Predição pluviométrica utilizando TensorFlow e Node.js

Marcio Almeida de Lima, Ricardo Frohlich da Silva
Curso de Sistemas de Informação
UFN - Universidade Franciscana
Santa Maria - RS
marcio.almeida@ufn.edu.br, ricardo.frohlich@ufn.edu.br

***Resumo*—As mudanças climáticas têm provocado eventos extremos, como as enchentes que afetaram o Rio Grande do Sul em 1941 e 2024. Este trabalho apresenta uma abordagem baseada em Redes Neurais Artificiais (RNA) do tipo LSTM para previsão de precipitação na região metropolitana de Porto Alegre, utilizando dados meteorológicos do INMET entre 2000 e 2024. A metodologia envolve coleta, normalização e modelagem de séries temporais, com desenvolvimento em Node.js e TensorFlow, e monitoramento dos experimentos via TensorBoard. Os resultados demonstram que, embora o modelo apresente limitações na precisão dos valores absolutos, ele consegue capturar tendências e padrões de variação nas séries temporais, evidenciando o potencial do uso de RNAs na previsão climática como apoio à gestão de riscos e desastres naturais.**

***Palavras-chave*: Redes Neurais Artificiais, Previsão Climática, Mudanças Climáticas, LSTM.**

I. Introdução

As mudanças climáticas e seus impactos têm se tornado uma preocupação crescente. Esse fenômeno afeta não apenas o meio ambiente, mas também a vida humana e as atividades econômicas. Com o passar dos anos, eventos como desastres naturais mais frequentes e alterações nos ecossistemas despertaram a atenção de governantes. Tais indícios evidenciam os riscos da exploração ambiental desenfreada à sobrevivência humana e ao equilíbrio da biodiversidade [1].

No Brasil, as recentes enchentes que atingiram o estado do Rio Grande do Sul ilustram a gravidade dessas tragédias climáticas. O fenômeno impactou milhões de pessoas, interrompendo a produção de indústrias importantes, como a metalmecânica, alimentícia, calçadista e química, além de causar grandes prejuízos ao setor agropecuário. Esses danos não afetaram apenas a economia local, mas também geraram consequências negativas para a cadeia produtiva em nível nacional [2].

Diante disso, torna-se essencial o uso de ferramentas capazes de modelar a imprevisibilidade e a não-linearidade desses fenômenos. Para isso, as Redes Neurais Artificiais (RNAs) se destacam como uma solução promissora para abordar este tema, pois são capazes de modelar e prever processos complexos e não-lineares, como o comportamento das chuvas e enchentes (chuva-vazão). As RNAs são conhecidas por sua capacidade de previsão, sendo particularmente úteis quando a precisão das estimativas é mais importante que a descrição física detalhada do processo. Além disso, elas se destacam ao lidar com grandes volumes de dados históricos e evitam a complexidade que a modelagem física de processos não-lineares exigiria. Uma vez treinadas, as RNAs conseguem generalizar e fornecer previsões mesmo para eventos que não foram observados diretamente, tornando-se uma ferramenta valiosa para apoiar decisões e ações corretivas [3].

As redes neurais artificiais apresentam benefícios devido ao ótimo desempenho e a simples implementação, uma vez desenvolvidas. Atribuído às suas características inerentes de aprendizado, classificação e generalização de informação, o ponto forte das RNAs é reconhecido na literatura científica e vem sendo aperfeiçoado há mais de três décadas [3].

A. Motivação

Este trabalho tem como motivação contribuir para a redução dos danos e a prevenção de perdas de vidas em decorrência de eventos climáticos extremos. Ao identificar e analisar padrões climáticos, torna-se possível monitorar com maior precisão esses fenômenos e prever, com antecedência, a ocorrência de desastres naturais. Dessa forma, ações preventivas podem ser implementadas, como evacuações e medidas de contenção, minimizando os impactos sobre as comunidades afetadas.

B. Objetivo geral

O objetivo geral deste trabalho é analisar os padrões de chuva na região metropolitana de Porto Alegre - RS, com foco no período de 2000 a 2024, para identificar possíveis tendências que contribuam para a previsão de eventos extremos. A escolha dessa região é fundamentada nas enchentes ocorridas em 1941 e 2024, cujos danos de grande escala geraram impactos sociais e econômicos de longa duração.

C. Objetivos específicos

- Coletar e organizar dados meteorológicos históricos do INMET referentes ao estado do Rio Grande do Sul, no período de 2000 a 2024.
- Normalizar os dados e prepará-los para uso em modelos de séries temporais.

- Implementar um modelo de Rede Neural Artificial do tipo LSTM (Long Short Term Memory) para previsão de precipitação.
- Avaliar o desempenho do modelo considerando diferentes abordagens, com e sem ponderação das variáveis de entrada.
- Analisar a precisão das previsões por meio de métricas estatísticas como MAE (Erro Absoluto Médio) e MSE (Erro Quadrático Médio).
- Comparar os resultados preditivos do modelo com os dados reais do evento climático extremo ocorrido em maio de 2024, na região metropolitana de Porto Alegre.
- Documentar os resultados, desafios e limitações, propondo melhorias para trabalhos futuros.

II. Referencial Teórico

O embasamento teórico consiste nos conceitos e ferramentas empregadas para a realização do trabalho. A seguir, são apresentados conceitos de inteligência artificial e a subárea de Redes Neurais Artificiais, juntamente com as ferramentas que serão utilizadas.

A. Redes Neurais Artificiais

No âmbito de Inteligência Artificial (IA), encontra-se as RNAs, mais precisamente no ramo de *Deep Learning* (Aprendizado Profundo). As RNAs são modelos computacionais inspirados na estrutura neural de organismos inteligentes, como é exemplificado na Figura 1. Seu comportamento inteligente surge das interações entre as unidades de processamento, a partir de seu ambiente, através de um processo de aprendizagem, cuja função é alterar os pesos sinápticos da rede. A partir disso, este conhecimento é disponibilizado para a aplicação em questão [4]. Como a RNA utiliza da organização do cérebro humano,

> "ela é formada por unidades de processamento, comumente chamadas de nós, neurônios ou células, interconectadas por arcos unidirecionais, também chamados de ligações, conexões ou sinapses (...). Cada célula possui uma única saída (axônio), a qual pode se ramificar em muitas ligações colaterais (...)" [5].

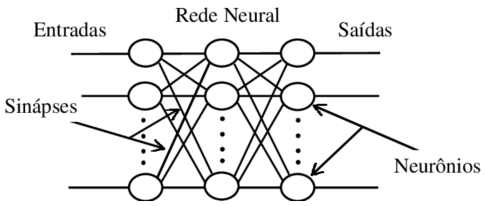

Figura 1. Representação de uma Rede Neural Artificial.

De acordo com Aires, Dametto e Crepaldi [6], um neurônio pode ser representado como mostrado na Figura 2, onde os sinais de entrada externos x_i são multiplicados pelos pesos sinápticos w_i, que variam conforme a importância de cada x_i. Em seguida, realiza-se uma soma ponderada dos sinais de entrada, feita pelo combinador linear $\sum$, e subtrai-se a variável θ, chamada de limiar de ativação. Dessa forma, obtém-se o potencial de ativação u, que representa a saída inicial de cada neurônio. As saídas dos neurônios artificiais são então passadas por uma função de ativação $g()$, cuja função é limitar a saída do neurônio a um intervalo específico, gerando o sinal final de saída y.

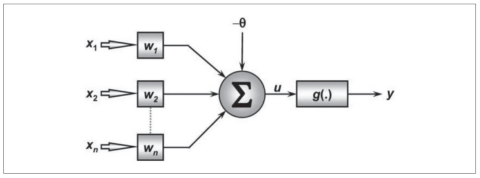

Figura 2. Neurônio Artificial.
Fonte: [6].

B. Rede Neural Long Short Term Memory

A rede neural LSTM é selecionada como modelo de previsão para séries temporais. A rede LSTM, que é uma versão especial da RNN (*Recurrent Neural Network*), possui uma cadeia de módulos repetitivos, assim como as RNNs. Nas redes LSTM, cada módulo, chamado de célula de memória, contém 3 portas diferentes: uma porta de entrada, uma porta de saída e uma porta de esquecimento. Essas portas controlam o fluxo de informações, e cada uma dessas portas utiliza funções de ativação como *sigmoid* e tanh para regular as informações na célula de memória. Esta célula de memória está localizada na camada oculta da rede LSTM [7], como pode ser visto na Figura 3.

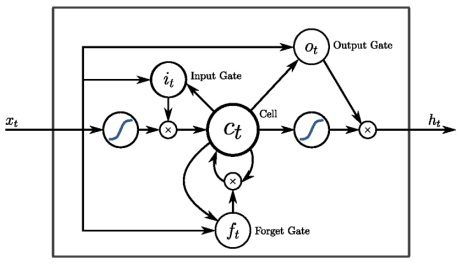

Figura 3. Estrutura de neurônios na LSTM. Fonte: [7]

C. TensorFlow

O TensorFlow é uma biblioteca *open source* mantida pela Google. Pode ser utilizada em Python, JavaScript, C++, Java, Go e Haskell. Criado em 2015, a Google dedicou uma parte da sua equipe de IA para sua implementação, com intuito de flexibilizar e melhorar seu desempenho em

comparação ao *DistBelief*, biblioteca que deu origem ao TensorFlow. O TensorFlow utiliza cálculos descritos em um modelo de fluxo de dados e os mapeia em grande variedade de hardware e plataformas diferentes, desde execução de inferência em dispositivos móveis, máquinas com múltiplas GPUs (*Graphic Processor Unit*) e processamento de centenas de máquinas especializadas com milhares de GPUs [8]. A biblioteca possui várias funcionalidades para a implementação de Redes Neurais, tais quais: funções de ativação, treinamento, criação de camadas e neurônios. A estrutura básica do TensorFlow consiste em:

- **Tensor**: estrutura de dados mais básica, sendo um vetor multi dimensional;
- **Grafo**: o cálculo é descrito por um grafo direcionado e é composto por um conjunto de nós, na qual são ligados uns aos outros, formando uma rede;
- **Sessão**: é um agrupamento dos nós do grafo, criando camadas de computação, podendo executar cada sessão em uma *thread* separada ou dispositivo.

D. Node.js

É um software que executa código JavaScript em ambiente de *server-side* (lado do servidor). Sua implementação é baseada no runtime JavaScript da Google, chamado de *V8 engine* [9]. Segundo Stefan Tilkov e Steve Vinoski [9], um processo Node.js é executado em *single-thread* e não depende de *threads* para execução concorrente. Ao contrário de muitos outros sistemas, o Node.js realiza suas operações a partir do conceito conhecido como arquitetura orientada a eventos.

E. Scrum Solo

O Scrum Solo é uma adaptação do *framework* Scrum para uso individual, onde uma única pessoa assume todos os papéis de *Scrum Master*, *Product Owner* e Time de Desenvolvimento. Os princípios ágeis, como trabalho iterativo, entregas incrementais e priorização de tarefas, são mantidos nesta metodologia, mas ela é simplificada para refletir as necessidades de um desenvolvedor solo. Além disso, o ciclo de *sprints* é ajustado para auto-organização, com foco em planejamento, execução e retrospectiva pessoal, permitindo controle efetivo do progresso e flexibilidade nas entregas.

No Scrum Solo, é proposto que as *sprints* tenham durações reduzidas a uma semana e não existam reuniões diárias; ao final de cada *sprint*, de forma equivalente ao Scrum, deve ser entregue, pelo desenvolvedor, um protótipo do software com novas funcionalidades e, podem existir, quando necessário, reuniões de orientação entre o grupo de validação (clientes e usuários finais) e o desenvolvedor [10].

Figura 4. Fluxo do processo Scrum Solo [10].

F. Kanban

Kanban é uma metodologia ágil de gestão de projetos que se concentra na visualização do fluxo de trabalho e na melhoria contínua. Originada no sistema de produção da Toyota, o Kanban utiliza cartões ou quadros para representar tarefas e seu progresso, permitindo que equipes visualizem rapidamente o estado do trabalho em andamento.

Kanban significa ”etiqueta” ou ”cartão”. O sistema de Kanban é usado como um meio de controle e coordenação e deve satisfazer algumas funções, como por exemplo: indicar as ordens de produção o que, quanto e quando produzir e para onde levar esses produtos. O número de ”cartões” controla a quantidade de bens que esta atravessando a produção, e tornam possível uma resposta mais flexível a variações de demanda através da simplificação de instruções de produção [11].

G. Feature Driven Development

A *Feature-Driven Development* (FDD) é uma metodologia de desenvolvimento de software que inclui alguns benefícios de processos rigorosos, como modelagem, planejamento prévio e controle do projeto, assim como contém características de processos ágeis, como foco na programação, interação constante com o cliente e entrega frequente de versão do produto. Prevê práticas apenas para o desenvolvimento de software em si não se preocupando com outros fatores como a escolha de tecnologias e ferramentas.

São cinco os processos da metodologia ágil FDD: Desenvolver um modelo abrangente, construir uma lista de funcionalidades, planejar através de funcionalidades, projetar através de funcionalidades e construir através de funcionalidades [12]. Na Figura 5, é possível visualizar a interação entre si desses processos.

1) **Desenvolver um modelo abrangente**: envolve criar uma visão geral da arquitetura do sistema, a fim de estabelecer uma base comum entre os desenvolvedores e identificar as principais áreas de funcionalidade.
2) **Construir uma lista de funcionalidades**: divide o sistema em funcionalidades de negócio específicas e pequenas.
3) **Planejar através de funcionalidades**: organiza as funcionalidades em um cronograma com estimativas

de tempo e complexidade, ajudando a priorizar entregas.

4) **Projetar através de funcionalidades**: cada funcionalidade é detalhada com um design específico, considerando padrões e melhores práticas, e com revisão da equipe antes da implementação.
5) **Construir através de funcionalidade**: é o desenvolvimento propriamente dito, com testes e integrações contínuas.

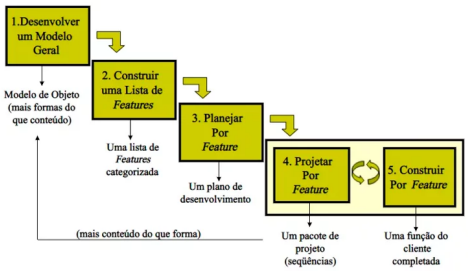

Figura 5. Diagrama do fluxo e interação do FDD. Fonte: [13].

III. Trabalhos correlatos

Nesta seção encontram-se alguns trabalhos relacionados ao tema escolhido.

A. Previsão pluviométrica por meio da aplicação de Redes Neurais Artificiais Recorrentes alimentadas com dados meteorológicos em tempo atual

O trabalho de Jordana Wilm Doninelli et al. [14] propõe a utilização de Redes Neurais Artificiais Recorrentes (RNN) com arquitetura LSTM para prever precipitações de curto prazo. A rede foi treinada com dados meteorológicos de uma estação em Okinawa, Japão, abrangendo um período de 10 anos (2010-2020). Foram usados dados de umidade, temperatura, pressão, vento e precipitação, aplicando a transformação *Box-Cox* para ajustar a distribuição dos dados.

A rede foi estruturada com seis camadas e treinada utilizando o algoritmo *Adam* e a função de perda *binary crossentropy*. O desempenho foi avaliado com métricas de acurácia, precisão e recall, alcançando resultados promissores, especialmente em termos de recall (aproximadamente de 90%), o que indica a eficácia do modelo em prever eventos de chuva, minimizando falsos negativos.

Os resultados confirmam que o modelo LSTM consegue aprender padrões temporais e apresentar uma boa previsão para chuvas, sendo semelhante a estudos anteriores, mesmo com a complexidade envolvida na previsão de chuvas.

B. Redes Neurais Artificiais aplicadas à previsão antecipada de precipitações na Região Central de Manaus

O estudo realizado por Rafaela dos Santos Sousa et al. [15] investiga a utilização de RNA para prever, com 30 minutos de antecedência, a ocorrência de precipitações na região central de Manaus, utilizando dados de estações meteorológicas ao redor da cidade. Um total de 523 redes neurais foi proposto, sendo que duas se destacaram por apresentar melhor acurácia e precisão. O treinamento adicional com balanceamento de classes resultou em um aumento na precisão dos testes. Os resultados sugerem que as RNAs são ferramentas eficazes para a previsão de chuvas, podendo auxiliar órgãos públicos na tomada de decisões estratégicas para minimizar os impactos negativos das precipitações, como alagamentos.

Foram utilizados dados meteorológicos coletados ao longo de 4 anos por três estações na região de Manaus. O treinamento envolveu a criação de 523 redes *multilayer perceptron*, utilizando o algoritmo de backpropagation com otimização Levemberg-Marquadt. A acurácia média alcançada foi de 97%, embora a precisão necessitasse de melhorias devido ao desbalanceamento de classes.

Os resultados são promissores, as melhores redes alcançaram 97,26% de acurácia, com uma arquitetura de duas camadas ocultas. Um segundo treino com dados balanceados melhorou a precisão, mas ainda indicou a necessidade de ajustes.

C. Uso de Redes Neurais Artificiais na previsão da precipitação de períodos chuvosos

Este estudo de Daniel Dantas et al. [16] teve como objetivo estimar a precipitação na estação chuvosa de Diamantina (MG) a partir de dados das estações secas anteriores, utilizando RNA. Foram utilizados dados históricos de precipitação diária de 1977 a 2014, organizados em séries temporais, e a melhor rede encontrada foi do tipo função de base radial. A rede neural apresentou um erro médio de 10%, com uma estimativa de precipitação média de 1.128 mm, em comparação com a média real de 1.099 mm para o período chuvoso. O uso de dados do período seco para prever a precipitação no período chuvoso apresentou resultados satisfatórios, sendo que a alteração da ordem cronológica dos dados resultou em uma rede com melhor desempenho preditivo.

O estudo utilizou redes neurais treinadas com dados do período seco para prever a precipitação no período chuvoso tanto no mesmo ano quanto no ano seguinte. A rede foi treinada com 50 RNAs para cada situação, selecionando-se a de melhor desempenho. A análise foi realizada utilizando erros relativos médios (ERM%) para avaliar a precisão dos modelos.

Na primeira situação, que usou dados do período seco para prever o período chuvoso do mesmo ano, a rede obteve um ERM% de 15,99%. Na segunda situação, que previu o período chuvoso do ano seguinte, o ERM% foi de 13,36%, indicando uma melhor performance. A alteração na ordem cronológica dos dados resultou em uma leve melhoria na

eficácia da rede, sugerindo que este ajuste pode aprimorar previsões futuras.

O uso de RNA mostrou-se uma ferramenta promissora para a previsão da precipitação em períodos chuvosos, com resultados satisfatórios, especialmente quando se utilizou a alteração da ordem cronológica dos dados. A pesquisa sugere que o desempenho das RNAs pode ser aprimorado com a inclusão de mais variáveis meteorológicas, o que pode torná-las uma ótima ferramenta para a previsão climática e o planejamento de atividades dependentes do clima, como a agricultura.

IV. Metodologia

Para a gestão do projeto, optou-se pela combinação das metodologias ágeis Scrum Solo e Kanban, que permitem um controle eficiente e visual das tarefas. Para o planejamento, divisão e priorização das atividades do projeto, foi escolhida a metodologia FDD. Essas três metodologias se complementam e tornam a execução do projeto mais clara e objetiva.

O Scrum Solo, adaptado para o trabalho individual, facilita a organização de sprints curtos e o acompanhamento contínuo do progresso em ciclos incrementais, permitindo uma abordagem ágil e ajustável às necessidades de cada fase. Paralelamente, o Kanban será utilizado para visualizar o fluxo de trabalho e organizar as atividades por prioridades e estágios de execução, desde a coleta e tratamento de dados até a análise dos resultados.

A integração do Scrum Solo e Kanban permite uma gestão flexível e dinâmica, proporcionando maior controle sobre prazos e avanços, e assegurando que ajustes possam ser feitos conforme necessário ao longo do projeto, sem comprometer a qualidade ou o rigor metodológico.

A metodologia escolhida para o planejamento do projeto foi a FDD. Esta abordagem é voltada para o desenvolvimento iterativo e incremental, com foco na entrega de funcionalidades específicas que agregam valor ao produto final. Seu ciclo de vida é constituído de duas fases, distribuídas em cinco processos principais, que são descritos a seguir.

A. Desenvolver um modelo abrangente

De acordo com Foggetti [17], o primeiro processo da FDD consiste em *brainstorming* entre os especialistas de domínio. São realizadas reuniões e sessões para definição do contexto e escopo do projeto, apresentadas propostas e modelos entre as equipes, e ao final, são escolhidos em comum quais modelos que devem ser seguidos.

B. Construir uma lista de funcionalidades

Este processo tem como objetivo estabelecer as funcionalidades do projeto. A ordem de definição das funcionalidades não determina sua priorização. A seguir, são listadas as principais funcionalidades necessárias para o projeto.

- Coletar dados pluviométricos do INMET (2000 a 2024).
- Normalizar e armazenar os dados em formato CSV.
- Desenvolver e configurar a RNA para o modelo de previsão.
- Treinar a RNA com os dados de 2000 a 2019.
- Realizar previsões para o período de 2020 a 2024.
- Visualizar o treinamento e os resultados de previsão no TensorBoard.
- Analisar a precisão das previsões da RNA comparando com os dados reais que compreendem o período de 30 de abril de 2024 a 31 de maio de 2024.

A lista de funcionalidades mencionada acima pode ser visualizada na Figura 6.

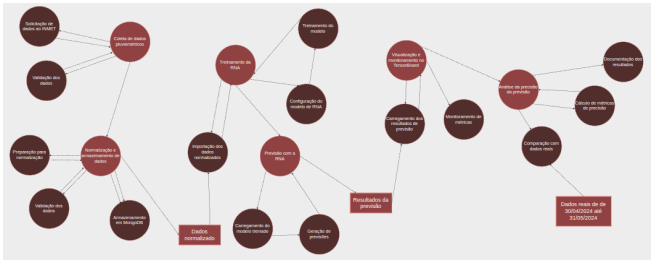

Figura 6. Diagrama de fluxo de dados nível 2. Elaboração própria.

C. Planejar através de funcionalidades

As funcionalidades listadas são distribuídas entre os membros da equipe com base em habilidades e disponibilidade, e realizando o planejamento da ordem e priorização das funcionalidades.

- **Sprint 1**: coleta dos dados pluviométricos disponibilizados no site do INMET.
- **Sprint 2**: normalização dos dados pluviométricos e, posterior, criação do arquivo CSV.
- **Sprint 3**: configuração do projeto em Node.js e instalação do TensorFlow e suas dependências.
- **Sprint 4**: implementação de um modelo de teste da RNA com TensorFlow.
- **Sprint 5**: testes com modelos de RNA LSTM e validação do comportamento.
- **Sprint 6**: realizar a integração do projeto com o TensorBoard.
- **Sprint 7**: alimentar RNA LSTM com os dados pluviométricos do ano 2000 a 2019.
- **Sprint 8**: realizar o treinamento, observar o comportamento e realizar ajustes necessários para tornar os resultados consistentes.
- **Sprint 9**: realizar várias rodadas de treinamento da RNA.
- **Sprint 10**: execução de previsões para o período de 30 de abril de 2024 a 31 de maio de 2024.

- **Sprint 11**: análise dos resultados comparando previsões com os dados reais, criação de relatório de precisão e ajuste final.

D. Projetar através de funcionalidades

Esta etapa consiste no detalhamento técnico de cada funcionalidade previamente planejada, antes do início da implementação. Cada funcionalidade é analisada individualmente com foco em design técnico, modelagem de dados, estrutura de pastas/código.

1) Coleta dos dados pluviométricos: a coleta dos dados foi realizada manualmente por meio da plataforma do INMET, que disponibiliza uma ferramenta para exportação de dados meteorológicos em formato CSV. Durante o processo, foram selecionadas as opções de dados horários provenientes de estações automáticas, abrangendo todas as 45 estações localizadas no estado do Rio Grande do Sul. O intervalo de tempo definido foi de 1º de janeiro de 2000 até 31 de dezembro de 2024. Foram incluídas as 20 variáveis meteorológicas disponíveis na ferramenta que estão listadas a seguir:

- Precipitação total;
- Pressão atmosférica ao nível da estação;
- Pressão atmosférica reduzida ao nível do mar;
- Pressão atmosférica máxima na hora anterior;
- Pressão atmosférica mínima na hora anterior;
- Radiação global;
- Temperatura da CPU na estação;
- Temperatura do ar - bulbo seco;
- Temperatura do ponto de orvalho;
- Temperatura máxima na hora anterior;
- Temperatura mínima na hora anterior;
- Temperatura máxima do ponto de orvalho na hora anterior;
- Temperatura mínima do ponto de orvalho na hora anterior;
- Tensão da bateria da estação;
- Umidade relativa máxima na hora anterior;
- Umidade relativa mínima na hora anterior;
- Umidade relativa do ar;
- Direção do vento;
- Velocidade da rajada máxima do vento;
- Velocidade do vento;

2) Normalização dos dados: antes de alimentar os dados no modelo LSTM, é necessário realizar uma etapa de pré-processamento conhecida como normalização. Os arquivos CSV originais contêm informações sobre cada estação meteorológica, seguidas pela série histórica de dados organizada em ordem cronológica. Para facilitar o processamento das informações, foi necessário separar essas duas partes. Assim, para cada estação, foram gerados dois novos arquivos CSV: um contendo exclusivamente os metadados da estação e outro com os dados históricos normalizados. Com os dados separados, aplicou-se a normalização *MaxMinScaler* aos valores meteorológicos. Essa técnica transforma os valores de cada variável para uma escala entre 0 e 1. A fórmula pode ser vista na Figura 7:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

goog

Figura 7. Fórmula do cálculo da normalização *MinMaxScaler*. Fonte: [18].

Essa normalização é essencial para modelos de redes neurais como a LSTM, pois evita que variáveis com escalas muito diferentes dominem o processo de aprendizado e garante uma convergência mais estável.

3) Treinamento da RNA: o treinamento da RNA consistiu em definir qual variável seria o alvo da previsão — neste caso, a precipitação total horária — e a janela temporal das variáveis de entrada, que foi estabelecida nas últimas 24 horas, uma vez que o objetivo era realizar previsões diárias. Para isso, foram utilizados dados históricos de 2000 até 2019, cobrindo uma faixa temporal significativa que permite ao modelo aprender padrões pluviométricos.

O otimizador utilizado durante o treinamento foi o Adam (*Adaptive Moment Estimation*), conhecido por sua eficiência em problemas envolvendo grandes volumes de dados e parâmetros. Ele combina as vantagens dos métodos AdaGrad e RMSProp, ajustando a taxa de aprendizado de forma adaptativa com base no histórico de gradientes,

O treinamento foi realizado com 5 épocas (epochs), utilizando 128 neurônios na camada escondida e 20% dos dados foram reservados para validação do modelo. O *batch size* utilizado foi de 32, o que significa que a atualização dos pesos da rede ocorre após o processamento de 32 amostras por vez. A camada de saída foi composta por um único neurônio, com função de ativação linear. No caso da LSTM no TensorFlow, não é necessário especificar uma função de ativação para a camada escondida, pois as células LSTM já contêm suas próprias funções de ativação internas.

As métricas utilizadas para avaliação foram o MAE e o MSE. O otimizador utilizado durante o treinamento foi o Adam.

4) Previsão da RNA: para a realização das previsões com a Rede Neural Artificial (RNA), foram selecionados dados do período de 30 de abril de 2024 a 31 de maio de 2024, representando o mês que ocorreu as enchentes no estado do Rio Grande do Sul. Durante o treinamento, foi definida uma janela temporal de 24 horas, o que implica que o modelo considera as últimas 24 horas de dados para gerar uma previsão para o próximo período temporal.

5) Integração com o TensorBoard: a integração com o TensorBoard permite monitorar o desempenho do modelo durante o treinamento e validar suas previsões em tempo real. Para isso, foram definidas as métricas que serão cole-

tadas e visualizadas durante o treinamento e as predições. As métricas escolhidas são:

- Durante o Treinamento e Validação:
  - Gráfico de Perda (loss): exibe o valor da função de perda ao longo das épocas, tanto para o conjunto de treinamento quanto para o conjunto de validação.
  - Gráfico de Monitoramento: exibe a evolução de métricas como o MAE e o MSE, fornecendo uma visão detalhada da precisão do modelo.
- Durante a Predição:
  - Histograma de Predições Diárias e Reais: exibe um histograma das predições feitas pelo modelo e seus valores reais diários, permitindo uma comparação visual entre os valores previstos e reais.
  - Histograma de Média Diária das Predições e Valores Reais: exibe um histograma com a média das predições diárias e a média dos valores reais, ajudando a avaliar o desempenho do modelo ao longo do tempo.
  - Gráfico Escalar de Predição vs Valor Real: exibe um gráfico escalar comparando as predições do modelo com os valores reais, permitindo uma análise visual de quão precisas são as previsões.

E. Construir através de funcionalidade

Após o design da funcionalidade, ela é desenvolvida, testada e integrada ao sistema. Cada funcionalidade passa por um processo de revisão e testes para garantir que esteja alinhada aos requisitos e à qualidade esperada.

1) Normalização dos dados: a normalização dos dados consistiu em processar os arquivos CSV obtidos a partir do site do INMET, realizando a extração e preparação dos valores para uso no modelo preditivo. Essa etapa foi dividida em quatro fases:

- Leitura dos arquivos CSV e geração de novos arquivos, separando os metadados das estações da série histórica de dados;
- Importação dos dados resultantes para um banco de dados MongoDB;
- Conversão dos valores originalmente em formato de texto (*string*) para valores numéricos (*float*);
- Aplicação da normalização utilizando o método *MaxMinScaler*, calculado individualmente para cada variável com base em toda a série histórica disponível.

2) Treinamento da RNA: com os dados devidamente normalizados e estruturados, iniciou-se a etapa de treinamento do modelo da RNA. Essa etapa consistiu em criar o modelo, configurar a arquitetura e definir as métricas de aprendizagem.

Na Figura 8, é possível observar a estrutura dos dados com as janelas temporais que foram utilizadas no treinamento da RNA. Com base no tamanho dessas janelas, foi necessário organizar os dados das variáveis antecedentes de forma que tenham o mesmo tamanho, para possibilitar a previsão que será treinada da variável alvo.

Figura 8. Estruturação dos dados com as janelas temporais. Elaboração própria.

A Figura 9 e Figura 10 demonstram o código de configurações da camada escondida da LSTM e a da camada de saída da rede, respectivamente.

```
model.add(tf.layers.lstm( args: {
    units: 128,
    inputShape: [
        WINDOW_SIZE,
        FEATURES_FOR_SHAPE.length,
    ]
}));
```

Figura 9. Código de configuração da camada LSTM. Elaboração própria.

```
model.add(tf.layers.dense( args: {
    units: 1,
    activation: 'linear',
}));
```

Figura 10. Código de configuração da camada de saída. Elaboração própria.

Em seguida, foram definidas as configurações do treinamento, tais como: o algoritmo otimizador de aprendizagem, função de perda e métrica utilizada para avaliação do desempenho do modelo. A Figura 11 demonstra esta configuração.

```
model.compile( args: {
    optimizer: tf.train.adam(),
    loss: 'meanSquaredError',
    metrics: ["mae"],
});
```

Figura 11. Estruturação dos dados com as janelas temporais. Elaboração própria.

Após a estruturação dos dados, foram passados os dados de entrada (xData) e de saída e (yData) para realizar o treinamento. Também foram informados: a quantidade de épocas a serem realizadas, o *batch size* e a quantidade de dados para validação. A Figura 12 demonstra o código para execução do treinamento.

```
await model.fit(xData, yData,  args: {
    epochs: 5,
    batchSize: 32,
    validationSplit: 0.2,
    callbacks: [
        tf.callbacks.earlyStopping( args: { patience: 3 }),
        tensorBoardCallback
    ]
});
```

Figura 12. Código de execução do treinamento da rede. Elaboração própria.

3) Previsão da RNA: após o treinamento do modelo LSTM com os dados normalizados e organizados em janelas temporais, a próxima etapa consistiu na realização das previsões. Essa etapa teve como objetivo utilizar o modelo treinado para estimar os valores futuros da variável-alvo com base nas sequências de entrada fornecidas.

Para a etapa de predição, foram utilizados os dados compreendidos entre 30 de abril de 2024 e 31 de maio de 2024. Assim como no processo ilustrado na Figura 8, foi necessário aplicar a mesma estruturação em janelas temporais aos dados de entrada. No entanto, diferentemente do treinamento, não foi utilizado a variável-alvo nesse momento, pois é justamente essa a variável que o modelo buscará prever.

Para a execução da predição, foi necessário criar um tensor tridimensional (tensor3d) com os dados de entrada e fornecê-lo à função *predict* do modelo previamente treinado, como demonstrado na Figura 13.

```
const xPredict : Tensor3D  = tf.tensor3d(inputs);
const yPred : Tensor | ...  = model.predict(xPredict);
const predictions : ArrayMap[R]  = await yPred.array();
```

Figura 13. Código de execução do treinamento da rede. Elaboração própria.

4) Análise dos resultados no TensorBoard: o TensorBoard permitiu observar de forma clara e intuitiva a evolução da função de perda (loss), precisão, e outras métricas definidas, além de fornecer recursos adicionais, como comparação entre execuções, visualização de gráficos, histogramas de pesos, e estrutura do grafo computacional do modelo. Na Figura 14 é possível visualizar a interface do TensorBoard.

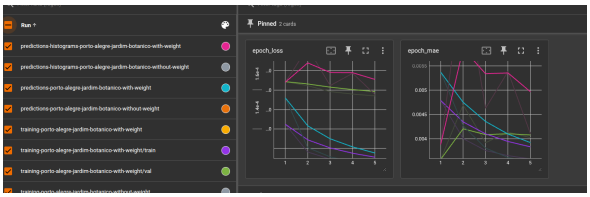

Figura 14. Dashboard do TensorBoard. Elaboração própria.

V. Resultados

Com a conclusão do processo de predição, foi possível obter os valores estimados pela RNA LSTM com base nos dados de entrada fornecidos. Nesta seção, os resultados gerados foram analisados e comparados com os valores reais da variável-alvo no intervalo estudado.

Foram realizados dois treinamentos distintos: (1) utilizando os dados de entrada apenas com a normalização via MinMaxScaler, e (2) aplicando pesos fixos às variáveis de entrada, além da normalização. O objetivo foi comparar o desempenho do modelo em cada um dos cenários propostos.

A. Análise do treinamento

A Figura 15 do gráfico demonstra as curvas de aprendizados para o treinamento sem pesos (linha rosa) e para a validação (linha azul).

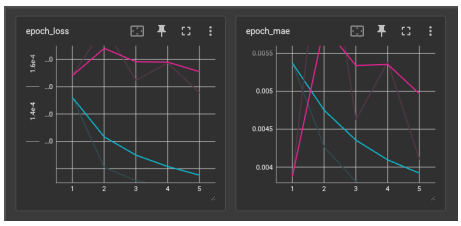

Figura 15. Gráfico de treinamento sem pesos. Elaboração própria.

O modelo apresentou queda consistente na perda e no erro absoluto médio nos dados de treinamento, indicando

aprendizado. No entanto, a validação mostrou um aumento nas métricas a partir da 2ª época, seguido de oscilações, sugerindo o início de overfitting. Isso indica que o modelo demonstrou sinais de sobreajuste (overfitting) aos dados de treino, prejudicando o desempenho em dados não vistos. O overfitting ocorre especialmente em modelos com muitos neurônios, muitas camadas ou muitas épocas, situações em que a rede deixa de aprender os padrões gerais e passa a memorizar os dados específicos do treinamento.

Ambas mostraram tendência de queda ao longo das épocas, indicando que o modelo estava aprendendo. Houve uma leve oscilação na MAE de validação nas primeiras épocas, o que é comum em séries temporais, mas a métrica se estabilizou depois, sem indícios de overfitting até a quinta época.

O gráfico da Figura 16 demonstra as curvas de aprendizados para o treinamento com pesos (linha roxa) e para a validação (linha verde).

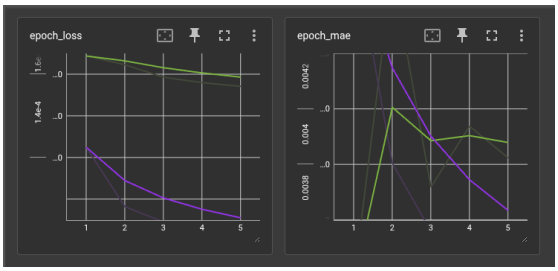

Figura 16. Gráfico de treinamento com pesos. Elaboração própria.

O modelo apresentou queda consistente na perda e no MAE nos dados de treinamento (linhas azuis), indicando aprendizado. No entanto, a validação (linhas rosa) mostrou um aumento nas métricas a partir da 2ª época, seguido de oscilações, sugerindo o início de *overfitting*. Isso indica que o modelo passou a se ajustar demais aos dados de treino, prejudicando o desempenho em dados não vistos.

B. Análise da predição

A seguir, os gráficos da Figura 17 e Figura 18 exibem o comparativos entre as curvas das predições absolutas e as reais para cada uma das *steps* analisados na predição.

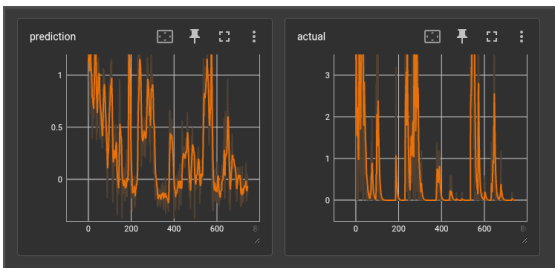

Figura 17. Gráfico comparativo entre as predições e valores reais (sem pesos). Elaboração própria.

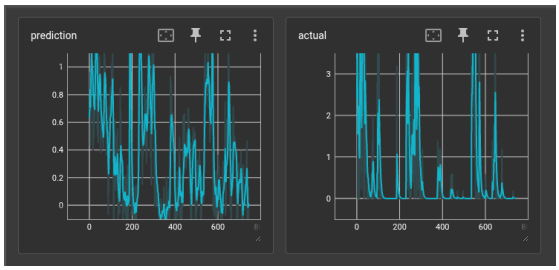

Figura 18. Gráfico comparativo entre as predições e valores reais (com pesos). Elaboração própria.

A predição nos dois cenários apresentou comportamentos semelhantes, especialmente em relação às tendências de alta e baixa. No entanto, os valores absolutos se mostraram bastante dispersos em comparação com os dados reais. Ainda assim, é possível observar que as tendências foram bem capturadas, indicando que, apesar da imprecisão nos valores absolutos, o modelo conseguiu aprender os padrões de variação ao longo do tempo.

C. Análise de histogramas - comportamento diário

As Figuras 19 e 20 permitem a análise do comportamento diário da predição, representando a quantidade de precipitação estimada para cada dia do período avaliado.

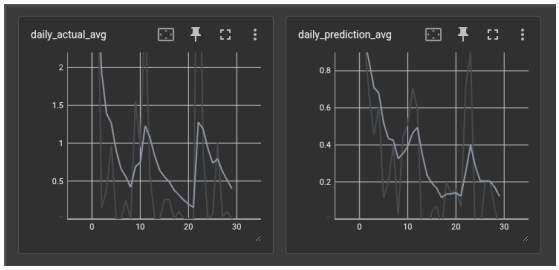

Figura 19. Histograma comparativo do comportamento diário entre a predição e valores reais (sem pesos). Elaboração própria.

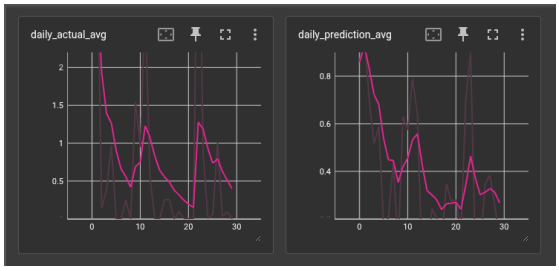

Figura 20. Histograma comparativo do comportamento diário entre a predição e valores reais (com pesos). Elaboração própria.

As predições representadas por ambos os histogramas são bastante semelhantes entre si, especialmente no que diz respeito às tendências de aumento e diminuição. No entanto, os valores absolutos apresentaram discrepâncias

significativas. Ainda assim, observa-se que o histograma da predição com pesos reproduz de forma ligeiramente mais fiel as tendências de variação em comparação ao histograma sem pesos.

VI. Conclusões

Este trabalho teve como objetivo explorar o desenvolvimento e a compreensão do uso de Redes Neurais Artificiais para a previsão de séries temporais, com foco na estimativa de precipitação pluviométrica. Foram discutidas tecnologias, metodologias e técnicas voltadas à construção de um modelo de Rede Neural, utilizando dados pluviométricos de diversos anos para seu treinamento.

Apesar da utilização de dados de vários anos, as predições geradas pelo modelo apresentaram dispersão em relação aos valores absolutos reais. Isso pode estar relacionado à qualidade e/ou à quantidade das variáveis utilizadas, que talvez não sejam adequadas para o modelo proposto. Além disso, o intervalo de tempo considerado na série temporal pode não ter sido o mais apropriado para proporcionar resultados mais precisos.

Por outro lado, observa-se que a Rede Neural Artificial foi capaz de aprender os padrões de variação ao longo do tempo, reproduzindo com razoável fidelidade as tendências de subida e descida dos dados. Esse comportamento indica que, embora os valores absolutos tenham sido imprecisos, o modelo foi capaz de captar a dinâmica temporal dos dados, indicando potencial para aplicações futuras com ajustes metodológicos.

Referências

[1] NAE – Núcleo de Assuntos Estratégicos da Presidência da República. *Mudança do Clima, Volume I. Negociações Internacionais sobre a mudança do clima: vulnerabilidade, impactos e adaptação à mudança do clima*. NAE-SECOM, 2005.

[2] Allisson David de Oliveira Martins et al. "Impactos econômicos no Rio Grande do Sul devido as enchentes: estimativas iniciais e repercussões para o Nordeste". Em: *Informe ETENE* (2024).

[3] Juliano Santos Finck. "Previsão em tempo atual de níveis fluviais com Redes Neurais Artificiais: Aplicação à bacia do rio Taquari-Antas/RS". Em: *Universidade Federal do Rio Grande do Sul* (2020).

[4] Juliana Aparecida Anochi. "Previsão climática de precipitação por Redes Neurais autoconfiguradas". Em: (2015). URL: http://mtc-m21b.sid.inpe.br/col/sid.inpe.br/mtc-m21b/2015/09.16.22.02/doc/publicacao.pdf.

[5] Marcelo S. Portugal e Luiz Gustavo L. Fernandes. "Redes neurais artificiais e previsão de séries econômicas: uma introdução". Em: *Nova Economia* 6.1 (dez. de 2013). URL: https://revistas.face.ufmg.br/index.php/novaeconomia/article/view/2269.

[6] Debora Barbosa Aires, Ronaldo César Dametto e Antonio Fernando Crepaldi. "Previsão de séries temporais financeiras utilizando redes neurais artificiais: um comparativo na crise de 2008". Em: *GEPROS. Gestão da Produção, Operações e Sistemas* (2018).

[7] Irem Islek e Sule Gunduz Oguducu. "Use of LSTM for Short-Term and Long-Term Travel Time Prediction". Em: *CEUR Workshop Proceedings* (2018).

[8] Martín Abadi et al. "TensorFlow: Um Sistema para Aprendizado de Máquina em Larga Escala". Em: *Associação USENIX* (2016).

[9] Stefan Tilkov e Steve Vinoski. "Node.js: Using JavaScript to Build High-Performance Network Programs". Em: *ComputingNow.computer.org.* (2010).

[10] Tiago Pagotto et al. "Scrum solo: Software process for individual development". Em: *2016 11th Iberian Conference on Information Systems and Technologies (CISTI)* (2016). DOI: 10.1109/CISTI.2016.7521555.

[11] Joanelise da Costa. *Análise da implementação do sistema Kanban no processo de fabricação de rolamentos automotivos*. Rel. técn. Universidade Federal do Paraná, 2019. URL: https://acervodigital.ufpr.br/xmlui/bitstream/handle/1884/70362/R%20-%20E%20-%20JOANELISE%20DA%20COSTA.pdf?sequence=1&isAllowed=y.

[12] F. G. Silva, S. C. P. Hoentsch e L. Silva. "Uma análise das Metodologias Ágeis FDD e Scrum sob a Perspectiva do Modelo de Qualidade MPS.BR". Em: *SCIENTIA PLENA* (2019).

[13] Devmedia. "Métodos Ágeis – Parte 02". Em: *Devmedia* (2008). Disponível em https://www.devmedia.com.br/metodos-ageis-parte-02/9443.

[14] Jordana Wilm Doninelli, Jose Mario Vicensi Grzybowski e Roberto Valmir da Silva. "Previsão pluviométrica por meio da aplicação de Redes Neurais Artificiais Recorrentes alimentadas com dados meteorológicos em tempo atual". Em: *Jornada de Iniciação Científica e Tecnológica da UFFS* (2020).

[15] Rafaela dos Santos Sousa et al. "Redes Neurais Artificiais Aplicadas à Previsão Antecipada de Precipitações na Região Central de Manaus". Em: *Escola Superior de Tecnologia Universidade do Estado do Amazonas* (2017).

[16] Daniel Dantas et al. "Uso de Redes Neurais Artificiais na previsão da precipitação de períodos chuvosos". Em: *Revista Espinhaço* (2016).

[17] Cristiano Foggetti. *Gestão ágil de projetos*. São Paulo: Pearson, 2015.

[18] Ingo Reichert Junior. "Normalização x Padronização: Qual a diferença?" Em: *Medium* (2023). Disponível em https://medium.com/@ingoreichertjr/normalização-x-padronização-qual-a-diferença-fa14352df501.