Assim, a contribuição deste trabalho foi o estudo do comportamento de um algoritmo de AP aplicado para um problema com dados tabulares, mais especificamente da Covid-19. Uma

[1]https://pytorch.org/

série de experimentos foram conduzidos, conjuntos de dados que se diferem na técnica de balanceamento foram testados, bem como variações de otimizadores e funções de ativação. O objetivo foi identificar as configurações mais eficazes para a rede neural em cada contexto específico. Além disso, houve uma comparação de resultados com a IC do proponente, onde um método clássico de AM foi aplicado, o *AdaBoost*[2].

O restante deste trabalho está organizado da seguinte forma. Na Seção 2 são apresentados os conceitos e métodos utilizados para a realização do trabalho. Na Seção 3, é detalhada a abordagem proposta. A Seção 4 analisa e discute os resultados dos experimentos realizados. Na Seção 5, são apresentadas as considerações finais e apontamentos para trabalhos futuros.

2. Fundamentação Teórica

Esta seção fornece uma base teórica dos conceitos centrais que sustentam este trabalho. Inicialmente, na Subseção 2.1, explora-se o Aprendizado Profundo. Em seguida, na Subseção 2.2, são discutidas as métricas utilizadas para avaliação dos experimentos. A Subseção 2.3 destaca as ferramentas usadas no desenvolvimento do projeto. Já na Subseção 2.4, são detalhados os conjuntos de dados empregados. Por fim, a Seção 2.5 elenca os trabalhos correlatos.

2.1. Aprendizado Profundo

No início da era da Inteligência Artificial (IA), o campo concentrou-se em resolver rapidamente problemas que eram considerados intelectualmente mais difíceis para os seres humanos, mas relativamente simples para computadores. Esses problemas podiam ser descritos por meio de uma lista de regras matemáticas formais. Porém, o verdadeiro desafio para a inteligência artificial revelou-se na resolução de tarefas fáceis para as pessoas (problemas que podem ser resolvidos intuitivamente), mas que são difíceis de descrever formalmente. Exemplos disso são reconhecimento de palavras faladas ou de rostos em imagens [Goodfellow et al. 2016].

Para solucionar esses problemas mais intuitivos, uma abordagem é permitir que os computadores aprendam com experiência e compreendam o mundo por meio de uma hierarquia de conceitos. Nessa abordagem, os conceitos são definidos em relação uns aos outros, o que permite ao computador aprender conceitos mais complexos a partir de conceitos mais simples. Essa estrutura hierárquica resulta em uma representação em forma de gráfico profundo, com várias camadas. Portanto, essa abordagem é conhecida como "Aprendizado Profundo" devido à sua representação em camadas profundas [Goodfellow et al. 2016].

Dentro do AP, as CNNs são um modelo de RNA que utilizam a operação matemática chamada convolução, que é um tipo especializado de operação linear. Basicamente, as redes convolucionais são redes neurais que aplicam a convolução em pelo menos uma de suas camadas, substituindo a multiplicação de matrizes convencional [Goodfellow et al. 2016].

Uma camada de convolução utiliza *kernels* sobre os dados de entrada, sendo que cada *kernel* detecta um atributo, gerando um mapa. A Figura 1 mostra um exemplo do *kernel* sendo aplicado em uma imagem 5x5, com cada quadrado representando um *pixel* da imagem. A área em amarelo representa o *kernel* convolucional que vai deslizar por toda a imagem (os valores do *kernel* estão em destaque em vermelho). Abaixo na figura está o mapa de atributos resultante das operações de convolução da região em amarelo.

A camada de *pooling* é aplicada após a convolução e serve para reduzir progressivamente o tamanho espacial da representação intermediária da imagem.

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

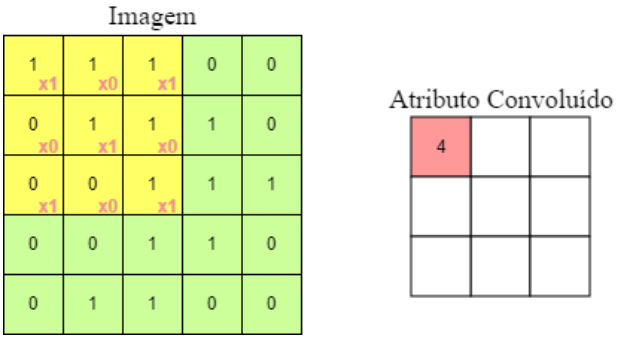

Figura 1. Operação de convolução. Fonte: o autor.

As CNNs são inspiradas na neurociência visual e possuem recursos-chave que exploram as propriedades dos sinais naturais, incluindo conexões locais no campo receptivo, compartilhamento de parâmetros via *kernel* convolucional e abstração de recursos hierárquicos por meio de *pooling* e múltiplas camadas [LeCun et al. 2015]. Esses recursos tornam as CNNs adequadas para analisar dados com dependências espaciais ou temporais entre componentes, ou seja cujos elementos mostram alguma correlação com elementos vizinhos, como em dados de imagem e áudio [Arel et al. 2010, Ismail Fawaz et al. 2019].

No entanto, muitos dados ainda existem em um formato 1D, como dados clínicos de registros médicos e, portanto, abrem questões de pesquisa desafiadoras sobre se eles podem ser efetivamente treinados para classificação usando CNN. A CNN elimina a necessidade de extração manual de atributos porque os atributos são aprendidos diretamente pelas diferentes camadas convolucionais [Guo et al. 2017].

2.2. Métricas de Avaliação

Neste trabalho foram utilizadas três métricas para avaliar o desempenho dos modelos. Essas métricas são acurácia, F_1-*score* e função de perda. Tais métricas são detalhadas abaixo.

As métricas consideradas utilizam as seguintes medições:

- VP (Verdadeiros Positivos): Refere-se às instâncias que foram corretamente classificadas como positivas pela rede neural;
- VN (Verdadeiros Negativos): Indica as instâncias que foram corretamente classificadas como negativas pela rede neural;
- FP (Falsos Positivos): São as instâncias que foram erroneamente classificadas como positivas pela rede neural; e
- FN (Falsos Negativos): Representam as instâncias que foram erroneamente classificadas como negativas pela rede neural.

Métrica Acurácia:

A acurácia do diagnóstico é dada pelo número de exemplos classificados corretamente dividido pelo número total de exemplos classificados e é calculada conforme a Equação 1 [Shaikh et al. 2021].

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{1}$$

Métrica F_1-*score*:

A métrica F_1-*score* combina as métricas de precisão (Equação 2) e sensibilidade (Equação 3) em uma única métrica. A Equação 4 define a métrica F_1 [Shalev-Shwartz and Ben-David 2014].

$$Precisão = \frac{VP}{VP + FP} \tag{2}$$

$$Sensibilidade = \frac{VP}{VP + FN} \tag{3}$$

$$F_1 = 2 \times \frac{Precisão \times Sensibilidade}{Precisão + Sensibilidade} = \frac{VP}{VP + \frac{FN+FP}{2}} \tag{4}$$

Função de Perda:

A função de perda indica o quanto a resposta da rede está distante da resposta desejada. Neste trabalho foi utilizada a *Binary Cross Entropy with Logits Loss*[3] (BCEWithLogitsLoss), que combina a função de ativação *Sigmoid* e a função de perda *BCELoss*, que é uma métrica usada para calcular a diferença entre as probabilidades previstas pelo modelo e os rótulos reais. Ela é usada quando a saída desejada é uma probabilidade entre 0 e 1 e a ativação *sigmoid* é aplicada diretamente às saídas da rede.

2.3. Ferramentas Utilizadas

Para a realização deste trabalho, foram utilizadas as ferramentas:

- *Frameworks* e Bibliotecas: *PyTorch*, *scikit-learn*[4], pandas[5], *NumPy*[6], *Matplotlib*[7] e *seaborn*[8];
- Ambiente de desenvolvimento: *Visual Studio Code*[9] (VSCode);
- Plataforma *GitHub*[10] utilizada para gerenciar versionamento.

A configuração de hardware utilizada foi: sistema operacional *Windows*, processador AMD *Ryzen* 5 5600X 3.7GHz, placa de vídeo RTX3060Ti 8GB e 16 GB de memória RAM.

2.4. Conjunto de Dados

O conjunto de dados utilizado neste trabalho é público pertencente ao Sistema Único de Saúde (SUS) do Ministério da Saúde. Ele foi obtido da base de dados do OpenDataSUS[11] e é referente a Vigilância da Síndrome Respiratória Aguda Grave (SRAG) no Brasil no ano de 2022.

[3]https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html
[4]https://scikit-learn.org/stable/
[5]https://pandas.pydata.org/
[6]https://numpy.org/
[7]https://matplotlib.org/
[8]https://seaborn.pydata.org/
[9]https://code.visualstudio.com/
[10]https://github.com/
[11]Disponível em: https://opendatasus.saude.gov.br/dataset/srag-2021-a-2023/resource/62803c57-0b2d-4bcf-b114-380c392fe825

Inicialmente este conjunto de dados era composto por 561.242 linhas (instâncias) e 166 colunas (atributos). Ele contém informações sobre exames clínicos, sintomas, datas, dados demográficos e geográficos. É relevante destacar que foi realizada uma ampla análise no dicionário de dados[12] para melhor compreensão das informações presentes no conjunto.

Por se tratar de dados reais, continha uma grande quantidade de valores faltantes e inconsistentes. O pré-processamento dos dados foi realizado durante uma IC do proponente.

Para fins de replicação do trabalho, o link do repositório do *GitHub* é o seguinte: https://github.com/Willian-P/TCC-CNN-1D. Onde pode ser conferido todos os experimentos realizados neste trabalho. Assim como, o pré-processamento do conjunto de dados durante a IC.

Abaixo são listadas algumas técnicas de pré-processamento aplicadas:

1. Delimitação do conjunto de dados para apenas notificações de Covid-19 no estado do Paraná;
2. Eliminação manual de atributos categorizados como Campos Internos, relacionados a localização e dados administrativos;
3. Limpeza de dados redundantes e inconsistentes;
4. Aplicação de um filtro para remover atributos e instâncias que possuíam mais de 50% dos seus dados incompletos;
5. Preenchimento dos dados ausentes com a mediana;
6. Tratamento de datas;
7. Conversão de tipos de dados;
8. Normalização de dados com o método *StandardScaler* [Pedregosa et al. 2011];
9. Balanceamento dos dados.

Após serem aplicadas essas técnicas de pré-processamento, com exceção do balanceamento de dados. O conjunto de dados ficou com 17.432 instâncias e 41 atributos.

Com o desbalanceamento de dados evidenciado na Figura 2, diferentes técnicas de balanceamento foram aplicadas, o que resultou em diversas versões do conjunto de dados, das quais quatro foram testadas neste estudo. Duas delas utilizam o *Synthetic Minority Over-sampling Technique* (SMOTE) que é uma técnica do *imbalanced-learn* usada para equilibrar classes desbalanceadas em conjuntos de dados. Ela cria exemplos sintéticos para a classe minoritária, ajudando algoritmos de aprendizado a lidar melhor com desequilíbrios [Lemaître et al. 2017].

As versões consideradas foram as seguintes:

- Nenhum: Nessa versão, nenhuma técnica de balanceamento foi aplicada, mantendo o conjunto de dados com 17.432 instâncias, conforme Figura 2;
- DadosSP: Essa versão envolveu o uso de dados de São Paulo para complementar a classe minoritária, expandindo o conjunto de dados para 25.210 instâncias;

[12]Disponível em: https://s3.sa-east-1.amazonaws.com/ckan.saude.gov.br/SRAG/pdfs/Dicionario_de_Dados_SRAG_Hospitalizado_19.09.2022.pdf

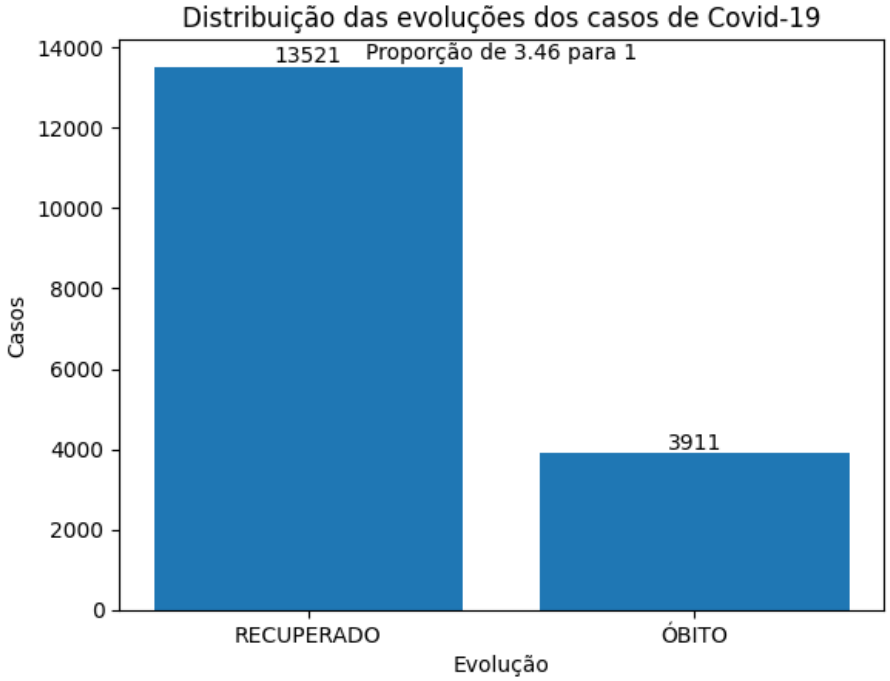

Figura 2. Desbalanceamento do conjunto de dados. Fonte: o autor.

- SMOTE85: Nessa abordagem, a técnica de balanceamento SMOTE foi empregada para elevar a classe minoritária a 85% da quantidade da classe majoritária. Resultando em 25.013 instâncias;

- SMOTE100: Aqui, o SMOTE foi aplicado para igualar a classe minoritária à classe majoritária, resultando em um total de 27.042 instâncias.

2.5. Trabalhos Correlatos

Esta seção, apresenta os trabalhos correlatos que aplicam CNN em dados tabulares. Os trabalhos selecionados são todos a partir de 2020.

Em [Borisov et al. 2022] é apresentado um *survey* de redes neurais profundas e dados tabulares. O artigo apresenta uma taxonomia unificada que categoriza as abordagens de aprendizado profundo para dados tabulares em três ramificações: métodos de transformação de dados, arquiteturas especializadas e modelos de regularização. As CNNs são apresentadas categorizadas nos métodos de transformação dos dados, citando o trabalho de [Zhu et al. 2021] (descrito mais à frente nesta seção). O foco neste caso, são as transformação de dados tabulares em imagens para servirem de entrada para as CNNs.

[Zhu et al. 2021] desenvolveu um método novo, (IGTD (*Image Generator for Tabular Data* - gerador de imagens para dados tabulares), para transformar dados tabulares em imagens para servir como entrada para CNNs. O algoritmo atribui cada atributo a um pixel na imagem. Exitem outros métodos de transformação, no artigo citam três: *DeepInsight*, REFINED (*REpresentation of Features as Images with NEighborhood Dependencies*) e OmicsMapNet. Os autores mostram as vantagens do método proposto em relação aos outros. Dois conjuntos de dados de triagem de medicamentos *in vitro* foram usados para teste, o CTRP (*Cancer Therapeutics Response Portal v2*) e o GDSC (*Genomics of Drug Sensitivity in Cancer*).

[Sharma and Kumar 2022] propuseram três novos métodos de pré-processamento de *data wrangling* que transformam um vetor de dados 1-D em uma imagem gráfica 2-D com correlações apropriadas entre os campos a fim de serem processadas por uma CNN. Os métodos foram testados nos conjuntos de dados WBC (*Wisconsin Original Breast Cancer*) e WDBC (*Wisconsin Diagnostic Breast Cancer*).

Em [Bragilovski et al. 2023] é apresentado um novo método, o TLTD (*Transfer Learning for Tabular Data*), que utiliza uma nova arquitetura de aprendizado projetada para extrair novos recursos de conjuntos de dados estruturados. Os dados tabulares são convertidos em imagens. A abordagem é testada com vinte e cinco conjuntos de dados estruturados e os resultados são favoravelmente comparados com oa algoritmos *Random Forest*, *eXtreme Gradient Boosting* (XGBoost) e Tabnet.

Em [Li et al. 2023b] é proposta uma nova estratégia para construir e otimizar modelos CNN 1D para diagnosticar danos em diferentes tipos de estrutura utilizando séries temporais relacionadas a monitoramento com sensores. Essa estratégia envolve a otimização de hiperparâmetros de uma CNN 1D com o algoritmo *bayesiano*.

Em [Ullah et al. 2022] é apresentada a nova aplicação de LRP (*Layer-wise Relevance Propagation*), que é uma técnica de explicabilidade desenvolvida para modelos profundos em visão computacional. Os autores aplicam a técnica em conjuntos de dados tabulares contendo dados mistos (categoriais e numéricos) usando uma rede neural profunda (CNN 1D), para casos de uso de detecção de fraude de cartão de crédito e previsão de telecomunicações. Os resultados indicam que o LRP é mais eficaz quando comparado a modelos tradicionais de explicabilidade, como LIME (*Local Interpretable Model-agnostic Explanations*) e SHAP (*Shapley Additive Explanations*). A rede CNN 1D utilizada foi composta por sete camadas, com três camadas de convolução (com 25, 50 e 100 *kernels*, respectivamente).

[Khan Mamun and Elfouly 2023] propuseram uma rede neural convolucional unidimensional híbrida (CNN 1-D), que usa um conjunto de dados (40.713 instâncias e 47 atributos) acumulado a partir de dados de pesquisa *online*. A CNN 1-D apresentou melhor acurácia em comparação com algoritmos clássicos de aprendizado de máquina (Floresta Aleatória, AdaBoost e Máquina de Vetores de Suporte) e *Perceptron* Multi-Camadas. O conjunto de dados escolhido foi o *National Health and Nutrition Examination Survey* para detecção de doença cardiovascular.

Em [Lima Neto 2023] é usada uma Rede Neural Convolucional Unidimensional (1D) para classificação usando a base de dados do Hospital Sírio Libanês relacionado a Covid-19, contendo exames clínicos, exceto imagens. Esse conjunto de dados é composto por 5643 instâncias (sendo 558 positivos e 5085 negativos) com 91 atributos.

Embora a pandemia de Covid-19 tenha acabado, diversos trabalhos recentes ainda utilizam conjuntos de dados sobre a doença aplicando métodos de aprendizado de máquina para predições ([Monshi et al. 2021, Elkin and Zhu 2022, Guarrasi and Soda 2023, Sofia et al. 2023, Li et al. 2023a]).

3. Abordagem proposta

Esta seção introduz a abordagem desenvolvida no âmbito deste trabalho. Primeiramente, a Subseção 3.1 detalha a metodologia adotada no decorrer do estudo. Em seguida, na Subseção 3.2, é apresentada a arquitetura da rede neural CNN-1D.

3.1. Metodologia

A Figura 3 mostra o fluxograma da metodologia adotada neste trabalho. Suas etapas são descritas a seguir:

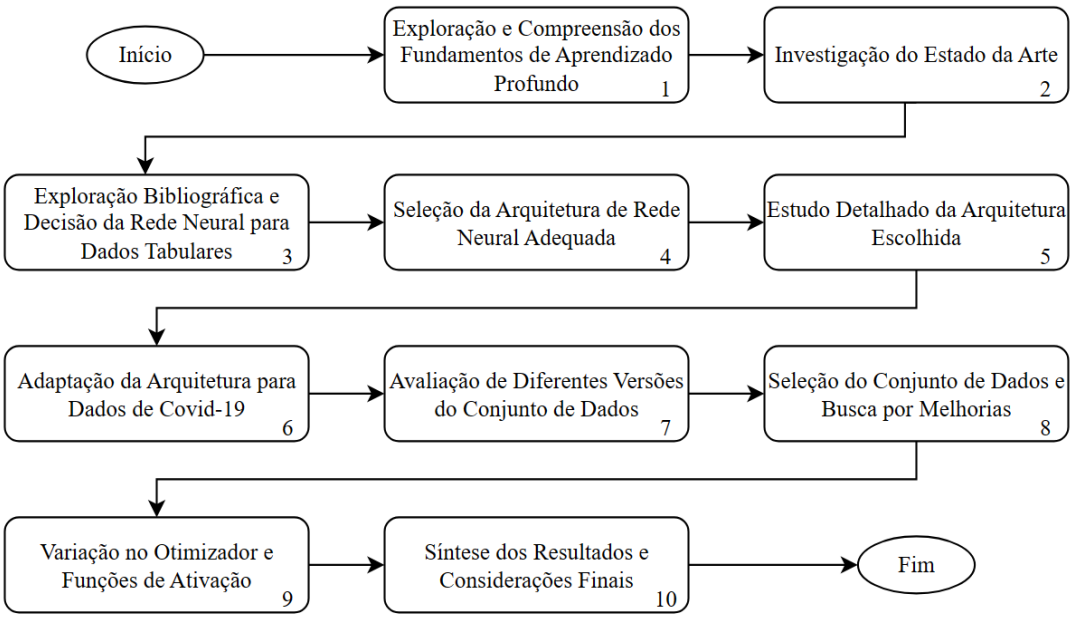

Figura 3. Fluxograma da metodologia abordada. Fonte: o autor.

1. Exploração e Compreensão dos Fundamentos de Aprendizado Profundo:

A primeira etapa deste trabalho envolveu a busca e assimilação dos conceitos fundamentais do Aprendizado Profundo. Foram explorados os princípios básicos dessa abordagem, buscando compreender seu escopo e aplicação. Além disso, houve uma contextualização da área de atuação do AP e sua importância em diversos campos.

2. Investigação do Estado da Arte:

Foi realizada uma investigação do estado da arte, que ressaltou o uso predominante dos métodos tradicionais de Aprendizado de Máquina. Embora esses métodos sejam o estado da arte, é notável que recentemente o uso de redes neurais em conjuntos de dados tabulares tem ganhado destaque (vide Seção 2.5). Mesmo que tal abordagem não seja considerada a melhor no momento, a escolha de explorar técnicas de AP assume relevância neste trabalho. Isso se dá pelo fato de contribuir para a crescente exploração da aplicação de redes neurais em dados tabulares.

3. Exploração Bibliográfica e Decisão da Rede Neural para Dados Tabulares:

Nesta etapa, foi realizado um levantamento bibliográfico, focando predominantemente em trabalhos científicos que exploram arquiteturas de redes neurais em dados tabulares. Foi optado pela arquitetura da *Convolutional Neural Networks 1-Dimensional* (CNN-1D), devido à quantidade de trabalhos científicos relacionados e sua abordagem desafiadora ao trabalhar com dados tabulares.

4. Seleção da Arquitetura de Rede Neural Adequada:

Já na quarta etapa, um processo criterioso foi conduzido para determinar a arquitetura de rede neural mais apropriada para este trabalho. Inicialmente, foi realizada uma pesquisa detalhada sobre trabalhos que aplicavam CNN-1D em dados tabulares, visando identificar abordagens que pudessem se adequar ao contexto proposto. Dentre as várias opções avaliadas, a arquitetura *SoftOrdering1DCNN*[13] destacou-se pela sua simples abordagem para

[13]https://www.kaggle.com/code/mavillan/softordering1dcnn-on-sctp/notebook

lidar com dados tabulares. Assim, essa arquitetura foi selecionada para um estudo mais aprofundado.

5. Estudo Detalhado da Arquitetura Escolhida:

A arquitetura denominada *SoftOrdering1DCNN*, foi replicada e estudada usando o conjunto de dados tabular do autor, que envolvia séries temporais. O foco desta etapa foi compreender como a arquitetura funcionava e se ajustava aos dados tabulares.

6. Adaptação da Arquitetura para Dados de Covid-19:

O conjunto de dados foi substituído por dados relacionados à Covid-19 e a arquitetura foi ligeiramente ajustada para acomodar esses novos dados. O objetivo foi analisar como a rede se comporta ao lidar com um novo conjunto de dados. Os ajustes realizados foram a troca do conjunto de dados, inserção de métricas de avaliação e atualização de comandos que ficaram depreciados.

7. Avaliação de Diferentes Versões do Conjunto de Dados:

Nesta etapa, foram conduzidos testes utilizando as quatro versões do conjunto de dados na arquitetura *SoftOrdering1DCNN*. Uma análise foi realizada para determinar qual versão seria a mais adequada para prosseguir nos experimentos. Essa avaliação levou em consideração critérios tanto quantitativos quanto qualitativos, avaliando não apenas o desempenho da rede, mas também seu comportamento em relação a cada versão do conjunto de dados.

8. Seleção do Conjunto de Dados e Busca por Melhorias:

Na etapa 8, ocorreu a seleção do conjunto de dados mais adequado com base em três métricas distintas e na observação do comportamento da rede. Porém, após uma análise detalhada dos resultados, optou-se por utilizar dois conjuntos de dados para dar continuidade aos experimentos. Essa escolha deu-se principalmente pelo fato de trabalhar com um conjunto onde os dados são puramente reais e não sintéticos. A partir dessa escolha, a atenção passou a ser direcionada para o aprimoramento dos resultados.

9. Variação no Otimizador e Funções de Ativação:

Na etapa 9, foram conduzidos um amplo conjunto de experimentos, sendo testadas variações com 5 otimizadores distintos e quatro funções de ativação diferentes. A partir dessas variações, buscou-se identificar a configuração mais adequada da rede para cada conjunto de dados, para posteriormente comparar com os resultados da IC, que utilizaram o método *AdaBoost*.

10. Síntese dos Resultados e Considerações Finais:

Na etapa final, os resultados dos experimentos foram analisados e sintetizados, abrangendo métricas quantitativas e considerações qualitativas. Com isso foram feitas as conclusões finais, destacando as contribuições deste trabalho e próximos passos possíveis, proporcionando uma compreensão do impacto geral desse estudo.

3.2. Arquitetura

A arquitetura da rede neural *SoftOrdering1DCNN* é ilustrada na Figura 4. Os valores numéricos abaixo do nome/tipo de cada camada representam seus tamanhos correspondentes. No início a camada de entrada que recebe 40 atributos do conjunto de dados, excluindo o alvo. Os dados são então processados através da camada *Fully Connected1* e, em seguida, passam pela camada

Reshape, que redimensiona os dados. Em seguida, ocorrem quatro operações de convolução, intercaladas com camadas de *Average Pooling* para redução de dimensionalidade. Posteriormente, a camada *Flatten* transforma os dados em um vetor unidimensional, preparando-os para a camada *Fully Connected2*, onde a decisão final é tomada em tarefas de classificação.

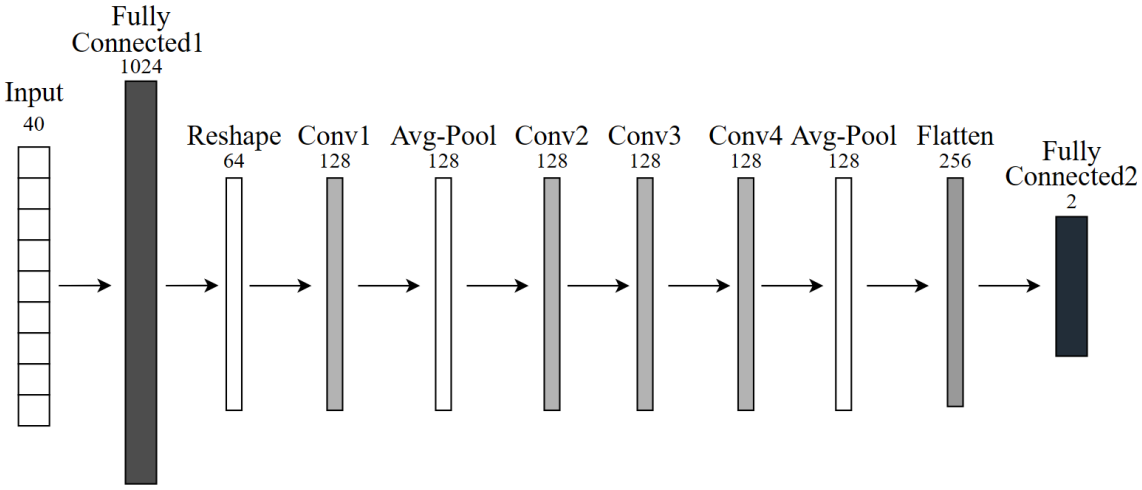

Figura 4. Arquitetura da rede. Fonte: o autor.

Uma característica distintiva desta arquitetura é a inclusão de uma camada totalmente conectada (*Fully Connected1*) logo no início de sua estrutura, antes de aplicar qualquer convolução. Isso difere de muitas arquiteturas convencionais, que não incorporam essa camada no início. Ao analisar os trabalhos correlatos, é possível comparar e identificar essa diferença fundamental na abordagem.

A camada totalmente conectada no início da arquitetura desempenha um papel importantíssimo na transformação dos dados tabulares para uma forma em que as técnicas de convolução possam ser tratadas. Essa camada tem o propósito de converter os atributos tabulares em uma representação que possua alguma forma de correlação como os *pixels* de uma imagem. Esse passo é fundamental, uma vez que os modelos convolucionais foram originalmente desenvolvidos para lidar com dados de imagem, em que busca-se explorar a propriedade de localidade espacial por meio de convoluções. Isso significa que os *kernels* são aplicados em regiões em que os *pixels* são altamente correlacionados. Essa propriedade, no entanto, não é satisfeita em dados tabulares, pois na grande maioria das vezes as colunas não possuem correlação. Portanto, a adição dessa camada ajuda a criar uma estrutura inicial que introduz certa correlação e pode ser mais facilmente processada pelas camadas seguintes.

Após essa camada inicial, segue-se a aplicação de convolução, que têm a tarefa de analisar os atributos aprendidos pela camada totalmente conectada. Essas convoluções são utilizadas para identificar padrões, relações e características relevantes nos dados tabulares reorganizados.

Portanto, a arquitetura *SoftOrdering1DCNN* emprega essa combinação de camada totalmente conectada seguida por camadas de convolução para superar o desafio de aplicar convoluções em dados tabulares. Deste ponto em diante esta arquitetura será denominada CNN-1D.

4. Experimentos

Nesta seção, são apresentados os resultados dos experimentos realizados com a rede neural CNN-1D. Os experimentos estão divididos em três subseções. Todos os experimentos os conjuntos

de dados foram divididos com 60% para treinamento, 20% para teste e 20% para validação. O número de épocas inicialmente foi definido como 200, com *early stopping*[14] de 21 épocas sem melhora na função de perda.

Na Subseção 4.1, são discutidos os resultados da CNN-1D diante dos quatro conjuntos de dados diferentes. Em seguida, na Subseção 4.2, são analisados os resultados ao variar otimizadores e funções de ativação. Por fim, na Subseção 4.3, os resultados deste trabalho são comparados com os resultados da técnica aplicada ao longo do trabalho de iniciação científica, a *AdaBoost*.

4.1. Resultados das diferentes versões do conjunto de dados considerando balanceamento

Os resultados das quatro versões do conjunto de dados considerando diferentes tipos de balanceamento são apresentados na Tabela 1. A tabela exibe os valores de acurácia, F_1-*score* e função de perda. A CNN-1D foi executada com os parâmetros do trabalho original, que utilizaram o otimizador SGD e a função de ativação Celu na camada "Fully Connected1" e a função Relu nas camadas convolucionais. A primeira coluna da Tabela 1 lista as diferentes versões para balanceamento no conjunto de dados, e as colunas subsequentes mostram os resultados correspondentes para cada métrica.

Tabela 1. Tabela de resultados em termos de acurácia, F_1-*score* e a função de perda para as 4 versões testadas. Os melhores resultados para cada métrica estão destacados em uma escala de cinza, sendo o mais escuro o melhor.

Balanceamento	Acurácia	F_1-*score*	Função de Perda
Nenhum	0.823	0.502	0.390
DadosSP	0.820	0.813	0.391
SMOTE85	0.805	0.807	0.416
SMOTE100	0.805	0.826	0.420

Ao analisar a Tabela 1, é possível compreender como a arquitetura proposta se comporta diante de um conjunto de dados desbalanceado. É notável que a versão sem qualquer tipo de balanceamento apresenta um F_1-*score* significativamente inferior em comparação com as versões balanceadas. Isso levou à escolha desta métrica como referência para o trabalho, devido à sua robustez com relação ao desbalanceamento, ao contrário de outras métricas que se mostraram menos sensíveis a essa problemática. É relevante destacar que nas versões balanceadas, as diferenças nos resultados não são tão acentuadas, ressaltando assim a importância de considerar múltiplas métricas para uma avaliação abrangente de um modelo.

Dessa forma, para a escolha da versão mais adequada do conjunto de dados para os experimentos subsequentes, a opção foi pela versão "SMOTE100", que obteve o melhor desempenho na métrica F_1-*score*, atingindo 82,6%. Contudo, também foi considerada a utilização de dois conjuntos de dados, ao invés de apenas um. Nesse sentido, a versão "DadosSP" foi a escolhida, não somente devido ao seu desempenho positivo em várias métricas, mas também pela sua característica real e não sintéticas dos dados.

Assim, para os experimentos subsequentes, foram adotados dois conjuntos de dados, as versões "SMOTE100" e "DadosSP". Essa escolha permitirá uma análise mais abrangente da arquitetura da CNN-1D em diferentes contextos de dados.

[14]Forma de regularização usada para evitar *overfitting* ao treinar um modelo.

4.2. Resultados dos Testes de Parâmetros: Otimizador e Função de Ativação

Nesta subseção, são apresentados os resultados das diferentes combinações de otimizadores e funções de ativação aplicadas à CNN-1D, mantendo os parâmetros padrões que são exibidos na Tabela 2. Na Subsubseção 4.2.1, são discutidos os resultados com o conjunto de dados "SMOTE100", enquanto na Subsubseção 4.2.2, são abordados os resultados com o conjunto de dados "DadosSP".

A Tabela 2 apresenta os otimizadores testados, juntamente com seus parâmetros, que foram mantidos no padrão para este estudo. Na primeira coluna, estão listados os cinco otimizadores utilizados: Adadelta[15], Adagrad[16], Adam[17], RMSProp[18] e SGD[19]. Na segunda coluna, são apresentados seus parâmetros.

Tabela 2. Parâmetros utilizados neste trabalho.

Otimizador	Parâmetros
Adadelta	lr=1.0, rho=0.9, eps=1e-06, weight_decay=0
Adagrad	lr=0.01, lr_decay=0, weight_decay=0, eps=1e-10
Adam	lr=0.001, betas=(0.9, 0.999), eps=1e-08, weight_decay=0
RMSProp	lr=0.01, alpha=0.99, eps=1e-08, weight_decay=0, momentum=0
SGD	lr=1e-2, momentum=0, dampening=0, weight_decay=0

4.2.1. Considerando o conjunto de dados com a versão "SMOTE100"

As Tabelas 3 e 4 apresentam os resultados dos experimentos variando os otimizadores e funções de ativação para o conjunto de dados com a versão "SMOTE100". A primeira coluna lista os cinco otimizadores utilizados. Nas colunas subsequentes, são exibidos os resultados para as diferentes combinações destes otimizadores com as quatro funções de ativação: Celu[20], Relu[21], Sigmoid[22] e Tanh[23], avaliadas em termos de F_1-*score* na Tabela 3 e função de perda na Tabela 4 no conjunto de dados "SMOTE100". As três melhores pontuações em cada métrica estão destacadas em tons de cinza, sendo o mais escuro o melhor valor.

As Tabelas 3 e 4 revelam que a variação nos otimizadores e funções de ativação tiveram impacto significativo nos resultados da CNN-1D para este conjunto de dados, mesmo utilizando os parâmetros padrões. Observa-se que o otimizador RMSProp obteve consistentemente os melhores resultados em termos de F_1-*score* e função de perda, independentemente da função de ativação escolhida. No entanto, ele apresenta uma melhora quando combinado com a função de ativação Tanh. Em contraste, o otimizador SGD teve os piores resultados em todas as funções de ativação, ficando consideravelmente abaixo das outras combinações em ambas as métricas testadas.

[15] https://pytorch.org/docs/stable/generated/torch.optim.Adadelta.html?highlight=adadelta#torch.optim.Adadelta

[16] https://pytorch.org/docs/stable/generated/torch.optim.Adagrad.html#torch.optim.Adagrad

[17] https://pytorch.org/docs/stable/generated/torch.optim.Adam.html#torch.optim.Adam

[18] https://pytorch.org/docs/stable/generated/torch.optim.RMSprop.html#torch.optim.RMSprop

[19] https://pytorch.org/docs/stable/generated/torch.optim.SGD.html#torch.optim.SGD

[20] https://pytorch.org/docs/stable/generated/torch.nn.functional.celu.html?highlight=functional+celu#torch.nn.functional.celu

[21] https://pytorch.org/docs/stable/generated/torch.nn.functional.relu.html#torch.nn.functional.relu

[22] https://pytorch.org/docs/stable/generated/torch.nn.functional.sigmoid.html#torch.nn.functional.sigmoid

[23] https://pytorch.org/docs/stable/generated/torch.nn.functional.tanh.html#torch.nn.functional.tanh

Tabela 3. Tabela de resultados das comparações de otimizador e função de ativação em termos de F_1-*score* no conjunto de dados "SMOTE100".

Otimizador	Função de Ativação			
	Celu	Relu	Sigmoid	Tanh
Adadelta	0.829	0.825	0.832	0.845
Adagrad	0.827	0.820	0.829	0.847
Adam	0.826	0.824	0.840	0.854
RMSProp	0.844	0.852	0.852	0.861
SGD	0.811	0.809	0.783	0.789

Tabela 4. Tabela de resultados das comparações de otimizador e função de ativação em termos de função de perda no conjunto de dados "SMOTE100".

Otimizador	Função de Ativação			
	Celu	Relu	Sigmoid	Tanh
Adadelta	0.412	0.416	0.397	0.374
Adagrad	0.416	0.431	0.400	0.371
Adam	0.416	0.420	0.386	0.365
RMSProp	0.392	0.399	0.359	0.346
SGD	0.439	0.445	0.469	0.463

No que diz respeito às funções de ativação, a função Tanh demonstrou consistentemente os melhores resultados em termos de F_1-*score* e função de perda. Ela liderou em quase todas as combinações com os otimizadores, exceto quando o SGD foi utilizado. É importante notar que, neste experimento, não foi possível determinar claramente qual das funções de ativação teve o desempenho mais fraco, uma vez que as diferenças entre elas não ficam tão acentuadas.

Contudo, fica evidente que o otimizador que está sendo combinado à função de ativação tem um impacto mais significativo nos resultados, e que a função de ativação Tanh exerceu consistentemente uma influência positiva sobre esses resultados.

A combinação que apresentou melhor desempenho neste conjunto de dados foi o otimizador RMSProp com a função de ativação Tanh com 86,1% na métrica de F_1-*score* e apenas 34,6% na métrica de função de perda (é importante destacar que quanto menor for a função de perda, melhor o desempenho). Em segundo lugar, a combinação Adam-Tanh com 85,4% de F_1-*score* e uma função de perda de 36,5%. E em terceiro lugar, têm-se a combinação RMSProp-Sigmoid, alcançando 85,2% de F_1-*score* e uma função de perda de 35,9%, que foi responsável para sua colocação nesta posição. A combinação que apresentou o pior desempenho foi o SGD-Sigmoid com 78,3% de F_1-*score* e 46,9% de função de perda.

Portanto, para o conjunto de dados "SMOTE100" a melhor combinação de otimizador e função de ativação foi RMSProp-Tahn.

4.2.2. Considerando o conjunto de dados com a versão "DadosSP"

As Tabelas 5 e 6 apresentam os resultados dos experimentos variando os otimizadores e funções de ativação para o conjunto de dados com a versão "DadosSP". A primeira coluna lista os cinco otimizadores testados. Nas colunas subsequentes, são exibidos os resultados para as diferentes combinações de otimizadores com as funções de ativação, avaliadas em termos de F_1-*score* na

Tabela 5 e função de perda na Tabela 6 sobre o conjunto de dados "DadosSP". As três melhores pontuações em cada métrica estão destacadas em tons de cinza, sendo o mais escuro o melhor valor.

Tabela 5. Tabela de resultados das comparações de otimizador e função de ativação em termos de F_1-*score* no conjunto de dados "DadosSP".

Otimizador	Função de Ativação			
	Celu	Relu	Sigmoid	Tanh
Adadelta	0.811	0.816	0.817	0.817
Adagrad	0.816	0.816	0.814	0.817
Adam	0.818	0.821	0.818	0.823
RMSProp	0.819	0.817	0.816	0.819
SGD	0.794	0.793	0.784	0.789

Tabela 6. Tabela de resultados das comparações de otimizador e função de ativação em termos de função de perda no conjunto de dados "DadosSP".

Otimizador	Função de Ativação			
	Celu	Relu	Sigmoid	Tanh
Adadelta	0.392	0.388	0.388	0.383
Adagrad	0.390	0.386	0.389	0.388
Adam	0.384	0.385	0.385	0.379
RMSProp	0.379	0.381	0.382	0.377
SGD	0.417	0.415	0.442	0.444

Ao analisar os resultados, percebe-se que a variação das combinações de otimizadores e funções de ativação não tiveram um impacto significativo no desempenho da CNN-1D para este conjunto de dados, com valores bastante semelhantes. A exceção fica por conta do otimizador SGD, que obteve os piores resultados.

A combinação que apresentou o melhor desempenho neste conjunto de dados foi a utilização do otimizador Adam com a função de ativação Tanh, alcançando um F_1-*score* de 82,3% e uma função de perda de 37,9%. Em seguida, a combinação de Adam-Relu obteve 82,1% de F_1-*score* e uma função de perda de 38,5%, enquanto a combinação de RMSProp-Tanh alcançou 81,9% de F_1-*score* e 37,7% de função de perda. A combinação com pior desempenho foi novamente a do otimizador SGD junto com a função de ativação Sigmoid, resultando em um F_1-*score* de 78,4% e uma função de perda de 44,2%.

Portanto para o conjunto de dados "DadosSP" a melhor combinação de otimizador e função de ativação foi Adam-Tanh.

4.3. Resultados comparando a CNN-1D e a técnica AdaBoost

Com base nos experimentos da Subseção 4.2, a melhor configuração da rede CNN-1D, considerando otimizador e função de ativação, testadas neste trabalho, para o conjunto de dados "SMOTE100" é a combinação RMSProp-Tanh e para o conjunto de dados "DadosSP" é a combinação Adam-Tanh. Então, esses resultados foram comparados com as versões correspondentes destes conjuntos de dados da IC, onde foi utilizado um método clássico de AM, o classificador *AdaBoost*, com hiper-parâmetros definidos por um otimizador que funciona testando todas as combinações possíveis dos hiper-parâmetros, exaustivamente para cada conjunto de dados. Vale

destacar que o conjunto de dados "SMOTE100" também demonstrou o melhor desempenho na IC.

Os resultados dessa comparação em termos de F_1-*score* são apresentados na Tabela 7. A primeira coluna mostra os dois conjuntos de dados. A segunda coluna exibe os resultados da CNN-1D na configuração (RMSProp-Tanh) para o conjunto de dados "SMOTE100" e (Adam-Tanh) no conjunto "DadosSP". A terceira coluna exibe os resultados da IC com o classificador *AdaBoost* (é importante salientar que este classificador não fez uso de técnicas de seleção de características).

Tabela 7. Tabela de resultados para comparação entre a melhor configuração da CNN 1-D para cada conjunto de dados com os resultados do *AdaBoost* (IC) em termos de F_1-*score*.

Conjunto de Dados	CNN-1D	*AdaBoost* (IC)
SMOTE100	0.861	0.854
DadosSP	0.823	0.759

Ao analisar os resultados, observa-se que CNN-1D superou o *AdaBoost* em ambos os conjuntos de dados. No conjunto "SMOTE100", a CNN-1D configurada com a combinação RMSProp-Tanh alcançou um F_1-*score* de 86,1%, enquanto o *AdaBoost* obteve 85,4%. No conjunto de dados "DadosSP", a CNN-1D configurada com a combinação Adam-Tanh apresentou um desempenho ainda mais significativo em relação ao *AdaBoost*, com um F_1-*score* de 82,3%, enquanto o *AdaBoost* atingiu apenas 75,9%.

Além disso, é importante destacar que a CNN-1D demonstrou uma consistência em seus resultados nas diferentes versões do conjunto de dados, enquanto o *AdaBoost* mostrou-se mais sensível aos dados sintéticos, exibindo maior variação entre seus resultados.

Portanto, a CNN-1D, mesmo com parâmetros padrões em seu otimizador, apresenta um desempenho superior ao *AdaBoost*, com parâmetros otimizados, na tarefa de classificação nos conjuntos de dados analisados. Essa superioridade não apenas se traduz em F_1-*scores* mais elevados, mas também na consistência de seus resultados diante de dados reais ou dados reais mistos com sintéticos.

5. Considerações Finais

Neste estudo, uma abordagem baseada em uma CNN-1D foi aplicada a dados tabulares para desenvolver um modelo de prognóstico para pacientes com Covid-19, utilizando o *framework PyTorch*.

Foram realizados testes em dois conjuntos de dados diferentes, variando em termos de balanceamento, e uma série de experimentos foram conduzidos, explorando diferentes otimizadores e funções de ativação.

Nos resultados, destacou-se que, para o conjunto de dados "DadosSP", a combinação do otimizador RMSProp com a função de ativação Tanh demonstrou o melhor desempenho. No entanto, ao tratar do conjunto de dados "SMOTE100", os experimentos indicaram que o otimizador Adam, em conjunto com a função de ativação Tanh, foi a configuração mais eficaz.

Além disso, ao comparar os resultados com a técnica de classificação *AdaBoost* desenvolvida no projeto de Iniciação Científica, observou-se que a CNN-1D superou o *AdaBoost* em ambos os conjuntos de dados e também demonstrou maior consistência em seus resultados, independentemente da natureza dos dados, real ou misto com instâncias sintéticas.

Este estudo sugere que a abordagem da CNN-1D pode ser promissora para análise de dados tabulares. No entanto, são necessárias mais pesquisas para avaliar seu verdadeiro potencial.

É importante ressaltar que a otimização de parâmetros em redes neurais é mais complexa e o treinamento é mais demorado em comparação com os métodos clássicos de AM.

A análise do comportamento de um algoritmo de AP aplicado a dados tabulares, especialmente relacionados à Covid-19, foi a contribuição deste trabalho para a crescente literatura científica na área.

Como trabalhos futuros, pode-se realizar testes com conjuntos de dados maiores, assim como a otimização de parâmetros da rede e também a aplicação de outras arquiteturas de RNAs neste conjunto de dados. Um caminho possível também seria a exploração de modelos de IA explicável a fim de entender as diversas escolhas da rede neural.

Referências

Arel, I., Rose, D. C., and Karnowski, T. P. (2010). Research frontier: Deep machine learning–a new frontier in artificial intelligence research. *Comp. Intell. Mag.*, 5(4):13–18.

Batista, A. F. M. and Chiavegatto, A. D. P. (2019). Machine learning aplicado à saúde. In *Workshop: Machine Learning*. Sociedade Brasileira de Computação. Acesso em: 24 mai 2023.

Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., and Kasneci, G. (2022). Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, PP:1–21.

Bragilovski, M., Kapri, Z., Rokach, L., and Levy-Tzedek, S. (2023). Tltd: Transfer learning for tabular data. *Applied Soft Computing*, page 110748.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA. Association for Computing Machinery.

Elkin, M. E. and Zhu, X. (2022). A machine learning study of covid-19 serology and molecular tests and predictions. *Smart Health*, 26:100331.

Géron, A. (2019). *Mãos à Obra: Aprendizado de Máquina com Scikit-Learn & TensorFlow*. Alta Books.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Guarrasi, V. and Soda, P. (2023). Multi-objective optimization determines when, which and how to fuse deep networks: An application to predict covid-19 outcomes. *Computers in Biology and Medicine*, 154:106625.

Guo, T., Dong, J., Li, H., and Gao, Y. (2017). Simple convolutional neural network on image classification. In *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, pages 721–724.

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A. (2019). Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(3):917–963.

Khan Mamun, M. M. R. and Elfouly, T. (2023). Detection of cardiovascular disease from clinical parameters using a one-dimensional convolutional neural network. *Bioengineering*, 10(7).

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Lemaître, G., Nogueira, F., and Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5.

Li, M., Esfahani, F., Xing, L., and Zhang, X. (2023a). Predicting the daily counts of covid-19 infection using temporal convolutional networks. *Journal of Global Health*, 13:03029.

Li, X., Guo, H., Xu, L., and Xing, Z. (2023b). Bayesian-based hyperparameter optimization of 1d-cnn for structural anomaly detection. *Sensors*, 23(11).

Lima Neto, A. N. (2023). Classificação de dados de covid-19 usando rede neural convolucional. Trabalho de Conclusão de Curso. Acesso Aberto.

Ministério da Saúde (2023). Painel de casos de doença pelo coronavírus (covid-19) no brasil pelo ministério da saúde.

Monshi, M. M. A., Poon, J., Chung, V., and Monshi, F. M. (2021). Covidxraynet: Optimizing data augmentation and cnn hyperparameters for improved covid-19 detection from cxr. *Computers in Biology and Medicine*, 133:104375.

Mueller, J. P. (2020). *Aprendizado Profundo para Leigos*. Editora Alta Books, Rio de Janeiro, e-book edition.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Saúde, M. D. (2021). O que é a covid-19? https://www.gov.br/saude/pt-br/coronavirus/o-que-e-o-coronavirus. Accessed: 2023-05-23.

Shaikh, K., Krishnan, S., and Thanki, R. (2021). Breast cancer detection and diagnosis using ai. In *Artificial Intelligence in Breast Cancer Early Detection and Diagnosis*, pages 79–92. Springer.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning - From Theory to Algorithms.* Cambridge University Press.

Sharma, A. and Kumar, D. (2022). Classification with 2-d convolutional neural networks for breast cancer diagnosis. *Scientific Reports*, 12.

Shwartz-Ziv, R. and Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90.

Sofia, Malik, A., Shabaz, M., and Asenso, E. (2023). Machine learning based model for detecting depression during covid-19 crisis. *Scientific African*, 20:e01716.

Tarwidi, D., Pudjaprasetya, S. R., Adytia, D., and Apri, M. (2023). An optimized xgboost-based machine learning method for predicting wave run-up on a sloping beach. *MethodsX*, 10:102119.

Toledo, M. (2022). Covid em 2022: queda de mortes, aumento de casos, autotestes e descobertas. https://www.cnnbrasil.com.br/saude/covid-em-2022-queda-de-mortes-aumento-de-casos-autotestes-e-descobertas/. Acesso em: 31 maio 2023.

Ullah, I., Rios, A., Gala, V., and Mckeever, S. (2022). Explaining deep learning models for tabular data using layer-wise relevance propagation. *Applied Sciences*, 12(1):136.

Zhu, Y., Brettin, T., Xia, F., Partin, A., Shukla, M., Yoo, H., Evrard, Y. A., Doroshow, J. H., and Stevens, R. L. (2021). Converting tabular data into images for deep learning with convolutional neural networks. *Scientific Reports*, 11(1):11325.