UNIVERSIDADE FEDERAL DO PARANÁ

GUILHERME PRATES LEANDRO DA SILVA

LOCALIZAÇÃO DE FALTAS EM SISTEMAS ELÉTRICOS COM MÚLTIPLAS ÁREAS USANDO APRENDIZADO DE MÁQUINA

CURITIBA

2024

GUILHERME PRATES LEANDRO DA SILVA

LOCALIZAÇÃO DE FALTAS EM SISTEMAS ELÉTRICOS COM MÚLTIPLAS ÁREAS USANDO APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso apresentado à disciplina TE348 – Trabalho de Conclusão de Curso II do curso de Graduação em Engenharia Elétrica com Ênfase em Sistemas Embarcados da Universidade Federal do Paraná, como requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Ricardo Schumacher

CURITIBA

2024

TERMO DE APROVAÇÃO

GUILHERME PRATES LEANDRO DA SILVA

LOCALIZAÇÃO DE FALTAS EM SISTEMAS ELÉTRICOS COM MÚLTIPLAS ÁREAS USANDO APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso apresentado à disciplina TE348 – Trabalho de Conclusão de Curso II do curso de Graduação em Engenharia Elétrica com Ênfase em Sistemas Embarcados da Universidade Federal do Paraná, como requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica, pela seguinte banca examinadora:

Prof. Dr. Ricardo Schumacher
Orientador

Prof. Dr. Gideon Villar Leandro
UFPR

Profa. Dra. Elizete Maria Lourenço
UFPR

Curitiba, 11 de Dezembro de 2024.

Com sentimento de dever cumprido e satisfação, dedico este trabalho de conclusão de curso ao jovem Guilherme que em sua infância desmontava carrinhos de controle remoto e se fascinava tentando entender seu funcionamento.

AGRADECIMENTOS

Ao concluir esta importante etapa da minha jornada acadêmica, gostaria de expressar minha profunda gratidão àqueles que estiveram ao meu lado em cada momento.

À minha família, especialmente à minha mãe, Elizete Prates da Silva, e ao meu pai, Nelson Leandro da Silva, sou eternamente grato pelo amor incondicional, pelo apoio constante e por sempre acreditarem nos meus sonhos. Vocês me ensinaram a ser resiliente, a perseverar diante das adversidades e a valorizar cada conquista, por menor que fosse. Sem vocês, nada disso seria possível.

Ao Instituto Federal do Paraná (IFPR), meu mais sincero agradecimento por ter sido a base do meu ensino técnico em eletrônica. Foi lá que meus horizontes se expandiram, e adquiri o conhecimento teórico e prático que pavimentou o caminho para o que sou hoje.

Ao meu orientador, professor Dr. Ricardo Schumacher, meu reconhecimento especial. Sua sabedoria, idoneidade e dedicação ao ensino foram fundamentais para meu crescimento profissional e pessoal. Agradeço por ter me guiado com paciência e por me incentivar a buscar sempre a excelência.

Aos meus colegas de curso, agradeço por cada momento compartilhado. Juntos, enfrentamos desafios, comemoramos vitórias e criamos laços que transcendem as paredes da universidade. Foi ao lado de vocês que aprendi o verdadeiro significado de cooperação e amizade.

Por fim, a todas as pessoas que, de alguma forma, fizeram parte dessa caminhada, deixo meu muito obrigado. Cada contribuição, seja grande ou pequena, foi essencial para que eu chegasse até aqui.

Este trabalho é um reflexo da soma de todos esses esforços e apoio, e agradeço a vocês por esta conquista.

“A eletricidade é o sangue da civilização moderna,
e os sistemas de potência são suas artérias.“
Willis R. Whitney - Tradução do autor

RESUMO

Este trabalho aborda um dos principais desafios enfrentados pelos sistemas elétricos modernos: a detecção e localização precisa de falhas. Com o aumento da complexidade das redes elétricas e a crescente demanda por confiabilidade e eficiência, torna-se essencial desenvolver métodos avançados para monitoramento e diagnóstico. A aplicação de aprendizado de máquina nesse contexto surge como uma abordagem promissora, capaz de identificar padrões em grandes volumes de dados, reduzindo o tempo de resposta e minimizando os impactos das falhas. Essa tecnologia não apenas melhora a gestão e manutenção das redes elétricas, mas também contribui para a segurança energética e a sustentabilidade do setor. Este trabalho propõe desenvolver um modelo de aprendizado de máquina para detectar e localizar falhas em um sistema elétrico. Utilizando dados simulados, o modelo será treinado para identificar padrões de falhas e determinar a área de ocorrência do evento. A aplicação de aprendizado de máquina visa melhorar a confiabilidade da rede elétrica, representando uma contribuição significativa para a inovação tecnológica e o desenvolvimento sustentável no setor de energia. O sistema elétrico analisado é o modelo de 39 barras do IEEE, extraído de bibliografia referenciada e modelado no software Anatem. Após a validação do modelo, foram realizadas diversas simulações de falhas para a geração de dados, abrangendo eventos como abertura de linha, curto-circuito na barra, afundamento de tensão, curto-circuito na linha e desligamento de barra. Esses eventos foram armazenados no Matlab como amostras temporais de variáveis físicas, como tensão, corrente e frequência, simulando dados coletados por equipamentos reais de monitoramento. Os dados foram usados para treinar quatro modelos de aprendizado de máquina, destinados a identificar a área de ocorrência das falhas. Os dois primeiros modelos, um de regressão logística e outro baseado em rede neural, foram aplicados à detecção de falhas em um sistema dividido em duas áreas. O terceiro modelo, também uma rede neural, realizou a detecção em quatro áreas, enquanto o quarto modelo aprimorou o desempenho deste último. Os resultados mostraram que o primeiro modelo apresentou precisão de cerca de 60%, enquanto o segundo, com base em redes neurais, alcançou 99,57% de acurácia para dados de teste. O terceiro modelo obteve 55,05% de precisão para validação, e o quarto modelo, 68,84%, destacando o potencial de melhorias. Esses resultados indicam a viabilidade de aplicar aprendizado de máquina para detecção e localização de falhas, com avanços significativos em cenários mais complexos. O trabalho evidencia o potencial para desenvolver modelos ainda mais robustos, contribuindo para redes elétricas mais confiáveis e resilientes.

Palavras-chaves: Aprendizado de máquina; Sistemas elétricos de potência; Falhas no sistema elétrico; Redes neurais.

ABSTRACT

This work addresses one of the main challenges faced by modern power systems: the accurate detection and localization of faults. With the increasing complexity of power grids and the growing demand for reliability and efficiency, it becomes essential to develop advanced methods for monitoring and diagnostics. The application of machine learning in this context emerges as a promising approach, capable of identifying patterns in large volumes of data, reducing response times, and minimizing the impacts of faults. This technology not only improves the management and maintenance of power grids but also contributes to energy security and the sustainability of the sector. This work proposes the development of a machine learning model to detect and locate faults in a power system. Using simulated data, the model will be trained to identify fault patterns and determine the area of occurrence of the event. The application of machine learning aims to improve the reliability of the power grid, representing a significant contribution to technological innovation and sustainable development in the energy sector. The analyzed power system is the IEEE 39-bus model, extracted from referenced literature and modeled in the Anatem software. After model validation, various fault simulations were conducted to generate data, covering events such as line outages, bus short circuits, voltage sags, line short circuits, and bus disconnections. These events were stored in Matlab as temporal samples of physical variables, such as voltage, current, and frequency, simulating data collected by real monitoring equipment. The data were used to train four machine learning models designed to identify the fault occurrence area. The first two models, one based on logistic regression and the other on a neural network, were applied to fault detection in a system divided into two areas. The third model, also a neural network, performed detection in four areas, while the fourth model improved the performance of the latter. The results showed that the first model achieved an accuracy of about 60%, while the second, based on neural networks, reached 99.57% accuracy for test data. The third model achieved 55.05% accuracy for validation, and the fourth model achieved 68.84%, highlighting the potential for improvement. These results indicate the feasibility of applying machine learning for fault detection and localization, with significant advancements in more complex scenarios. The work demonstrates the potential to develop even more robust models, contributing to more reliable and resilient power grids.

Key-words: Machine learning; Power systems; Electrical system faults; Neural networks.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E DE SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
AC	Corrente Alternada
AM	Aprendizado de Máquina
AMI	Advanced Metering Infrastructure
ANEEL	Agência Nacional de Energia Elétrica
API	Interface de Programação de Aplicações
CEPEL	Centro de Pesquisas de Energia Elétrica
CNN	Rede Neural Convolucional
F	Frequência Elétrica
Flxa	Fluxo de Potência Ativa
Flxr	Fluxo de Potência Reativa
GAN	Rede Adversária Generativa
I	Corrente Elétrica
IA	Inteligência Artificial
IBGE	Instituto Brasileiro de Geografia e Estatística
IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos
MAE	Erro Absoluto Médio
ML	Machine Learning
MLP	Rede Neural Perceptron Multicamada
MSE	Erro Quadrático Médio
MVar	Megavolt-ampere reativo
MW	Megawatt
ONU	Organização das Nações Unidas
P	Potência Ativa

PCA	Principal Component Analysis
PMU	Unidade de Medição Fasorial
Pcar	Potência Elétrica Ativa Consumida
Pger	Potência Elétrica Ativa Gerada
Q	Potência Reativa
Qcar	Potência Elétrica Reativa Consumida
Qger	Potência Elétrica Reativa Gerada
R	Resistência
R^2	Coeficiente de Determinação
RNN	Rede Neural Recorrente
SEP	Sistemas Elétricos de Potência
SVM	Máquinas de Vetores de Suporte
V	Tensão Elétrica
X	Reatância
Y	Admitância
k-NN	K-Nearest Neighbors

*

LISTA DE SÍMBOLOS

θ	ângulo
λ	termo de regularização

*

SUMÁRIO

1 INTRODUÇÃO

1.1 CONTEXTO E PROBLEMA

A primeira aplicação prática da eletricidade foi na iluminação pública em grandes centros comerciais e residenciais de alto padrão, com a invenção da lâmpada incandescente por Thomas Edison no final do século XIX. Isso inaugurou uma era de iluminação artificial que transformou a vida noturna e aumentou a produtividade humana. Simultaneamente, para a realização desses sistemas de iluminação, foram desenvolvidas centrais elétricas localizadas em áreas mais afastadas de onde necessariamente estavam instaladas as lâmpadas, iniciando também, o desenvolvimento de sistemas de geração e distribuição de eletricidade.

Desde então, a eletricidade tornou-se a espinha dorsal da sociedade moderna, se expandindo globalmente, proporcionando acesso à energia elétrica tanto em áreas urbanas quanto rurais e contribuindo para o avanço tecnológico da humanidade. Nos dias de hoje, é praticamente impossível imaginar a vida sem ela. Os sistemas elétricos de potência desempenham um papel crucial na nossa infraestrutura, fornecendo energia para uma vasta gama de aplicações, desde o funcionamento de eletrodomésticos até o suporte a indústrias de alta tecnologia. Além disso, a eletricidade impulsiona a inovação em áreas como transporte elétrico, armazenamento de energia e energias renováveis, desempenhando um papel fundamental na transição para uma economia mais sustentável.

De acordo com dados da Organização das Nações Unidas (ONU) e do Instituto Brasileiro de Geografia e Estatística (IBGE), atualmente cerca de 91% da população mundial tem acesso a eletricidade e no Brasil, cerca de 99,8% da população brasileira dispõe desse serviço (IEA *et al.*, 2024) e (IBGE, 2022), com expectativa de que esses números aumentem cada vez mais, com os sistemas ficando cada vez mais densos e interligados. Portanto, com esse panorama estabelecido, compreender os sistemas elétricos de potência é essencial para garantir a confiabilidade, eficiência e segurança do fornecimento de energia elétrica em escala global. Uma situação ocorrida no território brasileiro no ano de 2023 mostrou as fragilidades que os sistemas elétricos podem exibir. Uma abertura da interligação entre o sistema elétrico Norte e o sistema do Sudeste causou um apagão definido pela Agência Nacional de Energia Elétrica (ANEEL) como um "evento de grande porte", afetando mais de 29 milhões de pessoas e demorando mais de 15 horas para o total restabelecimento da energia em alguns estados.

Nos últimos anos, o setor de sistemas elétricos de potência tem passado por

grandes transformações tecnológicas, impulsionadas pela crescente demanda por confiabilidade, eficiência e sustentabilidade. O advento de dispositivos de sensoriamento inteligente, como os medidores avançados de energia, ou *Advanced Metering Infrastructure* (AMI) e dispositivos de monitoramento em tempo real, revolucionou a forma como os sistemas elétricos são gerenciados e operados. Esses dispositivos fornecem uma quantidade massiva de dados em tempo real, que podem ser utilizados para monitorar o comportamento da rede, detectar anomalias e responder rapidamente a falhas. Essa modernização permite que os operadores de sistemas de energia obtenham uma visão mais detalhada e precisa do estado da rede elétrica, tornando o sistema mais resiliente e eficiente.

A localização de falhas é um dos aspectos críticos em sistemas de energia, pois envolve a identificação rápida e precisa do local onde um problema ocorreu na rede elétrica. Essa tarefa é vital para reduzir o tempo de interrupção, minimizar os custos operacionais e garantir a confiabilidade do fornecimento de energia. Tradicionalmente, a localização de falhas era realizada com base em métodos convencionais, como a análise de circuitos e técnicas de proteção, que muitas vezes eram imprecisas ou demoradas. No entanto, com o surgimento de grandes volumes de dados oriundos de sistemas de medição e monitoramento avançados, novas oportunidades surgiram para melhorar a precisão e a eficiência desse processo.

Nesse contexto, o aprendizado de máquina (AM) tem se mostrado uma ferramenta promissora para a análise e interpretação de dados complexos em sistemas de energia. Técnicas avançadas, como redes neurais, Máquinas de Vetores de Suporte (SVM), e algoritmos de aprendizado profundo, têm sido aplicadas para detectar e localizar falhas de maneira automatizada e com maior precisão. Essas técnicas utilizam dados históricos e em tempo real para identificar padrões e correlações entre eventos, o que permite uma resposta mais rápida e precisa às falhas na rede elétrica. Trabalhos relevantes na área, como os de (Bunnoon, 2013), (Jamil *et al.*, 2015) e (Vaish *et al.*, 2021), demonstraram que o uso de modelos de aprendizado de máquina pode reduzir significativamente o tempo de detecção e melhorar a acurácia na localização de falhas em redes de distribuição e transmissão.

Portanto, a combinação de tecnologias de sensoriamento inteligente e técnicas de aprendizado de máquina está redefinindo a forma como as falhas são tratadas em sistemas elétricos. Essa convergência oferece uma nova perspectiva sobre a operação das redes elétricas, onde a análise de dados em larga escala permite uma resposta mais eficiente e precisa, trazendo benefícios tanto para as empresas de energia quanto para os consumidores finais.

1.2 OBJETIVOS

1.2.1 OBJETIVO GERAL

O objetivo geral deste trabalho é desenvolver um modelo de aprendizado de máquina, especificamente utilizando redes neurais artificiais, capaz de analisar uma base de dados com características físicas de um sistema elétrico quando este estiver sob a influência de um evento danoso. O modelo será projetado para detectar e localizar falhas dividindo o sistema elétrico em diversas áreas, permitindo uma análise mais segmentada e precisa. A partir dos dados coletados por dispositivos de sensoriamento, a rede neural artificial será treinada para identificar padrões associados a falhas e determinar com a maior precisão possível a área do sistema de origem do evento, auxiliando na mitigação de impactos e na restauração mais rápida da rede.

1.2.2 OBJETIVOS ESPECÍFICOS

- Conceituar eventos que ocorrem em sistemas elétricos de potência;
- Conceituar modelos de aprendizado de máquina e suas principais aplicações;
- Analisar o funcionamento das ferramentas mais utilizadas para o desenvolvimento de modelos de aprendizado;
- Localizar faltas em um sistema dividido em duas áreas;
- Localizar faltas em um sistema dividido em quatro áreas;

1.3 JUSTIFICATIVA

A aplicação de modelos de aprendizado de máquina para a identificação de falhas na rede elétrica é um tema em evidência e altamente relevante para aprimorar a confiabilidade e eficiência dos sistemas elétricos de potência. No contexto atual, a área de aprendizado de máquina está em ascensão, impulsionada pela disponibilidade de grandes volumes de dados e avanços significativos em algoritmos e técnicas de processamento de informações. Nesse sentido, este projeto se destaca como uma abordagem para aproveitar o potencial dessas tecnologias emergentes e aplicá-las no contexto da detecção de faltas em sistemas elétricos de potência.

A implementação de um modelo de aprendizado de máquina para identificação de falhas na rede elétrica oferece uma série de benefícios significativos, como visto em trabalhos como (Hasan *et al.*, 2017). Em primeiro lugar, a capacidade de prever e diagnosticar falhas de forma proativa pode ajudar a evitar interrupções no fornecimento de energia, minimizando o impacto negativo sobre os consumidores e a economia

como um todo. Além disso, ao identificar padrões e tendências nos dados operacionais, os modelos de aprendizado de máquina podem fornecer informações valiosas para otimizar a manutenção preventiva e o planejamento de investimentos em infraestrutura elétrica.

Além dos benefícios imediatos para a operação e manutenção dos sistemas elétricos, a pesquisa nesse campo tem o potencial de gerar uma série de soluções inovadoras e impactantes. A aplicação de técnicas avançadas de aprendizado de máquina pode abrir novas perspectivas para a automação e controle inteligente da rede elétrica, permitindo uma resposta mais ágil e eficaz às demandas dinâmicas do mercado de energia. Além disso, a integração de modelos preditivos com fontes de energia renovável e sistemas de armazenamento de energia pode impulsionar ainda mais a transição para uma matriz energética mais sustentável e resiliente.

Portanto, este projeto não apenas aborda uma necessidade premente na área de sistemas elétricos de potência, mas também representa uma oportunidade para contribuir para o progresso humano, promovendo a inovação tecnológica, a sustentabilidade ambiental e o desenvolvimento econômico. Ao explorar e desenvolver soluções baseadas em aprendizado de máquina, podemos moldar um futuro energético mais eficiente, confiável e sustentável para as gerações futuras.

1.4 ORGANIZAÇÃO DO TRABALHO

O conteúdo deste trabalho está dividido em cinco capítulos, cada um dos quais aborda diferentes aspectos do tema central. A estrutura do trabalho foi planejada para guiar o leitor de forma lógica e coerente ao longo do desenvolvimento da pesquisa. No primeiro capítulo, apresentamos o tema do trabalho, seus objetivos, a justificativa da pesquisa e a metodologia utilizada. Esta seção também inclui a formulação do problema de pesquisa e as hipóteses que norteiam o estudo. No segundo capítulo, realizamos uma revisão da literatura existente sobre o tema, abordando as principais teorias, conceitos e estudos prévios. Esta revisão visa fornecer o embasamento teórico necessário para a compreensão do tema e situar a pesquisa no contexto acadêmico atual. A metodologia empregada na pesquisa é detalhada no terceiro capítulo. Nele, descrevemos os métodos de coleta e análise de dados, a população e amostra estudadas, bem como as técnicas e instrumentos utilizados na pesquisa. Esta seção é fundamental para garantir a replicabilidade e a validade do estudo. No quarto capítulo, apresentamos os resultados obtidos com a pesquisa e realizamos a análise e discussão desses dados à luz da teoria revisada no segundo capítulo. Os resultados são interpretados de forma a responder às questões de pesquisa e testar as hipóteses propostas. No quinto capítulo, sintetizamos as principais conclusões do trabalho, discutimos as implicações dos resultados e sugerimos possíveis direções para pesquisas

futuras. Esta seção também aborda as limitações do estudo e suas contribuições para a área de conhecimento. Além disso, o capítulo se dedica a tentar encontrar aplicações práticas dos resultados adquiridos no estudo realizado, de forma a contribuir com a sociedade e o desenvolvimento humano. A lista de referências inclui todas as fontes citadas ao longo do trabalho, de acordo com as normas da Associação Brasileira de Normas Técnicas (ABNT). Informações complementares encontram-se disponíveis como apêndices.

2 FUNDAMENTAÇÃO TEÓRICA

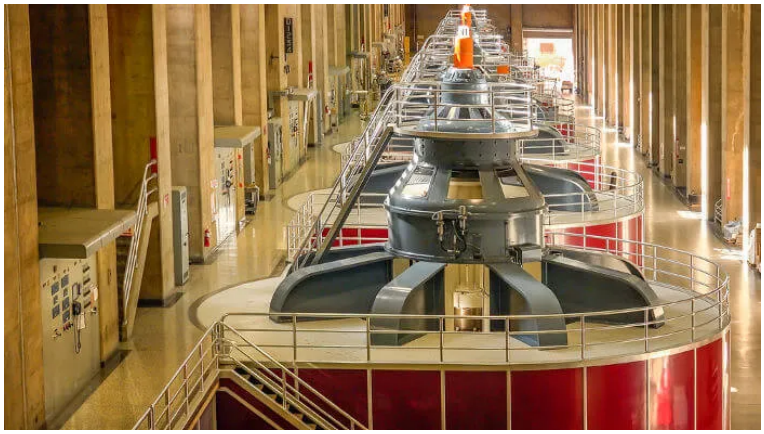
2.1 SISTEMAS ELÉTRICOS DE POTÊNCIA

Os Sistemas Elétricos de Potência (SEP) são redes complexas de infraestrutura elétrica projetadas para gerar, transmitir e distribuir energia elétrica de forma eficiente e confiável e, segundo (Monticelli; Garcia, 2011), para compreender melhor o funcionamento e a importância dos sistemas elétricos de potência, é essencial definir e analisar seus componentes. O funcionamento geral desses sistemas envolve três grandes pilares: geração, transmissão e distribuição, que agem, de forma resumida, na conversão de várias formas de energia em eletricidade, seu transporte através de linhas de transmissão de alta tensão e sua distribuição aos consumidores finais por meio de redes de distribuição de baixa tensão.

Em princípio, o processo de geração de energia elétrica envolve a conversão de diferentes formas de energia em eletricidade. As usinas geradoras, que podem ser termelétricas, hidrelétricas, nucleares, eólicas ou solares, são responsáveis por esse processo. O tipo mais comum de gerador de energia elétrica é um dispositivo que converte energia mecânica em energia elétrica por meio do princípio da indução eletromagnética, descoberto por Michael Faraday no século XIX. Ele desempenha um papel fundamental na geração de eletricidade em diversas aplicações, desde grandes usinas geradoras até pequenos geradores portáteis.

Os geradores atuais baseados em tecnologias já consolidadas e amplamente utilizadas desde sua invenção em 1866 por Werner von Siemens, por questões físicas e de isolamento elétrico, tem como padrão a operação com tensões elétricas na faixa de 10 V a 30 kV (Boylestad, 2012). Desse modo, os geradores que estão afastados dos centros de carga injetam sua potência gerada na rede através de transformadores elevadores que têm por finalidade transformar a potência gerada dos níveis de tensão de geração para os níveis de tensão de transmissão, que são comumente os níveis de 345 kV, 500 kV ou 750 kV. A FIGURA 1 mostra uma linha de turbinas geradoras de grande porte, utilizadas em usinas hidrelétricas.

FIGURA 1 – GERADORES

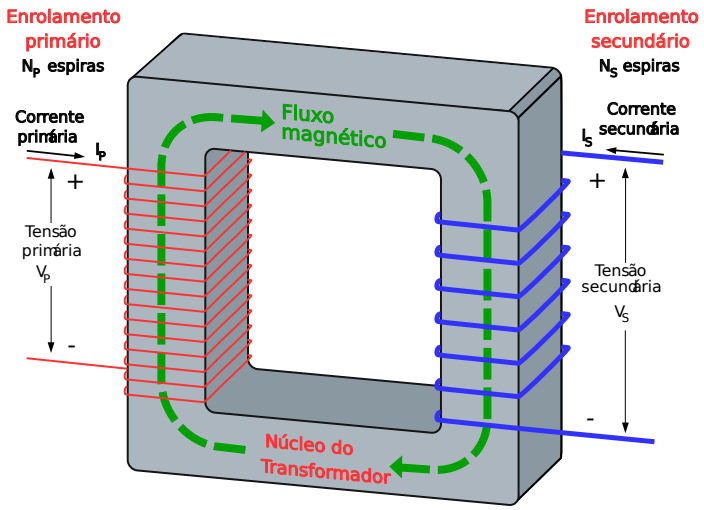


FONTE: Brasil Escola (2023).

Entrando no mérito da transmissão de energia elétrica vemos a figura do transformador como um importante dispositivo para a correta execução desse processo. Um transformador elétrico é um dispositivo que é usado para aumentar ou diminuir a tensão de uma corrente alternada (AC) em um sistema elétrico, mantendo a potência constante na entrada e saída do transformador. Ele é essencial no sistema de transmissão e distribuição de energia elétrica, permitindo que a eletricidade seja transportada de forma eficiente em diferentes níveis de tensão.

O transformador é composto por dois enrolamentos de fio condutor (bobinas) envolvidos em torno de um núcleo de material ferromagnético (como ferro laminado). Os dois enrolamentos são conhecidos como o enrolamento primário e o enrolamento secundário. A FIGURA 2 mostra um diagrama de um transformador ideal. Nela também podemos ver a relação entre o número de espiras no enrolamento primário e no enrolamento secundário. Essa relação é a responsável em alterar os níveis de tensão elétrica entre o primário e o secundário na transmissão, pois a tensão induzida nos enrolamentos é proporcional ao números de espiras dos mesmos.

FIGURA 2 – TRANSFORMADOR DE TENSÃO IDEAL



FONTE: Wikipedia (2007).

Na ponta final de um sistema elétrico de potência, se encontra a figura do sistema de distribuição que é responsável por entregar eletricidade da rede de transmissão de alta tensão (onde a eletricidade é transmitida por longas distâncias) aos consumidores finais, como residências, empresas e indústrias. O sistema de distribuição é uma parte crucial da infraestrutura elétrica, garantindo que a eletricidade seja fornecida de forma confiável, segura e eficiente para os usuários finais. Os principais componentes do sistema de distribuição incluem as subestações de distribuição, que recebem eletricidade da rede de transmissão em alta tensão e a reduzem para níveis adequados de distribuição, geralmente entre 7,2 kV e 33 kV. As subestações também desempenham funções como o controle da tensão e o chaveamento entre diferentes linhas de distribuição.

Além disso, as redes de distribuição são compostas por cabos condutores (em média tensão) que transportam eletricidade das subestações para áreas urbanas, industriais e rurais. Essas redes podem ser aéreas (utilizando postes e fios suspensos) ou subterrâneas (enterradas sob o solo). Os transformadores de distribuição são essenciais nas subestações e ao longo das redes de distribuição, ajustando a tensão para níveis adequados de uso pelos consumidores, elevando ou diminuindo conforme necessário para atender às demandas dos clientes.

Outros componentes incluem dispositivos de proteção e controle, como relés de proteção, disjuntores e chaves seccionadoras, que protegem a rede contra falhas elétricas e permitem o monitoramento e o controle remoto das operações do sistema. Os medidores de energia são instalados em residências, empresas e outras instalações para medir o consumo de eletricidade, sendo essenciais para o faturamento e o monitoramento do consumo.

Adicionalmente, o sistema de distribuição possui redes secundárias e ramais menores que se conectam diretamente aos clientes, operando em baixa tensão (127 V ou 220 V) e fornecendo energia diretamente aos consumidores. Equipamentos de manobra e proteção, como disjuntores e fusíveis, são instalados nas instalações dos consumidores para garantir a segurança e proteção contra sobrecargas e curtos-circuitos. A FIGURA 3 mostra um poste com transformador, que representa alguns dos componentes do sistema de distribuição que mais influenciam na vida das pessoas.

FIGURA 3 – POSTE ELÉTRICO DA DISTRIBUIÇÃO

FONTE: G1 (2007).

Em suma, um sistema elétrico de potência é uma infraestrutura complexa e interligada projetada para gerar, transmitir, distribuir e controlar a energia elétrica de forma eficiente e confiável. Esse sistema é composto por diversas partes integradas que trabalham em conjunto para garantir o fornecimento contínuo de eletricidade aos consumidores finais.

Após a geração, a eletricidade é transportada por meio de linhas de transmissão de alta tensão, que minimizam as perdas durante o transporte em longas distâncias. As subestações desempenham um papel crucial no sistema, convertendo e controlando a tensão da eletricidade entre os níveis de geração, transmissão e distribuição.

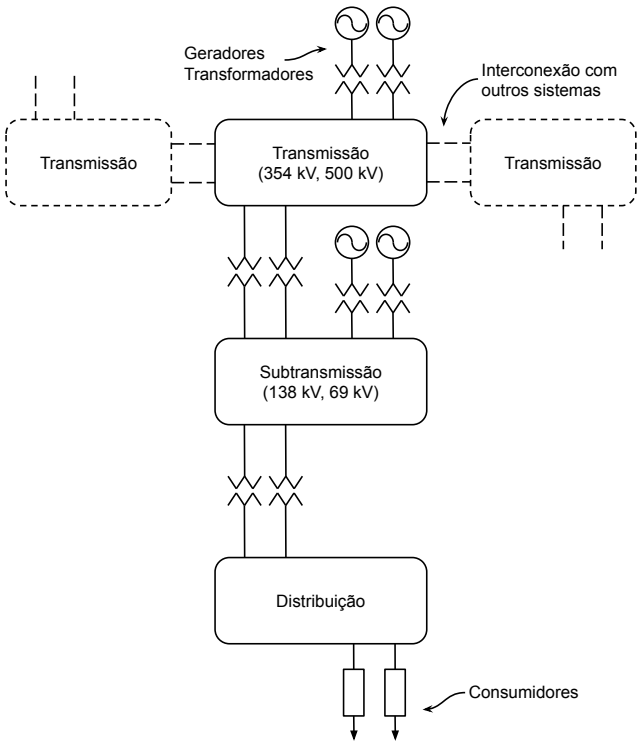
Na fase de distribuição, redes de distribuição entregam eletricidade aos consumidores finais por meio de cabos condutores operando em média e baixa tensão. Os transformadores são utilizados para ajustar a tensão conforme necessário, elevando-a para transmissão eficiente ou reduzindo-a para uso seguro em residências, indústrias e comércios.

O sistema elétrico também inclui equipamentos de controle e proteção, como relés, disjuntores e chaves seccionadoras, que garantem a segurança e a estabilidade do sistema elétrico. Além disso, a medição e o monitoramento do consumo de energia são realizados por meio de medidores instalados nos pontos de consumo,

permitindo o gerenciamento eficiente da rede. A FIGURA 4 mostra um diagrama de um sistema geração-transmissão-distribuição fictício, demonstrando a interconectividade dos processos necessária para seu pleno funcionamento.

Com a crescente integração de fontes de energia renováveis, como solar e eólica, o sistema elétrico de potência está evoluindo para incorporar tecnologias avançadas de controle, automação e monitoramento. Essas inovações visam melhorar a eficiência, a resiliência e a sustentabilidade dos sistemas elétricos, impulsionando o desenvolvimento econômico e social global e atendendo às demandas crescentes por energia de forma ambientalmente responsável.

FIGURA 4 – SISTEMA GERAÇÃO-TRANSMISSÃO-DISTRIBUIÇÃO GENÉRICO



FONTE: Adaptado de (Monticelli; Garcia, 2011).

2.2 FALTAS NA REDE ELÉTRICA

As faltas ou falhas na rede elétrica, que também são conhecidas como eventos da rede elétrica, consistem em conexões indesejadas entre dois ou mais condutores de um sistema elétrico. Existem diversas causas possíveis quando se trata dessas falhas, onde as que mais se destacam são elencadas pelo trabalho (Almobasher; Habiballah, 2020) como sendo as condições climáticas, o mau funcionamento de equipamentos e os erros humanos. As condições temporais/climáticas como causas de falhas na rede elétrica, têm por característica o fato de não poderem ser evitadas e tem efeitos mais

drásticos principalmente nas linhas de transmissão, podendo causar impedimentos nas operações.

As falhas por mau funcionamento de equipamentos ocorrem em decorrência do uso contínuo dos mesmos, fazendo com que a performance desses dispositivos diminua com o tempo e seus componentes apresentem deficiências, como a redução do isolamento elétrico. Essas condições podem levar a curtos circuitos e a um fluxo de corrente dissonante com o nominal que, quando ocorrendo em grandes equipamentos, pode ser sentido ao longo do sistema elétrico inteiro.

Em complemento, as falhas por erros humanos podem ser causadas por cálculos imprecisos que levam à seleção inadequada de dispositivos e equipamentos elétricos, como relés e disjuntores. Além disso, atrasar as manutenções programadas pode afetar o desempenho dos equipamentos, reduzindo sua eficiência e eventualmente causando mau funcionamento e falhas no sistema de energia.

Como explicado no trabalho *"An Overview of Transmission Line Protection by Artificial Neural Network: Fault Detection, Fault Classification, Fault Location, and Fault Direction Discrimination"* (Yadav; Dash, 2014), existem, primariamente, três tipos de faltas: as faltas simétricas, as faltas assimétricas e as faltas de circuito aberto. As falhas simétricas são aquelas em que as três fases de um sistema trifásico são afetadas igualmente, sendo consideradas falhas balanceadas. Em uma falha simétrica, a magnitude da corrente de curto-circuito é a mesma em todas as três fases. Normalmente, as falhas simétricas são mais severas, mas são menos comuns do que as falhas assimétricas, ficando em torno de 2 a 3% do total de falhas registradas em um sistema elétrico. Um exemplo clássico de falha simétrica é o curto-circuito trifásico, onde todas as três fases (A, B e C) entram em contato entre si ou com o neutro/terra simultaneamente.

Já as faltas assimétricas são aquelas em que as três fases de um sistema trifásico são afetadas de maneira desigual, sendo consideradas falhas não balanceadas. Em uma falha assimétrica, as correntes de curto-circuito nas fases não são iguais, resultando em um sistema desbalanceado. Essas falhas são mais comuns que as falhas simétricas, chegando a 80% de todas as falhas do sistema, e podem ser de vários tipos, dependendo das fases envolvidas. Exemplos incluem a falta fase-terra, onde uma única fase entra em contato com a terra, a falta fase-fase, onde duas fases entram em contato entre si, e a falta fase-fase-terra, onde duas fases entram em contato entre si e com a terra simultaneamente.

As falhas de circuito aberto ocorrem quando a continuidade do circuito elétrico é interrompida, impedindo o fluxo de corrente. Nessa situação, um ou mais condutores são desconectados ou rompidos, resultando em um circuito incompleto. As falhas de circuito aberto podem causar a interrupção do fornecimento de energia e a operação anormal de equipamentos. Elas podem ocorrer de diversas formas, como abertura em

uma fase, onde apenas uma fase está desconectada, abertura em duas fases, onde duas fases estão desconectadas, abertura em três fases, onde todas as três fases estão desconectadas, e abertura no neutro, onde o condutor neutro está desconectado, causando desequilíbrios de tensão em um sistema trifásico com carga desequilibrada.

Em resumo, as falhas simétricas afetam igualmente as três fases de um sistema trifásico, enquanto as falhas assimétricas afetam as fases de maneira desigual. Já as falhas de circuito aberto interrompem a continuidade do circuito, impedindo o fluxo de corrente. Compreender esses tipos de falhas é crucial para a análise, diagnóstico e mitigação de problemas em sistemas elétricos, ajudando a garantir a segurança, confiabilidade e eficiência do fornecimento de energia (Almobasher; Habiballah, 2020).

2.3 APRENDIZADO DE MÁQUINA

O aprendizado de máquina ou *machine learning* (ML) é um subcampo da inteligência artificial (IA) que se concentra no desenvolvimento de algoritmos e modelos capazes de aprender e fazer previsões ou tomar decisões baseadas em dados. Um dos estudiosos pioneiros na área do desenvolvimento de inteligências artificiais, Arthur Samuel, definiu o aprendizado de máquina como o campo de estudo que dá aos computadores a habilidade de aprender sem ter sido explicitamente programados, ou seja, diferentemente da programação tradicional, onde os desenvolvedores explicitamente codificam regras e instruções para resolver um problema específico, o aprendizado de máquina permite que os sistemas computacionais descubram padrões e informações a partir de grandes conjuntos de dados, ajustando automaticamente seus comportamentos com base nas experiências adquiridas.

Esse campo emergiu da confluência de estatística, ciência da computação e otimização, e vem revolucionando diversas áreas, como reconhecimento de voz e imagem, previsão de tendências de mercado, diagnósticos médicos, e muito mais. Atualmente algoritmos de aprendizado são utilizados em muitas aplicações que interagimos diariamente, tendo como exemplos claros os mecanismos de busca na web, como o Google, onde um dos motivos de que funciona tão bem é devido a um algoritmo de aprendizado que aprendeu a classificar as páginas da web toda vez que é utilizado para pesquisar na internet (Wang *et al.*, 2011).

Existem inúmeros algoritmos de AM que podem ser empregados na resolução de problemas, tanto que cientistas de dados gostam de enfatizar que não existe um algoritmo único que seja o melhor para resolver todos os problemas. O tipo de algoritmo empregado depende do tipo de problema que se deseja resolver, do número de variáveis, do tipo de modelo que se adequaria melhor a ele e outros diversos fatores. Desse modo, os programas de AM podem ser agrupados comumente em três grandes categorias, que são: aprendizado supervisionado; aprendizado não supervisionado e

aprendizado por reforço (Shinde; Shah, 2018).

O trabalho *"Machine Learning Algorithms - A Review"* (Mahesh, 2020) define que as aplicações de aprendizado supervisionado consistem em algoritmos que são treinados com um conjunto de dados rotulados, o que significa que cada entrada de treinamento vem com uma resposta ou rótulo desejado. Ou seja, a tarefa consiste em aprender uma função que mapeia uma entrada para uma saída com base em pares de entrada e saída de exemplo. Essa inferência da função ocorre a partir de dados de treinamento rotulados, que consistem em um conjunto de exemplos de treinamento. Esse tipo de algoritmo tem esse nome, justamente porque necessitam de assistência externa na forma da apresentação dos exemplos de entradas e saídas desejadas. O conjunto de dados de entrada é dividido em conjunto de treinamento e conjunto de teste onde os algoritmos aprendem algum tipo de padrão a partir do conjunto de dados de treinamento e os aplicam ao conjunto de dados de teste para previsão ou classificação.

As aplicações de aprendizado não supervisionado têm esse nome porque, ao contrário do aprendizado supervisionado, os dados recebidos não são rotulados, sendo deixados por conta própria do algoritmo para descobrir e apresentar uma estrutura peculiar nos dados que os relacionem. Quando novos dados são introduzidos, eles usam as características previamente aprendidas para reconhecer a classe dos dados. Esse tipo de aprendizado pode ser utilizado para descobrir novos padrões na base de dados ou como um uma parte de um processo de tratamento dos dados, e é principalmente utilizado para agrupamento (*clustering*) e redução de características.

Já o aprendizado por reforço é uma área de AM focada na interação do programa com um ambiente dinâmico. Nesse tipo de aprendizado, um agente interage com um ambiente, tomando ações e recebendo retorno em forma de recompensas ou penalidades, com o objetivo de maximizar a recompensa cumulativa ao longo do tempo. Esse paradigma envolve um processo iterativo no qual o agente aprende a associar estados do ambiente com ações que levam a recompensas mais altas, adaptando seu comportamento com base no retorno recebido. Amplamente utilizado em jogos, robótica e outras áreas, o aprendizado por reforço é especialmente adequado para problemas nos quais as ações do agente influenciam diretamente o ambiente e a melhor estratégia pode depender do contexto atual.

Dentro desses conceitos apresentados, ainda existem subconjuntos de algoritmos de aprendizado de máquina que também merecem destaque por suas contribuições e usos atualmente. Como por exemplo o aprendizado semi-supervisionado que combina elementos do aprendizado supervisionado e não supervisionado, aproveitando conjuntos de dados que possuem tanto exemplos rotulados quanto não rotulados (Van Engelen; Hoos, 2020). Ele utiliza os exemplos rotulados para aprender padrões e estruturas nos dados, e então usa essa informação para fazer previsões ou classi-

ficações nos exemplos não rotulados. Esta abordagem é especialmente útil quando rotular dados é caro ou demorado, mas ainda se deseja aproveitar ao máximo os dados disponíveis.

Na técnica de aprendizado de múltiplas tarefas, um único modelo é treinado para realizar várias tarefas relacionadas simultaneamente. Em vez de treinar modelos separados para cada tarefa, o aprendizado de múltiplas tarefas compartilha informações e conhecimentos entre as tarefas, resultando em modelos mais eficientes e robustos (Caruana, 1997). Isso pode levar a melhorias significativas no desempenho, especialmente quando as tarefas têm características semelhantes ou compartilham recursos.

Outro método de aprendizado, conhecido como aprendizado em conjunto, envolve o treinamento de vários modelos de aprendizado de máquina e a combinação de suas previsões para obter um modelo final mais robusto e preciso. Existem várias técnicas de *ensemble*, como *bagging*, *boosting* e *stacking*, cada uma com suas próprias abordagens para combinar os modelos base. *Ensemble learning* é frequentemente usado para reduzir o *overfitting*, melhorar a generalização e aumentar a estabilidade dos modelos.

A técnica de aprendizado baseado em instâncias se concentra em armazenar exemplos de treinamento e fazer previsões com base na similaridade entre novas instâncias e os exemplos existentes. Não envolve a construção explícita de um modelo, mas sim a retenção e consulta dos exemplos de treinamento durante a fase de teste. Algoritmos como *k-Nearest Neighbors* (k-NN) e *Support Vector Machines* (SVM) são exemplos de aprendizado baseado em instâncias. Essa abordagem é útil em problemas nos quais a estrutura dos dados é complexa e não linear.

Por fim, mas não menos importante, estão as redes neurais, que são modelos de aprendizado de máquina inspirados no funcionamento do cérebro humano. Elas consistem em camadas de neurônios interconectados, cada um realizando operações matemáticas nas entradas e passando os resultados para as camadas subsequentes. As redes neurais são capazes de aprender representações complexas e hierárquicas dos dados, tornando-as especialmente eficazes para tarefas como reconhecimento de padrões em imagens, processamento de linguagem natural e previsão.

No contexto de controle e monitoramento de sistemas elétricos, os modelos de aprendizado de máquina podem desempenhar um papel fundamental. Através do uso de algoritmos de aprendizado supervisionado, por exemplo, é possível desenvolver modelos precisos para prever a demanda de energia elétrica, ajudando as empresas a ajustar a produção e minimizar desperdícios. Além disso, técnicas de aprendizado não supervisionado podem identificar padrões e anomalias nos dados de operação, permitindo uma detecção precoce de falhas e manutenção preditiva. Os modelos

de redes neurais, por exemplo, são capazes de analisar grandes volumes de dados em tempo real e tomar decisões rápidas para controlar a distribuição de energia e evitar sobrecargas. Com o auxílio do aprendizado de máquina, os sistemas elétricos podem ser monitorados de forma mais inteligente e eficaz, garantindo um fornecimento confiável de energia elétrica para consumidores e empresas.

2.4 MÉTRICAS UTILIZADAS EM APRENDIZADO DE MÁQUINA

As métricas de avaliação em aprendizado de máquina são ferramentas essenciais que permitem analisar e mensurar o desempenho de modelos de aprendizado, indicando o quão bem um modelo está resolvendo a tarefa para a qual foi treinado. Em um contexto onde a precisão das previsões é crucial, como em sistemas de recomendação, diagnósticos médicos, e detecção de fraudes, escolher e interpretar corretamente as métricas é fundamental para avaliar a efetividade de um modelo e garantir que ele está adequadamente alinhado aos objetivos do problema.

Em termos gerais, as métricas de avaliação servem para medir a diferença entre as previsões do modelo e os valores reais, fornecendo uma base quantitativa para ajustes, comparações e otimizações. Em tarefas de classificação, métricas como acurácia, precisão, revocação e *F1-score* indicam o grau de acertos e o tipo de erros cometidos pelo modelo em classes específicas, ajudando a identificar se o modelo está, por exemplo, gerando muitos falsos positivos ou falsos negativos. Em problemas de regressão, onde o objetivo é prever valores contínuos, métricas como o erro absoluto médio (MAE), o erro quadrático médio (MSE) e o coeficiente de determinação (R^2) medem o quão próximas as previsões estão dos valores reais, capturando tanto a magnitude do erro quanto a capacidade de ajuste do modelo aos dados.

A importância das métricas de avaliação vai além de simplesmente avaliar o desempenho atual de um modelo; elas são cruciais para orientar o processo de desenvolvimento. Durante a fase de treinamento, as métricas ajudam a identificar o *overfitting*, também conhecido como sobreajuste que ocorre quando o modelo se ajusta demais aos dados de treino e não consegue classificar corretamente com dados novos, ou *underfitting*, também conhecido como subajuste, que ocorre quando o modelo se ajusta de menos, não capturando padrões relevantes. Além disso, em aplicações reais, métricas mal escolhidas podem levar a interpretações erradas e até mesmo a impactos negativos. Em um sistema de saúde, por exemplo, uma alta acurácia pode mascarar um problema sério de falsos negativos em um modelo de diagnóstico, colocando em risco a segurança dos pacientes.

Assim, a escolha correta das métricas de avaliação é um passo crítico na construção e validação de modelos, pois permite ao engenheiro ou pesquisador definir metas de desempenho específicas, avaliar a eficácia do modelo com base nos requi-

sitos do problema, comparar diferentes algoritmos e configurações e tomar decisões fundamentadas sobre o uso e aprimoramento do modelo. Dessa forma, as métricas de avaliação não apenas medem a qualidade do modelo, mas também guiam todo o processo de desenvolvimento, aprimorando a eficácia e a confiabilidade das soluções em aprendizado de máquina.

Entre todas as ferramentas disponíveis no quesito de avaliação de modelos de aprendizado, a acurácia é uma das métricas mais utilizadas para avaliar o desempenho de modelos de classificação. Em sua essência, a acurácia mede a proporção de previsões corretas feitas pelo modelo em relação ao número total de previsões. Em problemas de classificação, a acurácia indica o percentual de vezes em que o modelo classificou corretamente as instâncias em suas respectivas classes, oferecendo uma visão geral da capacidade do modelo em identificar as categorias corretamente.

Para um modelo de classificação onde as classes podem ser, por exemplo, “positivo” e “negativo,”, ou "0", "1", "2"e "3,"a acurácia pode ser definida pela seguinte fórmula:

$$\text{Acurácia} = \frac{\text{Previsões Corretas}}{\text{Total de Previsões}} = \frac{\sum_{i=0}^{n} \text{Nº de classificações corretas da classe i}}{\text{Total de Instâncias}} \tag{2.1}$$

onde n é o número de classes analisadas no problema de classificação.

A acurácia é uma métrica intuitiva e fácil de calcular, oferecendo uma visão direta do quão bem o modelo está performando em termos de previsões corretas totais. Em muitos casos, uma acurácia alta indica que o modelo está sendo eficaz em suas previsões, o que, em uma situação ideal de dados balanceados, sugere que ele é confiável e capaz de identificar corretamente as categorias.

Outra métrica importante no estudo e aplicação de aprendizado de máquina é a perda. A métrica de perda (ou *loss*) é uma medida fundamental usada para quantificar o erro de um modelo durante o treinamento. Ela indica o quão longe as previsões do modelo estão dos valores reais esperados. No contexto do aprendizado de máquina, a função de perda é central para ajustar e otimizar o modelo, uma vez que ela orienta o processo de aprendizado ao fornecer retorno sobre a qualidade das previsões em cada iteração de treinamento.

A função de perda transforma o erro das previsões do modelo em um valor numérico, fornecendo uma avaliação do desempenho para cada conjunto de pesos e parâmetros do modelo. Durante o treinamento, o objetivo do modelo é minimizar essa perda, o que significa reduzir o erro entre as previsões e os valores reais. Esse processo é conhecido como minimização da função de perda e é feito geralmente através de métodos de otimização, como o gradiente descendente.

No entanto, é importante distinguir a função de perda da métrica de avaliação. Embora ambas meçam o erro, a função de perda orienta o processo de treinamento, enquanto a métrica de avaliação mede o desempenho final do modelo nos dados de validação ou teste. Em alguns casos, elas são as mesmas (por exemplo, MSE como função de perda e métrica), mas, em outros casos, podem diferir. Em classificação, por exemplo, a perda logarítmica pode ser usada como função de perda, enquanto a acurácia é utilizada como métrica de avaliação final.

3 METODOLOGIA DE DESENVOLVIMENTO

3.1 RECURSOS NECESSÁRIOS

Para o desenvolvimento do trabalho serão utilizados recursos tecnológicos capazes de realizar a simulação de sistemas elétricos de potência e que possibilitam a implementação e utilização de modelos de aprendizado. Com isso definido, os *softwares* a serem empregados para a modelagem e simulação do sistema de potência serão o Anarede e Anatem. O aplicativo Anarede é desenvolvido pelo Centro de Pesquisas de Energia Elétrica (CEPEL) e voltado para a modelagem e análise de sistemas elétricos de potência. Este *software* é amplamente utilizado por empresas de energia, universidades e centros de pesquisa para planejar, operar e otimizar sistemas de transmissão e distribuição de energia elétrica.

Uma das principais funcionalidades do Anarede é a análise de fluxo de carga. O programa permite a determinação das tensões, correntes, potências ativas e reativas, e perdas em cada elemento da rede elétrica, essencial para garantir que o sistema opere dentro dos limites de segurança e eficiência. Além disso, o Anarede realiza cálculos precisos de correntes de curto-circuito para diferentes tipos de faltas (monofásicas, bifásicas e trifásicas). Essas análises são cruciais para o dimensionamento adequado dos equipamentos de proteção e a definição de estratégias de mitigação de falhas.

O Anarede suporta a modelagem detalhada de diversos componentes do sistema elétrico, incluindo geradores, linhas de transmissão, transformadores, cargas, e compensadores estáticos, permitindo uma representação fiel da realidade operacional do sistema. A biblioteca abrangente de modelos de componentes facilita a criação de redes elétricas complexas e detalhadas.

O *software* Anatem, também desenvolvido pelo CEPEL, é uma poderosa ferramenta utilizada para a simulação de sistemas elétricos de potência. Seu principal objetivo é auxiliar no estudo e análise da estabilidade transitória de sistemas de energia elétrica, permitindo que engenheiros e operadores tomem decisões informadas sobre a operação e planejamento da rede.

O Anatem é projetado para realizar simulações detalhadas de estabilidade transitória, que é a capacidade do sistema elétrico de manter a sincronização após perturbações severas, como curtos-circuitos ou perda de grandes blocos de geração. A ferramenta calcula a resposta dinâmica dos geradores e outros componentes do sistema a essas perturbações, ajudando a identificar condições que possam levar a instabilidades ou apagões (Oliveira *et al.*, 1994).

Além disso, o aplicativo permite a modelagem detalhada de uma ampla gama de componentes do sistema de potência, incluindo geradores, linhas de transmissão, transformadores, cargas e dispositivos de compensação. Essa capacidade de modelagem detalhada é crucial para realizar análises precisas e confiáveis, permitindo a inclusão de características específicas de cada equipamento.

O Anatem ainda oferece uma interface gráfica intuitiva que facilita a configuração das simulações e a visualização dos resultados. Os engenheiros podem ver gráficos e diagramas que mostram o comportamento dinâmico do sistema ao longo do tempo, como variações de tensão, ângulo de rotor e frequência dos geradores. Além disso, o *software* gera relatórios detalhados que resumem os principais achados das simulações, proporcionando uma documentação clara e completa para análise posterior, o que o torna uma ótima ferramenta utilizada por empresas e profissionais do setor elétrico para projetar, operar e manter sistemas elétricos de alta tensão.

Para o tratamento dos dados provenientes das simulações do Anatem, será utilizado o aplicativo Matlab. O Matlab, desenvolvido pela MathWorks, é uma plataforma de computação numérica e um ambiente de desenvolvimento amplamente utilizado em diversas áreas da engenharia, ciência, finanças e outras disciplinas que requerem cálculos complexos. Com uma linguagem de programação de fácil utilização e um conjunto robusto de ferramentas, o Matlab facilita a análise de dados, o desenvolvimento de algoritmos e a criação de modelos e aplicativos.

Suas principais funcionalidades incluem cálculos numéricos eficientes e precisos, com uma vasta biblioteca de funções matemáticas para álgebra linear, estatística e otimização. O aplicativo oferece avançadas capacidades de visualização de dados, permitindo a criação de gráficos 2D e 3D interativos, e facilita o desenvolvimento, teste e implementação de algoritmos com uma linguagem de programação de alto nível. Além disso, o Matlab é amplamente utilizado para simulação e modelagem de sistemas dinâmicos através do Simulink, análise de dados e aprendizado de máquina, e permite a integração com outras linguagens de programação e ambientes, bem como a geração de código para sistemas embarcados.

Para o desenvolvimento da aplicação com aprendizado de máquina, foi escolhida a linguagem de programação Python. O Python é uma linguagem de programação de alto nível, interpretada e de propósito geral, conhecida por sua simplicidade, legibilidade e vasta comunidade de usuários. Criada por Guido van Rossum e lançada pela primeira vez em 1991, o Python se destaca por sua sintaxe clara e intuitiva, que facilita a escrita e a manutenção de código. É amplamente utilizada em diversos domínios, incluindo desenvolvimento web, automação, análise de dados, inteligência artificial e aprendizado de máquina, devido a grande quantidade de bibliotecas disponíveis. A biblioteca mais conhecida para trabalhar com aprendizado de máquina é a Tensor-

Flow, mas outras opções populares incluem Keras, PyTorch e Scikit-learn, que serão utilizadas nesse trabalho.

Scikit-learn é uma biblioteca de aprendizado de máquina em Python que oferece ferramentas simples e eficientes para análise e mineração de dados. Construída sobre bibliotecas populares como NumPy, SciPy e Matplotlib, ela é amplamente utilizada para a criação de modelos de aprendizado de máquina. Suas funcionalidades incluem algoritmos de aprendizado supervisionado, como regressão linear, regressão logística, árvores de decisão, SVM e *ensembles*, bem como algoritmos não supervisionados, como *k-means* e *Principal Component Analysis* (PCA). A biblioteca também fornece ferramentas para pré-processamento de dados, incluindo normalização e redução de dimensionalidade, além de ferramentas para validação cruzada e avaliação de desempenho de modelos.

O TensorFlow, desenvolvido pelo Google Brain Team, é uma biblioteca de código aberto para computação numérica e aprendizado profundo, destacando-se por sua flexibilidade e escalabilidade. Ele usa grafos computacionais para representar operações matemáticas, suportando a construção e treinamento de redes neurais profundas, como CNNs, RNNs e GANs, e pode ser implementado em diversos ambientes, desde dispositivos móveis até grandes *clusters* de dados. Keras, uma *API* de alto nível integrada ao TensorFlow, facilita a construção rápida de modelos de aprendizado profundo com uma interface intuitiva e modular. Oferece duas principais *APIs*, *Sequencial* e *Funcional*, e é compatível com diversos *backends*, proporcionando uma coleção extensiva de camadas e ferramentas para pré-processamento de dados e visualização de treinamento.

3.2 MODELAGEM DO SISTEMA ELÉTRICO

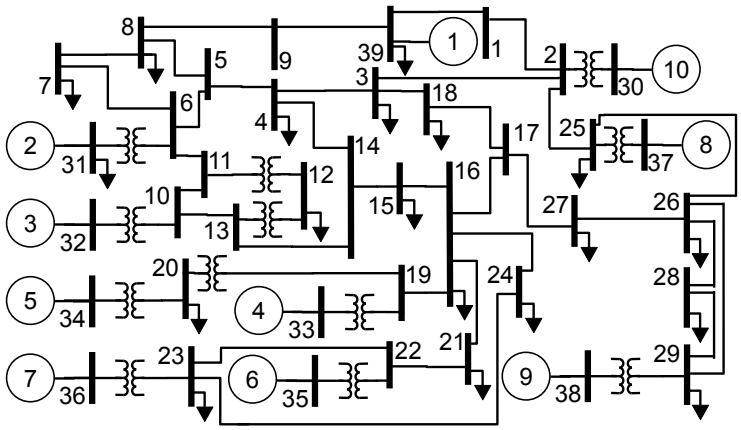
A primeira etapa para a realização do trabalho está na escolha do sistema elétrico a ser estudado. Nesse sentido, foi escolhido o sistema de 39 barras do Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE), também conhecido como o sistema New England de 39 barras, que é um dos sistemas de teste mais utilizados na pesquisa e análise de sistemas elétricos de potência. Este sistema foi desenvolvido para representar a rede elétrica da região da Nova Inglaterra nos Estados Unidos e foi apresentado pela primeira vez no trabalho *"A Practical Method for the Direct Analysis of Transient Stability"* (Athay *et al.*, 1979), sendo amplamente utilizado para estudos de estabilidade, fluxo de carga e outras análises de desempenho do sistema de energia.

O sistema é composto por 39 barras numeradas de 1 a 39, representando pontos de conexão na rede, como subestações, pontos de carga e locais de geração de energia. Essas barras são interconectadas por 46 linhas de transmissão que representam a rede de transmissão de energia elétrica entre geradores e consumidores. Das

39 barras, 10 possuem geradores conectados, numeradas de 30 a 39, representando grandes unidades geradoras, incluindo usinas de energia nuclear, térmica e hidráulica. Além disso, 12 transformadores estão presentes para ajustar os níveis de tensão entre diferentes partes do sistema.

Algumas barras do sistema contêm cargas associadas a elas, representando centros de consumo de energia. As linhas de transmissão possuem impedâncias definidas porresistências (R) e reatâncias (X), além de admitâncias (Y) de *shunt* para modelar as perdas na linha. Cada gerador tem parâmetros específicos, como potência ativa (P), potência reativa (Q), e limites de geração. A FIGURA 5 representa o diagrama unifilar do sistema utilizado no estudo.

FIGURA 5 – SISTEMA DE 39 BARRAS



FONTE: Adaptado de (Canizares *et al*., 2017), (2024).

Para a obtenção dos parâmetros do sistema para modelagem, foram utilizados como base o estudo original do sistema de 1979 e os dados gerados em um trabalho mais recente, (Canizares *et al*., 2017), que apresenta um conjunto de modelos de referência para sistemas de potência, incluindo o sistema estudado, e também apresenta uma série de estudos de caso para demonstrar a utilidade dos modelos de referência. As informações constitutivas das barras do sistema, sendo elas a tensão em pu das barras, o ângulo do fasor de tensão em cada barra, a potência, em MW e MVAr, gerada nas barras que possuem geradores conectados e a potência, em MW e MVAr, consumida nas barras que possuem carga acopladas, foram compiladas na TABELA 1.

TABELA 1 – PARÂMETROS CONSTITUTIVOS DAS BARRAS DO SISTEMA

Barra	Tensão (pu)	Ângulo (°)	Gerador		Carga	
			Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
1	1.047	-8.4	0	0	0	0

Continua...

Barra	Tensão (pu)	Ângulo (°)	Gerador		Carga	
			Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
2	1.049	-5.8	0	0	0	0
3	1.030	-8.6	0	0	322	2.4
4	1.004	-9.6	0	0	500	184
5	1.005	-8.6	0	0	0	0
6	1.008	-7.9	0	0	0	0
7	0.997	-10.1	0	0	233.8	84
8	0.996	-10.6	0	0	522	176
9	1.028	-10.3	0	0	0	0
10	1.017	-5.4	0	0	0	0
11	1.013	-6.3	0	0	0	0
12	1.000	-6.2	0	0	7.5	88
13	1.014	-6.1	0	0	0	0
14	1.012	-7.7	0	0	0	0
15	1.015	-7.7	0	0	320	153
16	1.032	-6.2	0	0	329	32.3
17	1.034	-7.3	0	0	0	0
18	1.031	-8.2	0	0	158	30
19	1.050	-1.0	0	0	0	0
20	0.991	-2.0	0	0	628	103
21	1.032	-3.8	0	0	274	115
22	1.050	0.7	0	0	0	0
23	1.045	0.5	0	0	247.5	84.6
24	1.037	-6.1	0	0	308.6	-92
25	1.058	-4.4	0	0	224	47.2
26	1.052	-5.5	0	0	139	17
27	1.038	-7.5	0	0	281	75.5
28	1.051	-2.0	0	0	206	27.6
29	1.050	0.7	0	0	283.5	26.9
30	1.048	-3.3	250	-	-	-
31	0.982	0	-	-	9.2	4.6
32	0.983	2.6	650	-	-	-
33	0.997	4.2	632	-	-	-
34	1.012	3.2	508	-	-	-
35	1.049	5.6	650	-	-	-
36	1.064	8.3	560	-	-	-
37	1.028	2.4	540	-	-	-
38	1.027	7.8	830	-	-	-
39	1.030	-10.1	1000	-	1104	250

FONTE: O Autor, (2024).

Já os dados constitutivos das linhas do sistema, ou seja, os valores de resistência e impedância em pu, de cada uma das linhas, além da potência reativa na linha

e o valor do *tap* nas linhas em que há a presença de transformadores também foram compilados e são mostrados na TABELA 2.

TABELA 2 – PARÂMETROS CONSTITUTIVOS DAS LINHAS DO SISTEMA

Linha		R (pu)	X (pu)	Y (pu)	Tap
Da Barra	Para Barra				
1	2	0.35	4.11	69.87	-
1	39	0.1	2.5	75	-
2	3	0.13	1.51	25.72	-
2	25	0.7	0.86	14.6	-
3	4	0.13	2.13	22.14	-
3	18	0.11	1.33	21.38	-
4	5	0.08	1.28	13.42	-
4	14	0.08	1.29	13.82	-
5	6	0.02	0.26	4.34	-
5	8	0.08	1.12	14.76	-
6	7	0.06	0.92	11.3	-
6	11	0.07	0.82	13.89	-
7	8	0.04	0.46	7.8	-
8	9	0.23	3.63	38.04	-
9	39	0.1	2.5	120	-
10	11	0.04	0.43	7.29	-
10	13	0.04	0.43	7.29	-
13	14	0.09	01.01	17.23	-
14	15	0.18	2.17	36.6	-
15	16	0.09	0.94	17.1	-
16	17	0.07	0.89	13.42	-
16	19	0.16	1.95	30.4	-
16	21	0.08	1.35	25.48	-
16	24	0.03	0.59	6.8	-
17	18	0.07	0.82	13.19	-
17	27	0.13	1.73	32.16	-
21	22	0.08	1.4	25.65	-
22	23	0.06	0.96	18.46	-
23	24	0.22	3.5	36.1	-
25	26	0.32	3.23	51.3	-
26	27	0.14	1.47	23.96	-
26	28	0.43	4.74	78.02	-
26	29	0.57	6.25	102.9	-
28	29	0.14	1.51	24.9	-
12	11	0.16	4.35	-	1.006
12	13	0.16	4.35	-	1.006
6	31	0	2.50	-	1.070
10	32	0	2.00	-	1.070
19	33	0.07	1.42	-	1.070

Continua...

Linha		R (pu)	X (pu)	Y (pu)	Tap
Da Barra	Para Barra				
20	34	0.09	1.80	-	1.009
22	35	0	1.43	-	1.025
23	36	0.05	2.72	-	1.000
25	37	0.06	2.32	-	1.025
2	30	0	1.81	-	1.025
29	38	0.08	1.56	-	1.025
19	20	0.07	1.38	-	1.060

FONTE: O Autor, (2024).

Com os dados do sistema adquiridos, foi utilizado o *software* Anarede para produzir uma modelagem do sistema de forma digital para que pudesse ser utilizada nas próximas etapas do trabalho.

3.3 SIMULAÇÃO DO SISTEMA ELÉTRICO

Com o sistema modelado, a próxima etapa do trabalho consiste na validação do modelo, ou seja, testar o modelo criado para determinar se seu funcionamento está correto e corresponde à realidade, podendo ser utilizado sem riscos nas etapas posteriores, ou se não está de acordo com a referência estabelecida, sendo necessários ajustes para a correta implementação do sistema elétrico. Para isso, foi utilizado o aplicativo Anatem (CEPEL, 2024), carregado com o modelo do sistema feito no Anarede.

A simulação foi configurada para coletar dados das seguintes grandezas elétricas:

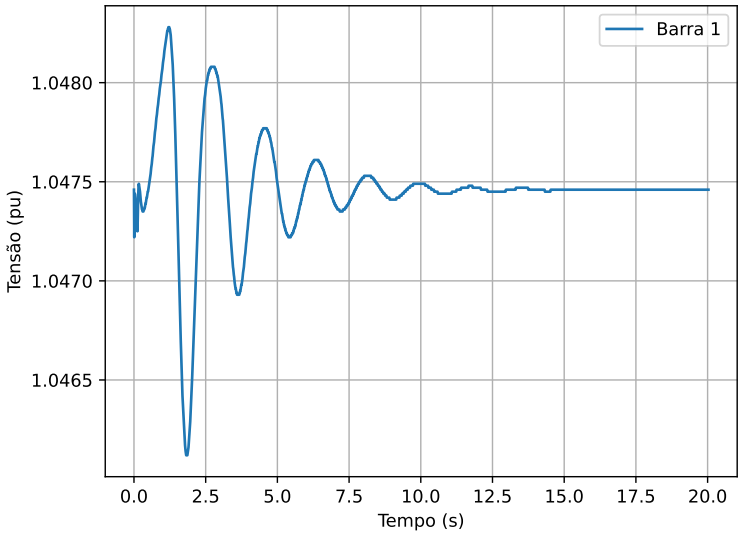
- Tensão (V): Tensão elétrica medida em cada um das barras do sistema;
- Corrente (I): Corrente elétrica medida em cada uma das linhas do sistema;
- Ângulo (θ): Ângulo do fasor de tensão em cada uma das barras do sistema;
- Frequência (F): Frequência elétrica em cada uma das barras do sistema;
- Potência Ativa Gerada (Pger): Potência ativa gerada nas barras com geradores acoplados;
- Potência Reativa Gerada (Qger): Potência reativa gerada nas barras com geradores acoplados;
- Potência Ativa Consumida (Pcar): Potência ativa consumida nas barras com cargas acopladas;

- Potência Reativa Consumida (Qcar): Potência reativa consumida nas barras com cargas acopladas;
- Fluxo de Potência Ativa (Flxa): Fluxo de potência ativa em cada uma das linhas do sistema;
- Fluxo de Potência Reativa (Flxr): Fluxo de potência reativa em cada uma das linhas do sistema.

Os dados foram coletados a cada 10 milissegundos (ms) durante um período de 20 segundos. Esta alta resolução temporal permite uma análise detalhada das possíveis transições e variações rápidas que podem ocorrer no sistema. Além disso, para fim de normalização e compreensão total do sistema, o valor do ângulo do fator de tensão de todas as barras foi subtraído do valor ângulo do fator de tensão da barra 31, assim definindo a barra 31 como barra de referência do sistema e a potência base do sistema foi definida como 100 MVA.

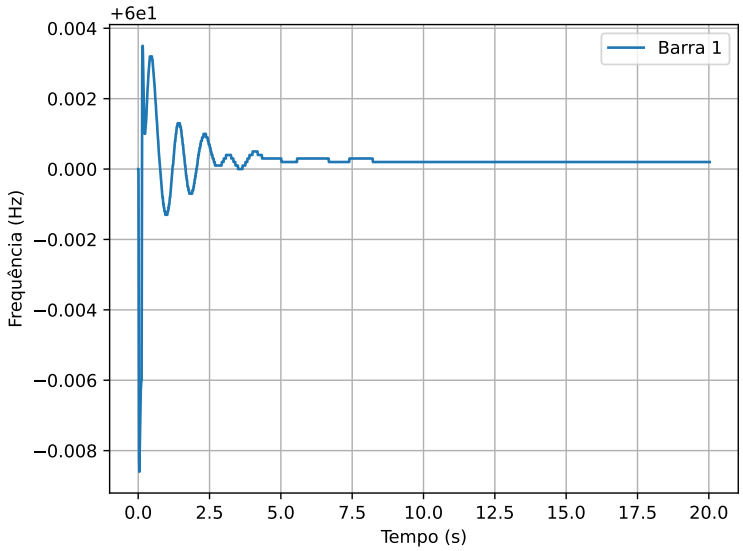
A FIGURA 6, FIGURA 7 e FIGURA 8 são representações gráficas dos resultados obtidos na simulação do sistema em sua operação normal. O código utilizado para a modelagem e simulação do sistema está disponível no Apêndice 1.

FIGURA 6 – SIMULAÇÃO DO SISTEMA EM REGIME PERMANENTE (TENSÃO NA BARRA 1)



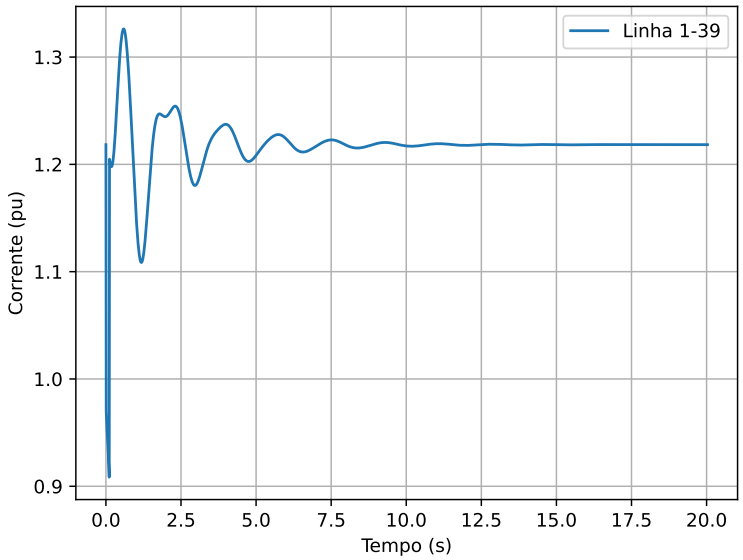
FONTE: O Autor, (2024).

FIGURA 7 – SIMULAÇÃO DO SISTEMA EM REGIME PERMANENTE (FREQUÊNCIA NA BARRA 1)



FONTE: O Autor, (2024).

FIGURA 8 – SIMULAÇÃO DO SISTEMA EM REGIME PERMANENTE (CORRENTE NA LINHA 1-39)



FONTE: O Autor, (2024).

A simulação foi realizada com o foco principal em garantir a precisão do modelo, por isso, após a obtenção de todos os dados simulados foi feita uma etapa de confirmação das leis e princípios do estudo de sistemas elétricos. O primeiro passo consistiu no cálculo das correntes do sistema, dividindo-se a potência aparente (número complexo formado pelo fluxo de potência ativa como parte real e o fluxo de potência reativa como parte imaginária) em cada linha pela tensão da barra em que está conectada a linha. Após todas as correntes terem sido calculadas, elas foram comparadas com as correntes simuladas. Além disso, foi feito o somatório das correntes

em cada uma das barras do sistema, para verificar a consistência da Lei de Kirchhoff das correntes que diz que o somatório das correntes que entram e saem de um nó de um circuito, nesse caso uma barra do sistema, devem ser iguais a zero (0).

Os resultados da simulação mostraram uma forte concordância com os valores teóricos e os dados de referência. As correntes calculadas a partir das potências e tensões coincidem com as correntes simuladas, e a aplicação correta da Lei de Kirchhoff das Correntes complementa os testes demonstrando a coerência dos dados adquiridos. Essa precisão confirma que a simulação reflete fielmente o comportamento real do sistema elétrico modelado.

3.4 SIMULAÇÃO DOS EVENTOS NO SISTEMA ELÉTRICO

Com o modelo do sistema testado e validado, inicia-se uma etapa de simulação dos eventos na rede elétrica com o intuito de gerar a base de dados necessária para a aplicação nos modelos de aprendizado de máquina. Neste trabalho, detalhamos a simulação de cinco tipos de falhas no sistema elétrico: abertura de linha, curto-circuito na barra, afundamento de tensão na barra, curto-circuito na linha e desligamento de barra.

Com o intuito de diversificar ainda mais a base de dados a ser obtida nas simulações a fim de torná-los mais parecidos com possíveis dados reais obtidos em sistemas físicos, foram utilizados artifícios para diferenciar alguns parâmetros do sistema. O primeiro artifício utilizado nas simulações, foi a aplicação de diferentes pontos de operação ao simular as falhas. Pontos de operação do sistema, no contexto de sistemas elétricos, referem-se a condições específicas de funcionamento do sistema sob diferentes cenários de carga e geração de energia. Esses pontos de operação são definidos por valores específicos de tensão, corrente, potência (ativa e reativa), frequência, entre outras grandezas elétricas, nas diversas partes do sistema elétrico, como barras, linhas de transmissão e transformadores. Foram utilizados 5 pontos de operação: o primeiro sendo o caso base do sistema; o segundo sendo um caso com o ponto de operação de +10%, onde a carga do sistema é 10% maior do que a carga base; o terceiro caso com o ponto de operação de +20%, onde a carga do sistema é 20% maior do que a carga base; o quarto caso com o ponto de operação de -10%, onde a carga do sistema é 10% menor do que a carga base e o quinto caso com o ponto de operação de -20%, onde a carga do sistema é 20% menor do que a carga base.

Para calcular os pontos de operação, primeiro é necessário calcular os fatores de participação de cada um dos geradores do sistema. No contexto de sistemas elétricos de potência, o fator de participação de um gerador é uma medida que indica a contribuição relativa de um gerador específico nas variações de frequência ou nas

variações de carga do sistema. Esse fator é crucial para a análise de estabilidade e para a operação econômica do sistema de potência e pode ser calculado dividindo-se a potência ativa gerada por cada um dos geradores pela potência total gerada no sistema.

Os fatores de participação dos gerados de um sistema podem ser calculados seguindo a seguinte fórmula:

$$Fp = \frac{Pg}{Pt} \tag{3.1}$$

onde Fp é o fator de participação de cada gerador, Pg é a potência ativa gerada no respectivo gerador e Pt é a potência ativa total gerada no sistema. Os fatores de participação de cada um dos geradores do sistema foram calculados e podem ser vistos na TABELA 3.

TABELA 3 – FATORES DE PARTICIPAÇÃO

Barra do Gerador	Fator de Participação
30	0,04
31	0,08
32	0,11
33	0,10
34	0,08
35	0,11
36	0,09
37	0,09
38	0,14
39	0,16

FONTE: O Autor, (2024).

Com o fator de participação de cada gerador calculado, é possível calcular os novos valores de carregamento do sistema para os respectivos pontos de operação. Os valores de potência ativa gerada para cada gerador nesses cenários podem ser calculados somando-se a medida do caso base com a multiplicação do parâmetro de carga pelo fator de participação respectivo e a potência ativa total das cargas do sistema, como desenvolvido na equação 3.2. As potências ativas e reativas das cargas do sistema nos pontos de operação podem ser calculados multiplicando-se os valores no caso base pelo valor em percentual do parâmetro de carga acrescido de 1, assim como explicitado na equação 3.3.

$$Pg_C = Pg + (Fp \times P \times Pl_T) \tag{3.2}$$

onde Pg_C é a potência ativa gerada pelo gerador corrigida pelo ponto de operação, Pg é a potência ativa padrão do gerador, Fp é o fator de participação do gerador, P é o

ponto de operação, ou parâmetro de carregamento do sistema e Pl_T é a potência ativa total utilizada pelas cargas do sistema na operação padrão.

$$S_C = S_L \times (1 + P) \tag{3.3}$$

onde S_C é a potência corrigida pelo fator de carga, S_L é a potência padrão das cargas do sistema e P é o parâmetro de carga do ponto de operação. O símbolo utilizado na representação, S_C, denotando potência aparente, foi empregado pois essa fórmula pode ser utilizada tanto para o cálculo da potência ativa (P_C) como da reativa (Q_C). Os valores dos parâmetros do sistema corrigidos pelos pontos de operação podem ser vistos na TABELA 4, TABELA 5, TABELA 6 e TABELA 7.

TABELA 4 – PONTO DE OPERAÇÃO EM -20% DE CARGA

Barra	Gerador		Carga	
	Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
1	-	-	0,0	0
2	-	-	0,0	0
3	-	-	257,6	1,92
4	-	-	400,0	147,2
5	-	-	0,0	0
6	-	-	0,0	0
7	-	-	187,0	67,2
8	-	-	417,6	140,8
9	-	-	0,0	0
10	-	-	0,0	0
11	-	-	0,0	0
12	-	-	6,0	70,4
13	-	-	0,0	0
14	-	-	0,0	0
15	-	-	256,0	122,4
16	-	-	263,2	25,84
17	-	-	0,0	0
18	-	-	126,4	24
19	-	-	0,0	0
20	-	-	502,4	82,4
21	-	-	219,2	92
22	-	-	0,0	0
23	-	-	198,0	67,68
24	-	-	246,9	-73,6
25	-	-	179,2	37,76
26	-	-	111,2	13,6
27	-	-	224,8	60,4
28	-	-	164,8	22,08

Continua...

Barra	Gerador		Carga	
	Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
29	-	-	226,8	21,52
30	200,36	-	0,0	0
31	417,38	-	7,4	3,68
32	520,93	-	0,0	0
33	506,5	-	0,0	0
34	407,12	-	0,0	0
35	520,93	-	0,0	0
36	448,8	-	0,0	0
37	432,77	-	0,0	0
38	665,18	-	0,0	0
39	801,42	-	883,2	200

FONTE: O Autor, (2024).

TABELA 5 – PONTO DE OPERAÇÃO EM -10% DE CARGA

Barra	Gerador		Carga	
	Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
1	-	-	0,0	0
2	-	-	0,0	0
3	-	-	289,8	2,16
4	-	-	450,0	165,6
5	-	-	0,0	0
6	-	-	0,0	0
7	-	-	210,4	75,6
8	-	-	469,8	158,4
9	-	-	0,0	0
10	-	-	0,0	0
11	-	-	0,0	0
12	-	-	6,8	79,2
13	-	-	0,0	0
14	-	-	0,0	0
15	-	-	288,0	137,7
16	-	-	296,1	29,07
17	-	-	0,0	0
18	-	-	142,2	27
19	-	-	0,0	0
20	-	-	565,2	92,7
21	-	-	246,6	103,5
22	-	-	0,0	0
23	-	-	222,8	76,14
24	-	-	277,7	-82,8
25	-	-	201,6	42,48
26	-	-	125,1	15,3

Continua...

Barra	Gerador		Carga	
	Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
27	-	-	252,9	67,95
28	-	-	185,4	24,84
29	-	-	255,2	24,21
30	225,18	-	0,0	0
31	469,09	-	8,3	4,14
32	585,46	-	0,0	0
33	569,25	-	0,0	0
34	457,56	-	0,0	0
35	585,46	-	0,0	0
36	504,4	-	0,0	0
37	486,38	-	0,0	0
38	747,59	-	0,0	0
39	900,71	-	993,6	225

FONTE: O Autor, (2024).

TABELA 6 – PONTO DE OPERAÇÃO EM +10% DE CARGA

Barra	Gerador		Carga	
	Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
1	-	-	0,0	0
2	-	-	0,0	0
3	-	-	354,2	2,64
4	-	-	550,0	202,4
5	-	-	0,0	0
6	-	-	0,0	0
7	-	-	257,2	92,4
8	-	-	574,2	193,6
9	-	-	0,0	0
10	-	-	0,0	0
11	-	-	0,0	0
12	-	-	8,3	96,8
13	-	-	0,0	0
14	-	-	0,0	0
15	-	-	352,0	168,3
16	-	-	361,9	35,53
17	-	-	0,0	0
18	-	-	173,8	33
19	-	-	0,0	0
20	-	-	690,8	113,3
21	-	-	301,4	126,5
22	-	-	0,0	0
23	-	-	272,3	93,06
24	-	-	339,5	-101,2

Continua...

Barra	Gerador		Carga	
	Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
25	-	-	246,4	51,92
26	-	-	152,9	18,7
27	-	-	309,1	83,05
28	-	-	226,6	30,36
29	-	-	311,9	29,59
30	274,82	-	0,0	0
31	572,51	-	10,1	5,06
32	714,54	-	0,0	0
33	694,75	-	0,0	0
34	558,44	-	0,0	0
35	714,54	-	0,0	0
36	615,60	-	0,0	0
37	593,62	-	0,0	0
38	912,41	-	0,0	0
39	1099,29	-	1214,4	275

FONTE: O Autor, (2024).

TABELA 7 – PONTO DE OPERAÇÃO EM +20% DE CARGA

Barra	Gerador		Carga	
	Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
1	-	-	0,0	0
2	-	-	0,0	0
3	-	-	386,4	2,88
4	-	-	600,0	220,8
5	-	-	0,0	0
6	-	-	0,0	0
7	-	-	280,6	100,8
8	-	-	626,4	211,2
9	-	-	0,0	0
10	-	-	0,0	0
11	-	-	0,0	0
12	-	-	9,0	105,6
13	-	-	0,0	0
14	-	-	0,0	0
15	-	-	384,0	183,6
16	-	-	394,8	38,76
17	-	-	0,0	0
18	-	-	189,6	36
19	-	-	0,0	0
20	-	-	753,6	123,6
21	-	-	328,8	138
22	-	-	0,0	0

Continua...

Barra	Gerador		Carga	
	Pg (MW)	Qg (MVAr)	Pg (MW)	Qg (MVAr)
23	-	-	297,0	101,52
24	-	-	370,3	-110,4
25	-	-	268,8	56,64
26	-	-	166,8	20,4
27	-	-	337,2	90,6
28	-	-	247,2	33,12
29	-	-	340,2	32,28
30	299,64	-	0,0	0
31	624,22	-	11,0	5,52
32	779,08	-	0,0	0
33	757,5	-	0,0	0
34	608,88	-	0,0	0
35	779,07	-	0,0	0
36	671,2	-	0,0	0
37	647,23	-	0,0	0
38	994,82	-	0,0	0
39	1198,58	-	1324,8	300

FONTE: O Autor, (2024).

Outro artifício utilizado para criar variabilidade nos dados foi o estabelecimento de durações dos eventos em cada uma das simulações realizadas. Durante a execução de cada simulação, o computador decide um tempo de duração do evento entre 0,040 e 0,080 segundos. Essa técnica permite uma amostragem de dados mais compatível com a realidade, onde não se pode determinar com exatidão que eventos de mesmo tipo ocorrerão em um mesmo espaço de tempo.

Em cada uma das simulações dos diferentes tipos de eventos estudados neste trabalho, os pontos de operação e tempo de ocorrência do evento foram definidos aleatoriamente na execução da simulação, tornando o processo mais dinâmico e diminuindo ao máximo o viés humano. Além disso, foi definido para que todos os eventos simulados começassem logo no início da simulação, gerando dados mais relevantes durante todo o período da simulação.

A primeira instância de eventos simulada foi a abertura de linha. Na abertura de linha, uma conexão entre duas barras, ou seja uma linha, é interrompida, criando uma separação física no sistema. Isso pode resultar em uma perda de continuidade de energia entre as áreas conectadas pela linha. A simulação envolve a interrupção repentina da corrente e a avaliação do impacto dessa falha na estabilidade do sistema e na distribuição de energia. Nesse sentido, foram feitas 46 simulações, uma para cada linha do sistema e os dados adquiridos dessa e das simulações dos outros tipos de

falhas foram os mesmos obtidos na simulação do sistema em regime permanente para a validação do modelo.

O segundo tipo de falha simulado foi o curto-circuito na barra. Um curto-circuito na barra ocorre quando há uma conexão direta entre a barra e o terra, causando uma sobrecarga significativa de corrente. Esta falha pode resultar em danos graves aos equipamentos e interrupção do fornecimento de energia. A simulação inclui o aumento súbito da corrente na barra afetada e a análise dos efeitos desse curto-circuito nas barras adjacentes. Nessa etapa foram feitas 39 simulações, sendo cada uma referente a uma das barras do circuito.

O terceiro evento simulado foi o afundamento de tensão na barra. Um afundamento de tensão na barra refere-se a uma queda temporária e significativa na tensão em uma barra específica do sistema. Isso pode ser causado por uma carga pesada repentina ou por problemas de regulação de tensão. A simulação envolve a redução abrupta da tensão na barra afetada e a avaliação dos efeitos dessa queda de tensão nos equipamentos conectados a ela. Assim sendo, também foram realizadas 39 simulações referentes a cada uma das barras.

O quarto tipo de falta emulado se trata do curto-circuito na linha. Um curto-circuito na linha ocorre quando há um curto entre duas linhas de transmissão ou entre uma linha e o terra. Isso pode levar a uma sobrecarga de corrente e danos aos equipamentos. A simulação inclui a avaliação do aumento repentino de corrente na linha afetada e a análise dos efeitos desse curto-circuito nas barras conectadas à linha.

Por fim, o último tipo de evento estudado e simulado neste trabalho foi o desligamento de barra. O desligamento de barra envolve a desconexão de uma barra específica do sistema elétrico. Isso pode ser feito por motivos de manutenção ou para isolar uma parte do sistema durante uma falha. A simulação inclui a interrupção controlada do suprimento de energia na barra desligada e a análise dos efeitos desse desligamento nas barras adjacentes.

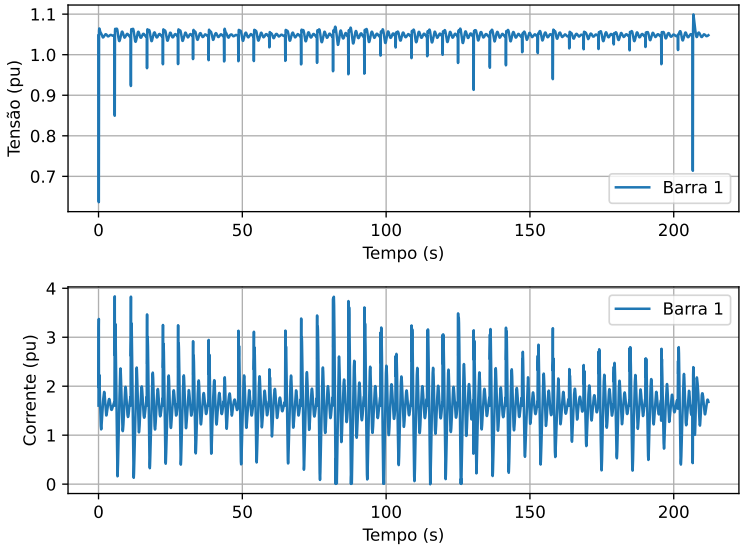
Todas as simulações foram feitas com parâmetros semelhantes à simulação do sistema em regime permanente, mas contendo um tempo de simulação maior do que os 10 segundos da simulação em regime permanente, mas ainda com amostragens sendo feitas a cada 10 milissegundos das grandezas físicas de tensão, corrente, frequência, etc. Isso foi feito para que os dados obtidos nas simulações fossem semelhantes aos dados que são adquiridos fisicamente nos sistemas elétricos por meio de PMUs.

Uma Unidade de Medição Fasorial (PMU) é um dispositivo usado em sistemas elétricos para medir com precisão as grandezas elétricas, como tensão e corrente, em tempo real. A principal função de uma PMU é fornecer informações sincrofasoriais, ou seja, medidas de fase e amplitude dos sinais elétricos em relação a uma referência

comum de tempo, chamada de "fase"(Dusabimana; Yoon, 2020). Esses dados sincrofasoriais são essenciais para monitorar a estabilidade, o desempenho e a segurança dos sistemas elétricos em tempo real, permitindo a detecção rápida de eventos como distúrbios de frequência, oscilações de energia e falhas no sistema. As PMUs desempenham um papel fundamental na modernização e na operação eficiente dos sistemas elétricos, contribuindo para a manutenção da confiabilidade e da qualidade do fornecimento de energia elétrica.

A FIGURA 9 representa um exemplo das características físicas capturadas por uma PMU durante um evento. O tipo de falha em questão utilizado para a obtenção dos gráficos foi o afundamento de tensão na barra 1 do sistema e os dados obtidos se referem as amostras de tensão e corrente em uma das linhas da barra em questão.

FIGURA 9 – EXEMPLO DE EVENTO CAPTURADO POR UMA PMU

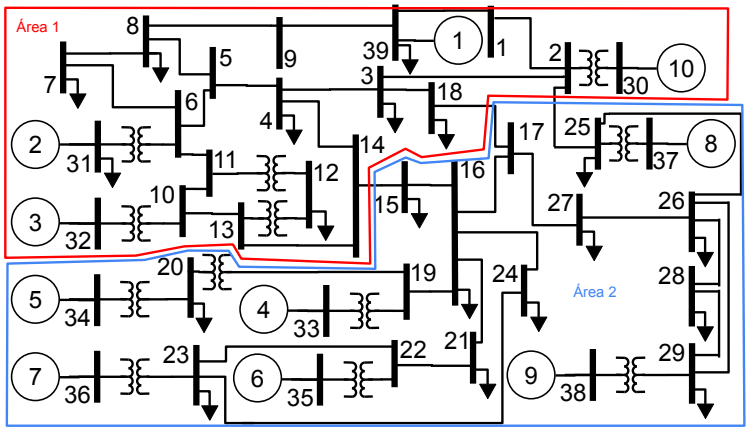


FONTE: O Autor, (2024).

3.5 PRÉ-PROCESSAMENTO

Com os dados dos eventos obtidos torna-se necessária a preparação dos dados para a utilização nos modelos de AM. Sendo assim, o sistema elétrico foi dividido em duas áreas, que podem ser vistas na FIGURA 10. Foi criado um script em Matlab responsável por rotular os dados das simulações dos eventos de acordo com a área em que ocorreram. Por exemplo, para um dado de um evento ocorrido na barra 6 (pertencente a área 1 do sistema) o programa irá rotular esse dado como zero “0”, e para um dado de um evento ocorrido na barra 21 (pertencente a área 2 do sistema) o programa irá rotular esse dado como um “1”.

FIGURA 10 – SISTEMA DE 39 BARRAS DIVIDIDO EM 2 ÁREAS

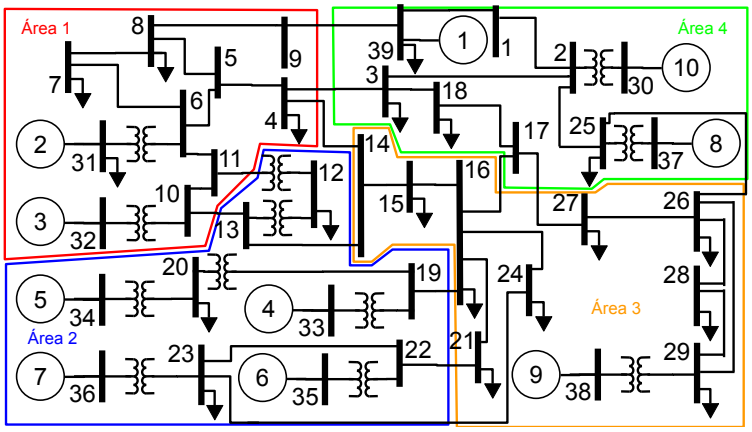


FONTE: O Autor, (2024).

Além disso, com o intuito de tornar o conjunto de dados mais compatível com algo que é mais comum em grandes sistemas interligados, foi considerado que apenas as barras localizadas na área dois do sistema são propriedade da concessionária que vai aplicar o sistema de localização. Portanto, apenas os dados medidos nas barras da área 2 foram considerados para serem utilizados na aplicação de AM.

Posteriormente, o mesmo processo de rotulação dos dados e divisão em dados de treinamento e validação foi realizado novamente, mas dessa vez foi feito para um sistema dividido em quatro áreas, como pode ser verificado na FIGURA 11.

FIGURA 11 – SISTEMA DE 39 BARRAS DIVIDIDO EM 4 ÁREAS



FONTE: O Autor, (2024).

Após a rotulação dos dados das falhas, o programa em Matlab separa os dados dos eventos em um conjunto de treinamento e um conjunto para validação do modelo. O conjunto de dados de treinamento é o subconjunto de dados usado para treinar o modelo de aprendizado de máquina. Este conjunto contém amostras de entrada e suas

respectivas saídas ou rótulos, e é usado para ajustar os parâmetros internos do modelo. O principal objetivo desse conjunto de informações é permitir que o modelo aprenda as relações subjacentes entre as entradas e as saídas. Durante o treinamento, o algoritmo ajusta seus parâmetros (por exemplo, pesos em uma rede neural) para minimizar a função de perda, que mede o quão bem o modelo está se ajustando aos dados de treinamento.

O conjunto de dados de validação é um subconjunto separado de dados que não é usado durante o treinamento do modelo, mas sim para avaliar o desempenho do modelo durante o treinamento. Este conjunto também contém entradas e suas respectivas saídas. O principal objetivo do conjunto de dados de validação é fornecer uma avaliação imparcial do modelo enquanto ele é treinado. Ele ajuda a monitorar o desempenho do modelo em dados que ele não viu durante o treinamento, permitindo detectar o *overfitting* (quando o modelo aprende os detalhes e ruídos do conjunto de treinamento a ponto de prejudicar seu desempenho em novos dados).

Para cada classe de eventos na rede elétrica simulados, os dados foram separados de forma que 80% deles fossem destinados ao treinamento dos modelos e os outros 20% ficassem destinados à validação dos modelos.

3.6 CRIAÇÃO DOS MODELOS DE APRENDIZADO

Tendo em vista tudo que já foi discutido na revisão bibliográfica e considerando os objetivos requisitados para a realização do trabalho, algumas condições e características dos modelos podem ser adquiridas. Como os dados obtidos nas simulações já foram rotulados, o aprendizado aplicado será do tipo supervisionado. Além disso, considerando que o sistema de localização resultará em resultados onde as saídas são discretas e o fato de que o sistema deve ter um baixo tempo de resposta por se tratar de um monitoramento em tempo real do sistema elétrico, é crucial escolher modelos de aprendizado de máquina que ofereçam um bom equilíbrio entre precisão e eficiência computacional. Sendo assim, alguns modelos que são adequados para essas necessidades foram inicialmente considerados, sendo eles:

- Árvore de Decisão Floresta Aleatória;
- *K-Nearest Neighbors* (K-NN);
- Máquinas de Vetores de Suporte (SVM);
- Regressão Logística;
- Redes Neurais Artificiais (Simples).

As Árvores de Decisão são notavelmente rápidas tanto no treinamento quanto na predição, tornando-as ideais para aplicações em tempo real. Além disso, são facilmente interpretáveis, permitindo uma compreensão clara das decisões do modelo. No entanto, uma desvantagem significativa é a tendência ao *overfitting*, especialmente se a árvore não for adequadamente podada, o que a torna muito complexa, ajustando-se excessivamente aos dados de treinamento, incluindo ruídos e variações específicas, o que a torna menos capaz de generalizar para novos dados.. Além disso, elas podem ser bastante sensíveis a pequenas variações nos dados de entrada, o que pode resultar em modelos instáveis.

As Florestas Aleatórias, que são um conjunto de várias árvores de decisão, oferecem maior precisão e são mais robustas, reduzindo a tendência ao *overfitting*. Elas são também menos sensíveis a dados discrepantes e variações nos dados, o que aumenta sua estabilidade. Um benefício adicional é a capacidade de fornecer estimativas da importância das variáveis. Contudo, essa abordagem é mais complexa e demorada para treinar, e embora a predição seja rápida, pode não ser tão imediata quanto a de uma única árvore de decisão.

O modelo *K-Nearest Neighbors* (K-NN) é simples de implementar e frequentemente eficaz para problemas de classificação. Sua natureza não paramétrica significa que ele não faz suposições fortes sobre a distribuição dos dados. Entretanto, uma desvantagem crítica é que o K-NN pode ser lento para grandes conjuntos de dados, pois precisa calcular distâncias para todos os pontos no conjunto de treinamento. Além disso, o desempenho do K-NN depende fortemente da normalização ou padronização dos dados, o que pode exigir etapas adicionais de pré-processamento.

As Máquinas de Vetores de Suporte (SVM) são eficazes em espaços de alta dimensão e focam em maximizar a margem entre classes, o que pode melhorar a generalização do modelo. Apesar disso, o treinamento das SVMs pode ser demorado e exigir ajuste cuidadoso dos hiperparâmetros e da escolha do kernel, embora a predição seja geralmente rápida uma vez que o modelo esteja treinado.

A Regressão Logística é uma opção simples e fácil de implementar, oferecendo tanto treinamento quanto predição rápidos. Sua simplicidade também facilita a interpretação dos resultados. No entanto, assume uma relação linear entre as características e o logaritmo das odds, o que pode não capturar adequadamente todas as relações presentes nos dados, limitando seu desempenho em alguns casos.

Já as Redes Neurais Artificiais (simples) são altamente flexíveis e podem modelar relações complexas e não lineares nos dados. Com arquiteturas otimizadas, elas podem ser bastante eficientes em termos de tempo de resposta, tornando-as adequadas para aplicações em tempo real. Contudo, o treinamento pode ser demorado, e as redes neurais são geralmente menos interpretáveis do que modelos mais simples,

o que pode dificultar a compreensão das decisões tomadas pelo modelo.

Dada a necessidade de baixíssimo tempo de resposta e saídas discretas, a Árvore de Decisão e a Floresta Aleatória são boas opções iniciais devido à sua velocidade de predição e robustez. A Floresta Aleatória, em particular, pode oferecer um bom compromisso entre precisão e tempo de resposta se o sistema tiver capacidade computacional suficiente.

Para cenários onde a simplicidade e a velocidade são cruciais, a Regressão Logística e *K-Nearest Neighbors* (com otimização para busca de vizinhos) também são opções viáveis. Embora a Regressão Logística seja mais interpretável, o K-NN pode oferecer uma boa precisão se os dados forem adequadamente normalizados. Finalmente, uma Rede Neural Artificial Simples pode ser utilizada se houver necessidade de capturar relações complexas, desde que a arquitetura da rede seja otimizada para tempo de resposta rápido.

Entre as opções apresentadas, foram escolhidas dois tipos de algoritmos para a resolução do problema, o primeiro sendo a utilização de um modelo de regressão logística e o segundo sendo a utilização de uma rede neural artificial. Como o modelo de aprendizado de máquina por regressão logística é, como descrito na literatura à vista, por si só mais simples e fácil de implementar, os primeiros testes do sistema foram feitos utilizando essa metodologia.

Foi desenvolvido um programa em Python utilizando a biblioteca scikit-learn para treinar e avaliar um modelo de Regressão Logística. O código calcula a média e o desvio padrão para os dados de treinamento e aplica a normalização a essas informações, transformando-os para ter média 0 e desvio padrão 1. Depois, aplica a mesma transformação (usando os parâmetros ajustados no treinamento) aos dados de teste, garantindo que ambos conjuntos estejam na mesma escala. O modelo de regressão logística foi configurado com parâmetros empíricos que estão dispostos no QUADRO 1 e após a configuração do modelo, o mesmo passou pela etapa de treinamento com os dados de treinamento e posteriormente pela etapa de validação do modelo com os dados de teste.

QUADRO 1 – PARÂMETROS DO MODELO DE REGRESSÃO LOGÍSTICA

Parâmetro	Valor	Descrição
class_weight	balanced	Ajusta os pesos das classes inversamente proporcionais as suas frequências no conjunto de treinamento
solver	'lbfgs'	Algoritmo de otimização Quasi-Newton.
max_iter	1000	Número máximo de iterações do algoritmo de otimização.
tol	1e-6	Tolerância para a convergência.
multi_class	'ovr'	Estrutura de classificação One-vs-Rest para problemas multiclasses.

FONTE: O Autor, (2024).

Com o intuito de testar se para o problema proposto o modelo em questão seria uma opção válida, os dados utilizados para treinamento e validação do modelo de regressão logística também foram simplificados, sendo compostos de grandezas físicas retiradas de apenas um tipo de evento no sistema elétrico, sendo ele o afundamento de tensão, com a configuração do sistema dividido em duas partes como mostrado na Figura 6, ou seja, o modelo deveria decidir em qual das duas áreas do sistema elétrico houve o evento baseado nos dados medidos das grandezas elétricas do sistema.

O segundo modelo de AM aplicado foi uma Rede Neural Perceptron Multicamada (MLP). Esse protótipo também foi treinado com os dados de um tipo de falha, que foram normalizados anteriormente, e seus parâmetros estão disponíveis para visualização no QUADRO 2. O código referente a criação dos dois modelos está presente no Apêndice 2.

QUADRO 2 – PARÂMETROS DO MODELO DE REDE NEURAL MLP

Parâmetro	Valor	Descrição
solver	'lbfgs'	Algoritmo de otimização Quasi-Newton.
alpha	1e-5	Parâmetro de regularização L2.
hidden_layer_sizes	Não especificado (opcional: (5, 2))	Tamanho das camadas ocultas.
random_state	1	Semente para geração de números aleatórios.
max_iter	1000	Número máximo de iterações do algoritmo de otimização.

FONTE: O Autor, (2024).

O treinamento de uma rede neural é um processo iterativo que visa ajustar os pesos das conexões entre as unidades de processamento (neurônios) da rede para que ela possa aprender a mapear corretamente os dados de entrada para a saída desejada. Os dados de treinamento são apresentados à rede uma amostra por vez, e a saída da rede é comparada com a saída desejada (também conhecida como rótulo ou label) para aquela amostra. Com base na diferença entre a saída da rede e a saída desejada, um erro é calculado. O objetivo do treinamento é minimizar esse erro, ajustando os pesos da rede de forma que a saída da rede para cada amostra seja o mais próximo possível da saída desejada. O treinamento de uma rede neural geralmente é concluído quando o erro da rede não diminui mais para um conjunto de validação de dados ou quando um número máximo de épocas (iterações) de treinamento é atingido. Uma vez treinada, a rede neural pode ser usada para fazer previsões para dados não vistos durante seu treinamento.

Com base nos resultados obtidos nessa primeira etapa de aplicação dos modelos, que serão discutidos no capítulo de resultados deste trabalho, foi decidido que a utilização de um modelo mais complexo é mais indicada na continuação do trabalho, portanto, a aplicação de AM empregada nos testes futuros se encaixa na categoria das redes neurais. Com isso, foi desenvolvida mais uma rede neural, com fundamentos diferentes da utilizada anteriormente, para fazer o teste com todos os dados de eventos simulados. Os parâmetros dessa nova rede neural, que foi desenvolvida com o apoio da API Keras estão dispostos no QUADRO 3.

Os procedimentos de testes realizados com esse último modelo são mais completos, mas também consistem na normalização dos dados de entrada e uma etapa de treinamento e por fim uma etapa de teste. Por ser um modelo mais completo, com inúmeros pontos a serem levados em conta, algumas métricas de eficiência do modelo foram empregadas no desenvolvimento desse último processo, que não tinham sido utilizadas nos modelos anteriores e todos esses aspectos são discutidos no próximo capítulo, destinado para os resultados de todos os protótipos criados neste trabalho. O código criado para definição do modelo pode ser visto no Apêndice 3

QUADRO 3 – PARÂMETROS DO MODELO DE REDE NEURAL COM KERAS

Parâmetro	Valor	Descrição
tf.keras.Input	shape=(n,)	Dimensão do vetor de entrada.
Dense (camada 1)	60, activation="relu", kernel_regularizer=tf.keras.regularizers.l2(0.0000)	Camada densa com 60 neurônios, ativação ReLU e regularização L2.
Dense (camada 2)	20, activation="relu", kernel_regularizer=tf.keras.regularizers.l2(0.0000)	Camada densa com 20 neurônios, ativação ReLU e regularização L2.
Dense (camada de saída)	4, activation="softmax"	Camada de saída com 4 neurônios e ativação softmax.
loss	SparseCategoricalCrossentropy	Função de perda para classificação multi-classe de classes inteiras
optimizer	Adam(0.001)	Otimizador Adam com taxa de aprendizado de 0.001.
epochs	20	Número de épocas para treinamento do modelo.

FONTE: O Autor, (2024).

3.7 ATUALIZAÇÃO DO MODELO DE APRENDIZADO

Tendo em mãos os resultados preliminares conseguidos com a aplicação da rede neural desenvolvida com a API Keras, deu-se início ao processo de atualização e aprimoramento do modelo, com o intuito de melhorar o desempenho do projeto. Para melhorar os resultados de um modelo de rede neural, pode-se considerar várias abordagens que envolvem ajustes na arquitetura, técnicas de regularização, otimização de hiperparâmetros e aprimoramento dos dados.

Nesse sentido, foram exploradas diversas abordagens para aprimorar o modelo protótipo inicial, composto por três camadas (entrada, saída e uma camada intermediária) treinado por 20 épocas. As principais estratégias adotadas envolveram a variação no número de épocas de treinamento, o ajuste no número de neurônios por camada, a adição de uma nova camada intermediária e a modificação dos termos de regularização.

O primeiro conjunto de atualizações empregado para a melhoria do sistema, consistiu na alteração do número de épocas de treinamento. Encontrar o número adequado de épocas é essencial para equilibrar o aprendizado pois com poucas épocas, o modelo pode não aprender padrões suficientes dos dados, resultando em baixa acurácia tanto no conjunto de treinamento quanto no de validação, se enquadrando em uma situação de *underfitting*. Porém, um número excessivo de épocas pode levar o modelo a memorizar os dados de treinamento, prejudicando sua generalização para dados novos se enquadrando em uma situação de *overfitting*. Inicialmente, o modelo havia sido treinado por 20 épocas, então para identificar o ponto ideal de treinamento, foram realizados experimentos treinando a mesma rede neural com diferentes números

de épocas, sendo variados entre 20 e 110 épocas em passos de 10.

O segundo tipo de modificações na rede neural como intuito de melhoria de desempenho, se deu na alteração do número de neurônios na primeira camada, ou seja, o número de neurônios da camada de entrada. Incrementar os neurônios pode permitir que o modelo capture padrões mais complexos, visto que cada neurônio em uma rede neural aprende uma característica dos dados e com mais neurônios, a rede tem uma capacidade maior de aprender padrões e nuances mais detalhados e complexos, o que é especialmente útil para tarefas que envolvem dados ricos e de alta dimensionalidade. Porém, isso deve ser feito com cautela visto que redes muito grandes têm maior tendência ao *overfitting* (ajuste excessivo aos dados de treinamento) e demandam mais recursos computacionais, tanto para o treinamento quanto para a execução. Para esse caso, o estudo feito consistiu em variar o número de neurônios entre 60 e 150, em passos de 10.

A terceira variante ocorreu na forma de alteração do número de neurônios na segunda camada, ou seja, o número de neurônios da camada intermediária. Esse caso foi realizado pois, assim como o segundo conjunto de testes com ajustes, o aumento de neurônios está relacionado a uma melhora na resposta do sistema. Para esse conjunto de simulações, a variação do número de neurônios foi de 20 até 110, em passos de 10.

A quarta iteração na mudança de parâmetros da rede consistiu na adição de mais uma camada intermediária no modelo. Aumentar o número de camadas em uma rede neural, transformando-a em uma rede profunda, pode melhorar o desempenho ao permitir que a rede aprenda hierarquias de características dos dados, capturando desde padrões simples até detalhes complexos. Nas camadas iniciais, a rede aprende características básicas, enquanto nas camadas mais profundas, ela combina essas informações em padrões mais complexos. Esse tipo de alteração é eficaz mas aumenta a propensão da rede ao *overfitting* e à necessidade de dados de treinamento maiores. A aplicação dessa iteração consistir em criar uma camada nova antes da camada de saída e variar o números de neurônios dela entre 10 e 100, em passos de 10.

A última alteração de parâmetros nessa etapa do trabalho se baseou na alteração do termo de regularização das camadas. Ajustar o termo de regularização nas camadas de uma rede neural ajuda a melhorar o desempenho ao controlar a complexidade do modelo e reduzir o risco de *overfitting*, que ocorre quando a rede se ajusta excessivamente aos dados de treinamento e perde sua capacidade de generalização. A regularização ajuda a estabilizar o treinamento, proporcionando uma representação mais generalizada dos dados e, consequentemente, melhorando a precisão do modelo em dados novos. Nessa aplicação, o termo de regularização (λ) das camadas, foi alternado entre diversos valores entre 0,00 e 0.45.

Após essas tentativas, tendo se chegado em resultados que serão discutidos

no próximo capítulo, foi decidido por fazer novos testes com o intuito de melhorar o desempenho da rede neural com base em alterações dos parâmetros, utilizando-se para isso, uma abordagem mais robusta por meio da biblioteca Optuna.

O Optuna é uma biblioteca de otimização de hiperparâmetros que realiza uma busca automatizada e eficiente dos melhores valores para melhorar o desempenho de um modelo de aprendizado de máquina. O Optuna utiliza uma técnica chamada "Otimização Bayesiana,"que busca equilibrar exploração (testar diferentes valores) e exploração (aprimorar valores já promissores), tornando-o mais eficiente que uma busca manual ou *grid search*. Ele é ideal para problemas com uma grande quantidade de hiperparâmetros e é especialmente vantajoso em redes neurais, onde diferentes combinações de parâmetros (número de neurônios, taxa de aprendizado, número de camadas, etc.) podem impactar consideravelmente o desempenho final do modelo.

No código implementado, o Optuna é utilizado para definir e buscar os melhores valores para alguns dos hiperparâmetros da rede neural. O código cria uma função chamada objetivo, que define os parâmetros a serem otimizados para o modelo neural e realiza o treinamento com uma rede especificada. Nessa função, o Optuna usa um comando para indicar ao algoritmo que ele deve buscar valores inteiros em intervalos pré-definidos, como o número de neurônios em cada camada da rede e o número de épocas de treinamento. Isso permite que o Optuna teste automaticamente várias configurações de rede, encontrando aquelas que melhoram a precisão do modelo nos dados de validação.

Na função objetivo, são definidos os hiperparâmetros a serem otimizados, que nesse caso são: o número de neurônios na primeira camada; o número de neurônios na segunda camada; o número de neurônios na terceira camada e o número de épocas de treinamento. A cada execução da função, o Optuna define valores diferentes para esses parâmetros e treina a rede neural para verificar a performance com essa nova configuração. A rede neural é compilada com a função de perda, que é usada para problemas de classificação com múltiplas classes, e é otimizada com uma taxa de aprendizado fixa. Ao final do treinamento, a função retorna a métrica de precisão calculada no conjunto de validação, permitindo ao Optuna avaliar o desempenho dessa configuração específica de parâmetros.

Para encontrar os melhores hiperparâmetros, o código cria um objeto de estudo do Optuna, especificando que o objetivo é maximizar a precisão da rede neural. O método realiza a otimização, executando a função objetivo por um número determinado de vezes, testando diferentes combinações de hiperparâmetros e selecionando a melhor combinação ao final. Nesse trabalho em específico foi definido que a função objetivo fosse executada 10 vezes. O teste foi construído com um parâmetro que indica que a busca deve focar em maximizar a precisão do modelo. O código utilizado para

aprimoramento do modelo utilizando o Optuna está disponível no Apêndice 4.

Com os melhores hiperparâmetros encontrados, disponíveis no QUADRO 4 o código constrói um novo modelo de rede neural usando esses valores ideais, garantindo uma configuração otimizada para o treinamento final. O modelo é então treinado com esses parâmetros ideais e sua precisão é testada no conjunto de validação. A configuração final do Optuna, com os melhores hiperparâmetros, é armazenada na memória do sistema, que é então aplicado na construção final da rede neural. Por fim, a precisão da rede e um conjunto de outras métricas são geradas para avaliar a qualidade da classificação, proporcionando uma avaliação completa do desempenho da rede neural.

QUADRO 4 – PARÂMETROS DO MODELO DE REDE NEURAL OBTIDOS COM OPTUNA

Parâmetro	Valor	Descrição
tf.keras.Input	shape=(n,)	Dimensão do vetor de entrada.
Dense (camada 1)	70, activation="relu", kernel_regularizer=tf.keras.regularizers.l2(0.0000)	Camada densa com 70 neurônios, ativação ReLU e regularização L2.
Dense (camada 2)	40, activation="relu", kernel_regularizer=tf.keras.regularizers.l2(0.0000)	Camada densa com 40 neurônios, ativação ReLU e regularização L2.
Dense (camada 3)	100, activation="relu", kernel_regularizer=tf.keras.regularizers.l2(0.0000)	Camada densa com 100 neurônios, ativação ReLU e regularização L2.
Dense (camada de saída)	4, activation="softmax"	Camada de saída com 4 neurônios e ativação softmax.
loss	SparseCategoricalCrossentropy	Função de perda para classificação multi-classe de classes inteiras
optimizer	Adam(0.001)	Otimizador Adam com taxa de aprendizado de 0.001.
epochs	60	Número de épocas para treinamento do modelo.

FONTE: O Autor, (2024).

4 RESULTADOS

Nesta seção, são descritos os resultados obtidos derivados dos testes realizados com os modelos de aprendizado de máquina. Os resultados são organizados de acordo com cada modelo empregado no trabalho e leva em conta os objetivos específicos definidos no início do trabalho, de modo a nortear as questões de pesquisa e testar as hipóteses formuladas.

4.1 MODELO DE REGRESSÃO LOGÍSTICA

Um importante resultado que evidencia a eficiência de um modelo de AM é o score dele, que é medida em percentual e indica a proporção de acertos de previsão dos modelos com os dados empregados. Quando aplicamos a técnica do cálculo do score nos dados de treinamento, o resultado é a proporção de previsões corretas em relação ao total de previsões. Já quando aplicamos a técnica com os dados de teste, a saída fornece uma medida de quão bem o modelo generaliza para novos dados. A TABELA 8 demonstra o score obtido com os dados de treinamento e validação para o modelo de regressão logística.

TABELA 8 – SCORE DO MODELO DE REGRESSÃO LOGÍSTICA

Score (Dados de Treinamento)	0,599568849366747
Score (Dados de Validação)	0,600125746620559

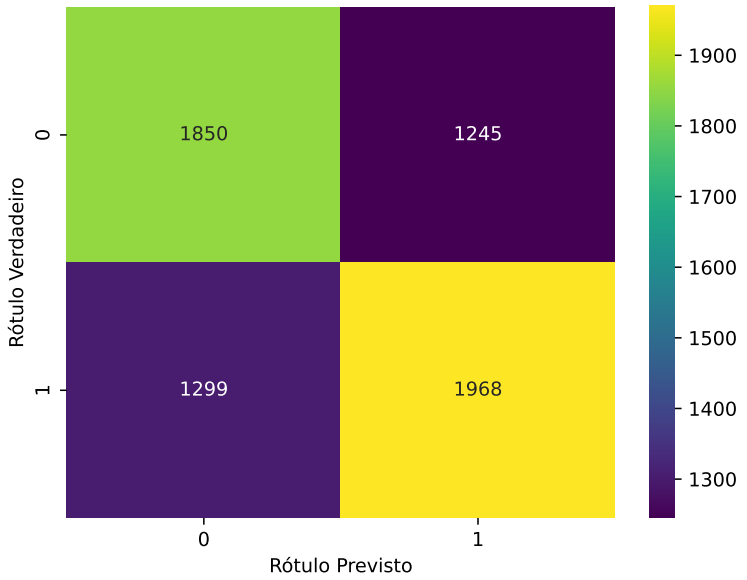
FONTE: O Autor, (2024).

Com base nos resultados até o momento, é evidente que o processo de aprendizado do modelo empregando regressão logística não está apresentando uma eficiência satisfatória. Tanto nos dados de treinamento quanto nos dados de teste do sistema, a acurácia atingiu cerca de 60%, um desempenho insuficiente para uma técnica de identificação com potencial de aplicação em outros sistemas elétricos. É interessante também destacar que, um modelo com saída binária aleatória teria uma acurácia de 50%, mostrando que o primeiro modelo está longe de ser eficaz e seus resultados revelam uma classificação quase aleatória.

Outra métrica importante para se utilizar nesses tipos de estudo, é a matriz de confusão. Uma matriz de confusão é uma tabela usada para avaliar o desempenho de um modelo de classificação. Ela compara as classificações feitas pelo modelo com as classificações reais dos dados e resume os resultados em uma estrutura que facilita a compreensão das métricas de avaliação, como precisão, recall, F1-score, entre outras. Uma matriz de confusão tem quatro quadrantes principais: Verdadeiros Positivos

(TP): Representa os casos em que o modelo classificou corretamente uma amostra como positiva; Falsos Positivos (FP): Indica os casos em que o modelo classificou erroneamente uma amostra como positiva quando na verdade era negativa; Falsos Negativos (FN): Mostra os casos em que o modelo classificou erroneamente uma amostra como negativa quando na verdade era positiva; Verdadeiros Negativos (TN): Refere-se aos casos em que o modelo classificou corretamente uma amostra como negativa. A matriz de confusão do modelo de regressão logística está disponível para visualização na FIGURA 12.

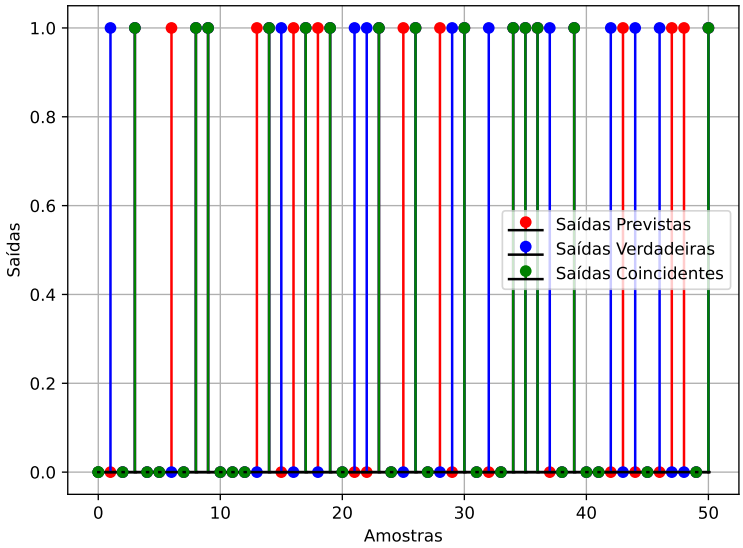
FIGURA 12 – MATRIZ DE CONFUSÃO DO MODELO DE REGRESSÃO LOGÍSTICA



FONTE: O Autor, (2024).

Outro resultado complementar para a análise da eficácia do modelo, está presente na FIGURA 13 que ilustra os resultados das classificações realizadas pelo modelo de regressão logística para um subconjunto de 50 amostras do conjunto de validação. Ela apresenta as saídas previstas pelo modelo em relação às saídas verdadeiras, destacando as amostras em que ambas coincidem. As saídas coincidentes são indicadas de maneira clara, demonstrando o alinhamento do modelo com os dados reais em muitos casos. Esse comportamento sugere que o modelo apresenta uma boa capacidade de generalização, especialmente em padrões comuns no conjunto de dados. No entanto, as discrepâncias evidenciadas reforçam a necessidade de ajustes adicionais ou talvez a integração de modelos mais complexos para abordar casos específicos.

FIGURA 13 – RESULTADOS DA CLASSIFICAÇÃO DO MODELO DE REGRESSÃO LOGÍSTICA



FONTE: O Autor, (2024).

4.2 MODELO DE REDE NEURAL MLP

Os resultados da utilização do modelo de rede neural MLP se mostraram relativamente discrepantes com os conseguidos no modelo anterior, considerando que ambos modelos foram empregados para o mesmo conjunto de dados, sendo eles os dados da simulação do evento de afundamento de tensão com o sistema dividido em duas áreas. A TABELA 9 e a FIGURA 14 demonstram graficamente os resultados obtidos com esse protótipo.

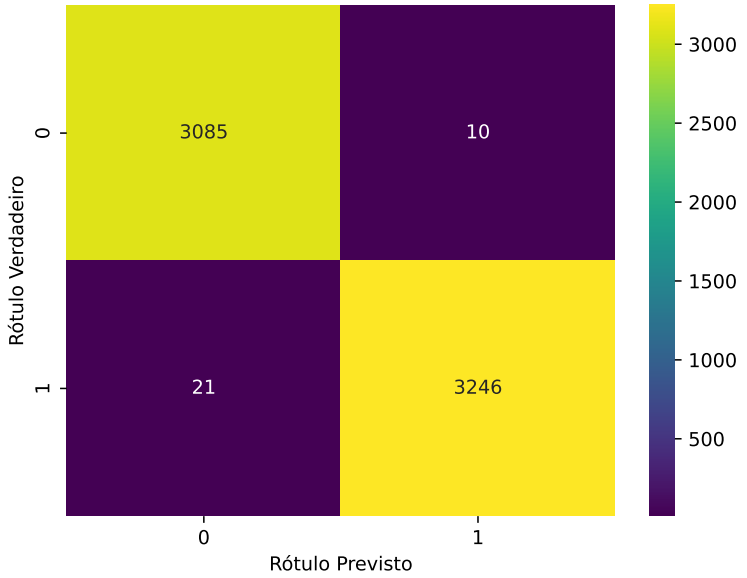
TABELA 9 – SCORE DO MODELO DE REDE NEURAL MLP

Score (Dados de Treinamento)	0,999191592562651
Score (Dados de Validação)	0,995756051556114

FONTE: O Autor, (2024).

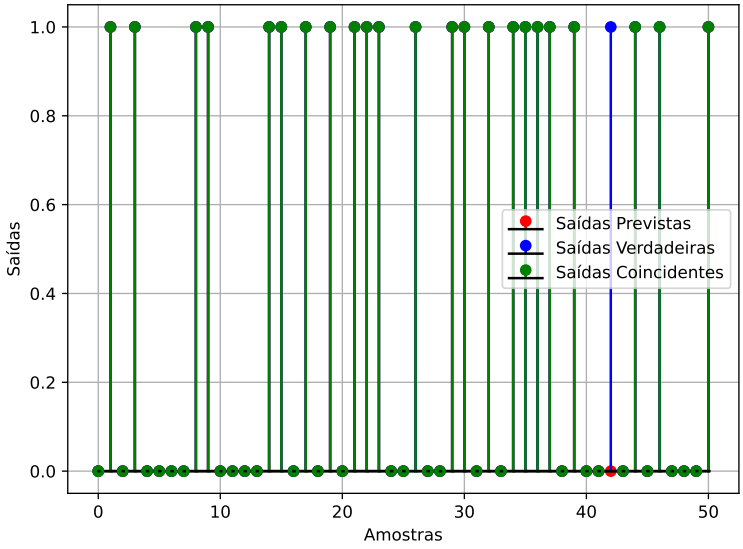
A FIGURA 15 apresenta os resultados das classificações feitas pelo modelo de rede neural para o subconjunto de 50 amostras dos dados de validação. Assim como no exemplo anterior, ele exibe as saídas previstas em comparação com as saídas verdadeiras, destacando os casos em que ambas coincidem. Nesse caso, observa-se que a rede neural conseguiu acertar um número significativo de classificações, com as saídas coincidentes sendo claramente representadas e tomando quase que a totalidade do espaço amostral, reforçando a eficiência do modelo empregado em relação ao anterior.

FIGURA 14 – MATRIZ DE CONFUSÃO DO MODELO DE REDE NEURAL MLP



FONTE: O Autor, (2024).

FIGURA 15 – RESULTADOS DA CLASSIFICAÇÃO DO MODELO DE REDE NEURAL MLP



FONTE: O Autor, (2024).

4.3 MODELO DE REDE NEURAL KERAS

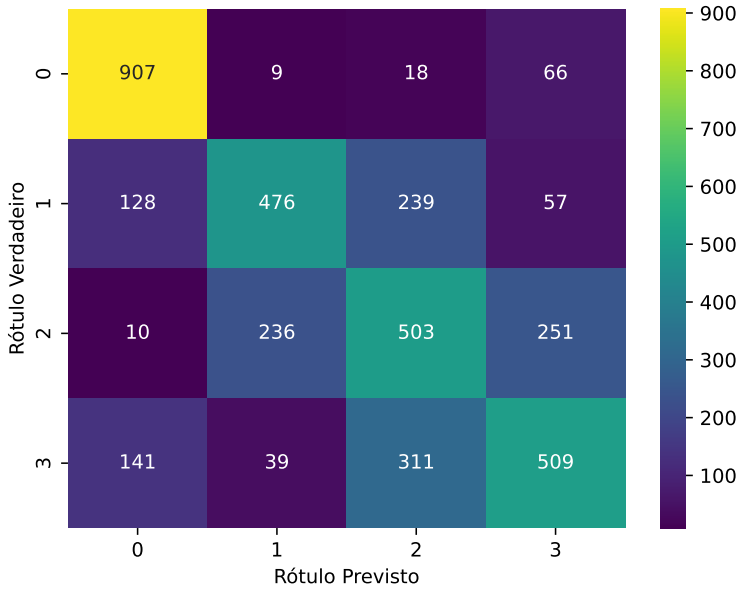
O primeiro modelo de rede neural Keras aplicado, conta com dados de todos os tipos de eventos estudados e a divisão do sistema em quatro regiões, o que é um fator complicador em protótipo de AM que busca criar relações entre os dados. Tendo isso em vista, a TABELA 10 e a FIGURA 16 representam os resultados obtidos nessa configuração.

TABELA 10 – SCORE DO MODELO DE REDE NEURAL KERAS

Score (Dados de Treinamento)	0,66400000000000
Score (Dados de Validação)	0,55051282051282

FONTE: O Autor, (2024).

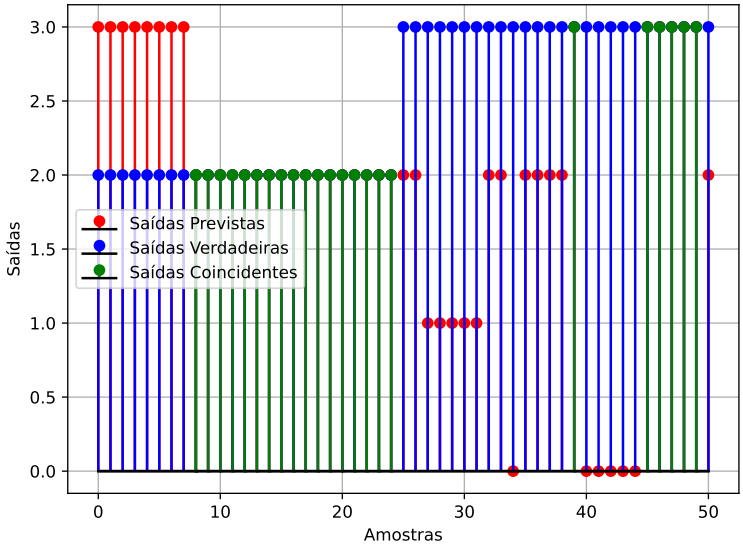
FIGURA 16 – MATRIZ DE CONFUSÃO DO MODELO DE REDE NEURAL KERAS



FONTE: O Autor, (2024).

Para esse modelo, também foram selecionadas amostras da classificação feita pelo protótipo de rede neural utilizando o Keras. A primeira e mais importante diferença que se nota com relação aos dados amostrais das outras classificações mostradas neste trabalho, e que nesse caso, existem 4 classes. O gráfico mostra os resultados da rede neural, evidenciando diferenças entre as saídas previstas e as saídas verdadeiras, o que indica precisão limitada no modelo atual. Embora algumas saídas coincidam, a inconsistência sugere a necessidade de ajustes. A classificação está disposta na FIGURA 17.

FIGURA 17 – RESULTADOS DA CLASSIFICAÇÃO DO MODELO DE REDE NEURAL KERAS



FONTE: O Autor, (2024).

4.4 RESULTADOS DOS TESTES DE APRIMORAMENTO DO MODELO

Os resultados das performances do modelo, feitas as devidas alterações de parâmetros, foram compilados em tabelas e estão dispostas a seguir. A TABELA 11 mostra os resultados dos testes com alteração no número de épocas de treinamento.

TABELA 11 – PERFORMANCE VARIANDO ÉPOCAS DE TREINAMENTO

Quantidade de Épocas	Score (Dados de Estimação)	Score (Dados de Validação)
20	0.7105925925925926	0.5656410256410257
30	0.7753333333333333	0.6189743589743589
40	0.7963703703703704	0.6515384615384615
50	0.802074074074074	0.6043589743589743
60	0.8594814814814815	0.6707692307692308
70	0.8381481481481482	0.6548717948717949
80	0.8688888888888889	0.6851282051282052
90	0.8907407407407407	0.6833333333333333
100	0.88	0.6607692307692308
110	0.8805185185185185	0.671025641025641

FONTE: O Autor, (2024).

A análise dos dados revela algumas tendências importantes sobre o desempenho da rede neural em função da quantidade de épocas de treinamento, entre elas, o fato de que entre 20 a 60 épocas, tanto o score dos dados de estimação quanto o score dos dados de validação aumentam progressivamente. Essa fase sugere que a rede neural está aprendendo com os dados de treinamento e generalizando relativamente bem para o conjunto de validação. A partir de 60 épocas, o score dos dados de

estimação continua a crescer, alcançando valores elevados como 0.868 nas 80 épocas e 0.890 nas 90 épocas. No entanto, o score nos dados de validação não aumenta proporcionalmente e, a partir de 70 épocas, começa a apresentar flutuações (ex. 0.654, 0.685, 0.683, e depois decai para 0.660). Esse comportamento indica que a rede neural pode estar sofrendo *overfitting*, onde o modelo ajusta-se muito bem aos dados de treinamento, mas perde capacidade de generalização para novos dados. De 90 a 110 épocas, o score nos dados de estimação permanece próximo de 0.88, enquanto o score de validação fica entre 0.660 e 0.671. Esse período sugere que o aumento de épocas não está mais proporcionando benefícios à generalização do modelo e pode estar inclusive prejudicando-a.

Continuando na análise dos resultados, a TABELA 12 mostra os resultados dos testes com alteração no número de neurônios na primeira camada da rede.

TABELA 12 – PERFORMANCE VARIANDO OS NEURÔNIOS NA CAMADA 1

Neurônios na Camada 1	Score (Dados de Estimação)	Score (Dados de Validação)
60	0.8694814814814815	0.6492307692307693
70	0.8768148148148148	0.6741025641025641
80	0.8914814814814814	0.7038461538461539
90	0.8848888888888888	0.6817948717948717
100	0.9038518518518519	0.6666666666666666
110	0.9262222222222222	0.6897435897435897
120	0.9072592592592592	0.6971794871794872
130	0.9167407407407407	0.6753846153846154
140	0.932074074074074	0.6879487179487179
150	0.9084444444444445	0.6720512820512821

FONTE: O Autor, (2024).

A análise dos resultados obtidos com a variação do número de neurônios na primeira camada indica que com o aumento de neurônios de 60 até 80, observa-se uma melhoria tanto no score de estimação (de 0.869 para 0.891) quanto no score de validação, que atinge seu ponto mais alto em 80 neurônios com um score de 0.703. Esse comportamento sugere que o aumento inicial no número de neurônios ajuda a rede neural a capturar melhor a complexidade dos dados, favorecendo a generalização. A partir de 90 neurônios, o score de estimação continua a crescer, alcançando valores elevados como 0.926 e 0.932, mas o score de validação não melhora de forma consistente e até começa a decair em alguns pontos (por exemplo, 0.666 com 100 neurônios e 0.675 com 130 neurônios). Esse padrão sugere que a rede neural pode estar começando a se ajustar excessivamente aos dados de treinamento (*overfitting*), especialmente a partir de 100 neurônios, o que reduz sua capacidade de generalização. Entre 110 e 150 neurônios, o score de validação apresenta uma oscilação, permanecendo em torno de valores como 0.689, 0.697, e 0.672. Isso indica que a adição de mais neurônios além de certo ponto não está mais trazendo benefícios

claros para a generalização, e o modelo parece estar próximo do limite ideal para o número de neurônios.

Com relação a mudança no número de neurônios na segunda camada, que pode ser visualizada na TABELA 13, vemos que ao aumentar os neurônios de 20 para 60, há uma melhoria significativa no score de validação, alcançando seu ponto mais alto com 60 neurônios (0.718). Nesse ponto, a rede neural parece ter um bom equilíbrio entre a complexidade do modelo e sua capacidade de generalização. A partir de 70 neurônios, o score de estimação continua a subir, indicando que o modelo está ficando mais preciso nos dados de treinamento (exemplo, 0.956 com 70 neurônios e 0.964 com 90 neurônios). No entanto, o score de validação não segue a mesma tendência e começa a oscilar (0.707 com 70 neurônios e depois 0.698 com 90 neurônios), sugerindo uma possível perda de generalização, característica de *overfitting*. Entre 60 e 110 neurônios, o score de validação permanece próximo, variando entre 0.690 e 0.718. Isso sugere que, apesar do aumento no número de neurônios, a rede neural não está melhorando sua performance em dados não vistos, sugerindo que o limite de neurônios que a segunda camada pode adicionar para melhorar a generalização está entre 20 e 60.

TABELA 13 – PERFORMANCE VARIANDO OS NEURÔNIOS NA CAMADA 2

Neurônios na Camada 2	Score (Dados de Estimação)	Score (Dados de Validação)
20	0.9096296296296297	0.7138461538461538
30	0.9228148148148149	0.6841025641025641
40	0.9391851851851852	0.6841025641025641
50	0.9472592592592592	0.6905128205128205
60	0.9317777777777778	0.7184615384615385
70	0.9560740740740741	0.7071794871794872
80	0.947037037037037	0.7076923076923077
90	0.9641481481481482	0.6987179487179487
100	0.9612592592592593	0.7120512820512821
110	0.9482962962962963	0.6938461538461539

FONTE: O Autor, (2024).

A tendência observada na variação do número de neurônios na terceira camada mostra que com o aumento dos neurônios de 10 para 40, o score de estimação melhora consistentemente, atingindo 0.973 com 40 neurônios. No entanto, o score de validação não apresenta uma tendência de melhora, variando entre 0.687 e 0.709 nesse intervalo, indicando uma possível falta de generalização. O melhor score de validação (0.721) ocorre com 70 neurônios na terceira camada. Nesse ponto, a rede neural parece alcançar o melhor equilíbrio entre aprendizado e generalização, sugerindo que essa configuração permite ao modelo capturar padrões dos dados de forma eficiente, sem ajustar-se demais aos dados de treinamento. Após 70 neurônios, o score de validação cai novamente e permanece próximo de 0.698, apesar de o score de estimação

continuar alto (por exemplo, 0.972 com 80 neurônios e 0.976 com 90 neurônios). Esse comportamento indica que adicionar mais neurônios à terceira camada após esse ponto pode aumentar o risco de *overfitting*, com o modelo se ajustando excessivamente aos dados de treinamento, mas sem ganho real em sua capacidade de generalização. Os dados desse comportamento podem ser verificados na TABELA 14

TABELA 14 – PERFORMANCE VARIANDO OS NEURÔNIOS NA CAMADA 3

Neurônios na Camada 3	Score (Dados de Estimação)	Score (Dados de Validação)
10	0.9482222222222222	0.7092307692307692
20	0.9628888888888889	0.6879487179487179
30	0.9476296296296296	0.6748717948717948
40	0.9732592592592593	0.698974358974359
50	0.9578518518518518	0.7048717948717949
60	0.9718518518518519	0.6920512820512821
70	0.9804444444444445	0.7217948717948718
80	0.9728148148148148	0.698974358974359
90	0.9769629629629629	0.6956410256410256
100	0.9445185185185185	0.6920512820512821

FONTE: O Autor, (2024).

A última iteração nessa rodada de testes, que consistiu na variação do termos de regularização das camadas, pode ser visto na TABELA 15 e mostra que sem regularização, o modelo alcança um alto score nos dados de estimação (0.971) e um score de validação razoável (0.696), mas a diferença sugere *overfitting*. Isso ocorre porque o modelo se ajusta aos dados de treinamento sem restrições, o que o torna mais suscetível a memorizar esses dados sem generalizar bem. Com uma pequena quantidade de regularização, o score de estimação diminui para 0.917 e o score de validação para 0.688. Essa configuração ainda permite um bom ajuste aos dados de treinamento, mas com uma leve redução no risco de *overfitting*. Ao aumentar λ para 0.01, há uma queda significativa no score de estimação para 0.640 e no score de validação para 0.552. Esse valor de regularização parece ser elevado o bastante para restringir significativamente a capacidade de ajuste do modelo, o que resulta em uma performance mais limitada em ambos os conjuntos de dados. Com valores de λ de 0.05 ou mais, o modelo alcança scores muito baixos e constantes (0.281 para estimação e 0.256 para validação), indicando que a regularização excessiva está restringindo o modelo a ponto de ele se tornar incapaz de capturar os padrões dos dados. Isso causa *underfitting*, onde o modelo se torna muito simples para representar a complexidade dos dados.

TABELA 15 – PERFORMANCE VARIANDO O PARÂMETRO DE REGULARIZAÇÃO

Lambda	Score (Dados de Estimação)	Score (Dados de Validação)
0.0	0.9712592592592593	0.6961538461538461
0.001	0.9178518518518518	0.6879487179487179
0.01	0.6398518518518519	0.5523076923076923
0.05	0.2814814814814815	0.2564102564102564
0.1	0.2814814814814815	0.2564102564102564
0.2	0.2814814814814815	0.2564102564102564
0.3	0.2814814814814815	0.2564102564102564
0.45	0.2814814814814815	0.2564102564102564
90	0.9769629629629629	0.6956410256410256
100	0.9445185185185185	0.6920512820512821

FONTE: O Autor, (2024).

Os resultados da otimização com Optuna mostram um score de treinamento alto de 0.944 e um score de validação de 0.688, que estão dispostos na TABELA 16. O score de 0.944 nos dados de treinamento mostra que o modelo conseguiu capturar bem os padrões do conjunto de treinamento. Isso pode ter sido resultado de uma alta complexidade do modelo, como um número elevado de neurônios, camadas, ou a ausência de técnicas de regularização adequadas. O score de validação de 0.688, muito inferior ao score de treinamento, sugere que o modelo não generaliza bem para novos dados. Esse problema ocorre frequentemente quando o modelo é ajustado para os dados de treinamento a ponto de perder sua capacidade de ser flexível o suficiente para padrões ligeiramente diferentes nos dados de validação. No treinamento de redes neurais, as métricas auxiliares perda por época e acerto por época ajudam a monitorar o desempenho do modelo ao longo do tempo. A perda por época mede o erro do modelo em cada época e representa o quão distante as previsões estão dos valores reais. O objetivo é minimizar essa perda ao longo do treinamento. O acerto por época mede a precisão do modelo, ou seja, a porcentagem de previsões corretas em cada época e indica como o modelo está se aproximando do resultado esperado. Para cada teste feito com o Optuna, essas métricas foram calculadas e estão disponíveis no Apêndice 5 e Apêndice 6 respectivamente.

TABELA 16 – SCORE DO MODELO DE REDE NEURAL FINAL

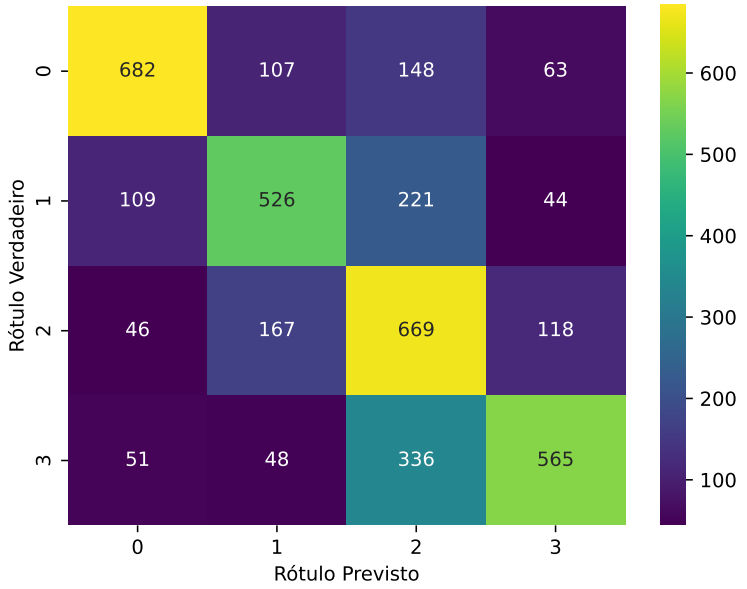
Score (Dados de Treinamento)	0.9438518518518518
Score (Dados de Validação)	0.6884615384615385

FONTE: O Autor, (2024).

A matriz de confusão para o modelo de rede neural final conseguido pelo Optuna está disponível na FIGURA 18 e indica que o modelo possui um bom número de classificações corretas, mas apresenta uma confusão significativa entre algumas

classes, especialmente entre as classes 0 e 2, classes 1 e 2, e classes 2 e 3. Esses erros podem ser decorrentes de similaridades entre características das classes, levando o modelo a confundir uma classe com outra, particularmente quando as classes têm padrões sobrepostos. Além disso, a diferença entre os scores de treinamento e validação sugere *overfitting*, o que prejudica a generalização do modelo e pode contribuir para essas confusões. Para melhorar, seria útil considerar técnicas de regularização mais fortes, simplificar a arquitetura ou ajustar o pré-processamento para enfatizar diferenças entre classes e, possivelmente, ponderar algumas classes para que o modelo aprenda a distingui-las com maior precisão.

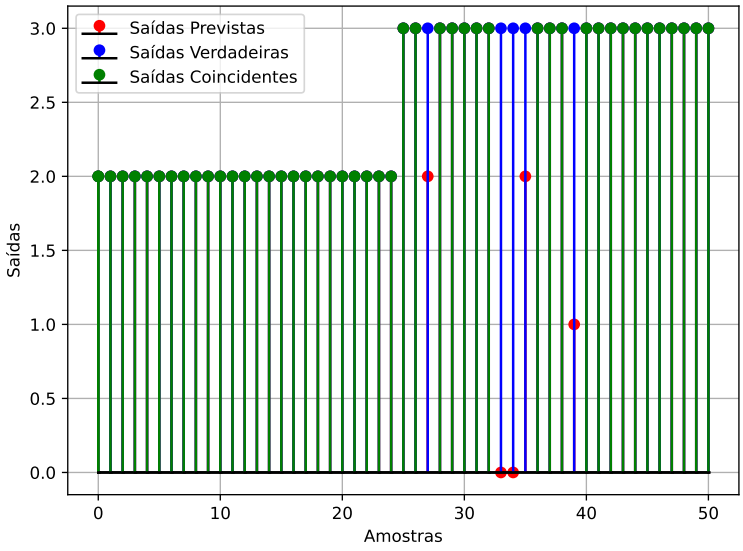
FIGURA 18 – MATRIZ DE CONFUSÃO DO MODELO DE REDE NEURAL FINAL



FONTE: O Autor, (2024).

Por fim, assim como todos os outros modelos estudados nesse trabalho, foram selecionadas amostras do processo de classificação dos dados de validação, a fim de comparação com os outros modelos e análise de eficiência. Os resultados da classificação podem ser visualizados na FIGURA 19. Comparando o gráfico deste modelo de rede neural com o primeiro que analisa dados em 4 áreas, percebe-se uma melhora significativa na precisão da classificação. Enquanto no primeiro modelo havia discrepâncias visíveis entre as saídas previstas e as verdadeiras, no segundo modelo essas diferenças são muito menores, com uma maior quantidade de saídas coincidentes. Isso indica que o novo modelo consegue aprender melhor os padrões presentes nos dados de treinamento. A amplitude dos erros diminuiu, sugerindo maior estabilidade e consistência no desempenho.

FIGURA 19 – RESULTADOS DA CLASSIFICAÇÃO DO MODELO DE REDE NEURAL FINAL



FONTE: O Autor, (2024).

5 CONCLUSÕES E TRABALHOS FUTUROS

O desenvolvimento de modelos de aprendizado de máquina (AM) para a detecção de falhas em sistemas elétricos é uma área de estudo extremamente relevante, especialmente no contexto atual, em que a complexidade das redes de energia está em constante crescimento. A identificação rápida e precisa de áreas onde ocorrem falhas é crucial para garantir a eficiência operacional, reduzir o tempo de resposta e melhorar a segurança das redes. Este trabalho complemento a bibliografia técnica já existente sobre o ramo da inteligência artificial ao demonstrar como diferentes abordagens de aprendizado de máquina podem ser aplicadas para resolver um problema crítico na operação de sistemas elétricos, reforçando a importância da inteligência artificial na modernização desse setor.

Os resultados obtidos evidenciam os desafios e as potencialidades das técnicas utilizadas. O primeiro modelo, baseado em regressão logística e aplicado a um sistema dividido em duas áreas, apresentou uma precisão de 59,95% nos dados de treinamento e 60,01% nos dados de validação. Esses valores mostram que, embora a regressão logística seja uma técnica robusta para problemas lineares, ela se mostrou limitada para a complexidade deste problema específico, onde a distinção entre as áreas de falha exige uma modelagem mais sofisticada.

Por outro lado, o segundo modelo, que utilizou uma rede neural para a mesma divisão em duas áreas, apresentou resultados notáveis, com 99,91% de precisão nos dados de treinamento e 99,57% nos dados de validação. Esses números demonstram a superioridade das redes neurais na captura de padrões complexos presentes nos dados, destacando sua capacidade de generalização quando a arquitetura é bem projetada e os dados são adequados. Esse alto nível de precisão reforça o potencial das redes neurais em aplicações críticas, como a detecção de falhas em sistemas elétricos.

A transição para um cenário mais complexo, onde o sistema foi dividido em quatro áreas, apresentou novos desafios. O terceiro modelo, também baseado em rede neural, alcançou uma precisão de 66,4% nos dados de treinamento e apenas 55,41% nos dados de validação. Esses resultados indicam dificuldades de generalização, sugerindo que o modelo pode ter superajustado os dados de treinamento ou que a complexidade adicional da tarefa requer ajustes mais refinados na arquitetura ou no pré-processamento dos dados.

O quarto modelo, uma versão aprimorada do terceiro, apresentou uma melhora significativa, com 94,38% de precisão nos dados de treinamento e 68,84% nos dados

de validação. Embora a precisão nos dados de validação ainda possa ser considerada modesta, a evolução em relação ao modelo anterior demonstra que as melhorias aplicadas, possivelmente envolvendo ajustes nos hiperparâmetros ou na estrutura da rede neural, tiveram impacto positivo. Esse avanço mostra que há espaço para otimizações adicionais, sugerindo caminhos promissores para futuros estudos.

As perspectivas para trabalhos futuros são amplas e variadas. Uma das direções possíveis é a otimização dos hiperparâmetros dos modelos, utilizando métodos como *grid search* ou *bayesian optimization* para explorar combinações que possam melhorar a generalização. Além disso, a coleta e a utilização de conjuntos de dados mais diversificados e equilibrados, contendo diferentes tipos de falhas e condições operacionais, podem aumentar a robustez e a capacidade de adaptação dos modelos.

Outra possibilidade é a exploração de arquiteturas mais avançadas, como redes neurais convolucionais (CNNs) e redes recorrentes (RNNs), que poderiam capturar de maneira mais eficiente as características espaciais e temporais das falhas nos sistemas elétricos. A implementação em tempo real dos modelos também se apresenta como um desafio interessante, possibilitando testes em ambientes reais para avaliar seu desempenho operacional e o impacto na redução do tempo de resposta a falhas. Finalmente, a integração dos modelos com sistemas automatizados de controle e proteção pode potencializar ainda mais a eficiência das redes elétricas, tornando possível uma resposta automatizada e precisa diante de eventos críticos.

Em resumo, este trabalho não apenas demonstrou a eficácia de modelos de aprendizado de máquina na detecção de áreas de falhas em sistemas elétricos, mas também abriu portas para novas pesquisas e aplicações. Os resultados obtidos fornecem uma base sólida para futuras investigações e destacam o potencial da inteligência artificial como uma ferramenta indispensável para a modernização e melhoria contínua dos sistemas elétricos.

REFERÊNCIAS

ALMOBASHER, L. R.; HABIBALLAH, I. O. A. Review of Power System Faults. **International Journal of Engineering Research Technology (IJERT)**, 2020. Citado 2 vezes nas páginas 25, 27.

ATHAY, T.; PODMORE, R.; VIRMANI, S. A Practical Method for the Direct Analysis of Transient Stability. **IEEE Transactions on Power Apparatus and Systems**, PAS-98, n. 2, p. 573–584, 1979. DOI: 10.1109/TPAS.1979.319407. Citado 1 vez na página 35.

BOYLESTAD, R. L. **Introdução a Análise de Circuitos**. 12. ed. São Paulo: Pearson, 2012. Citado 1 vez na página 21.

BUNNOON, P. Fault Detection Approaches to Power System: State-of-the-Art Article Reviews for Searching a New Approach in the Future. **International Journal of Electrical and Computer Engineering (IJECE)**, 2013. Citado 1 vez na página 17.

CANIZARES, C.; FERNANDES, T.; GERALDI, E.; GERIN-LAJOIE, L.; GIBBARD, M.; HISKENS, I.; KERSULIS, J.; KUIAVA, R.; LIMA, L.; DEMARCO, F.; MARTINS, N.; PAL, B. C.; PIARDI, A.; RAMOS, R.; SANTOS, J. dos; SILVA, D.; SINGH, A. K.; TAMIMI, B.; VOWLES, D. Benchmark Models for the Analysis and Control of Small-Signal Oscillatory Dynamics in Power Systems. **IEEE Transactions on Power Systems**, v. 32, n. 1, p. 715–722, 2017. DOI: 10.1109/TPWRS.2016.2561263. Citado 1 vez na página 36.

CARUANA, R. Multitask learning. **Machine learning**, Springer, v. 28, p. 41–75, 1997. Citado 1 vez na página 29.

CEPEL. **Manual ANATEM**. [S.l.], 2024. Disponível em: https://see.cepel.br/manual/anatem/index.html. Citado 1 vez na página 39.

DUSABIMANA, E.; YOON, S.-G. A survey on the micro-phasor measurement unit in distribution networks. **Electronics**, MDPI, v. 9, n. 2, p. 305, 2020. Citado 1 vez na página 50.

HASAN, A. N.; EBOULE, P. P.; TWALA, B. The use of machine learning techniques to classify power transmission line fault types and locations. **2017 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM) 2017**

Intl Aegean Conference on Electrical Machines and Power Electronics (ACEMP), p. 221–226, 2017. DOI: 10.1109/OPTIM.2017.7974974. Citado 1 vez na página 18.

IBGE. Características gerais dos domicílios e dos moradores 2022, 2022. Disponível em: https://agenciadenoticias.ibge.gov.br/media/com_mediaibge/arquivos/1cd893a10b3cabf31fc31e994 Citado 1 vez na página 16.

IEA; IRENA; UNSD; BANK, W.; WHO. Tracking SDG 7: The Energy Progress Report, 2024. Disponível em: https://trackingsdg7.esmap.org/data/files/download-documents/sdg7-report2024-0611-v9-highresforweb.pdf. Citado 1 vez na página 16.

JAMIL, M.; SHARMA, S. K.; SINGH, R. Fault detection and classification in electrical power transmission system using artificial neural network. **SpringerPlus**, 2015. Citado 1 vez na página 17.

MAHESH, B. Machine Learning Algorithms - A Review. **International Journal of Science and Research (IJSR)**, 2020. Citado 1 vez na página 28.

MONTICELLI, A.; GARCIA, A. **Introdução a Sistemas de Energia Elétrica**. 2. ed. São Paulo: Unicamp, 2011. Citado 1 vezes nas páginas 21, 25.

OLIVEIRA, S. E. M.; RANGEL, R. D.; THOMÉA, L. M.; BAITELLI, R.; GUIMARÃES, C. H. Programa ANATEM para Simulação do Desempenho Dinâmico dos Sistemas Elétricos de Potência. **IV Symposium of Specialista in Electric Operational and Expansion Planning**, 1994. Citado 1 vez na página 33.

SHINDE, P. P.; SHAH, S. A Review of Machine Learning and Deep Learning Applications. **2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)**, p. 1–6, 2018. DOI: 10.1109/ICCUBEA.2018.8697857. Citado 1 vez na página 28.

VAISH, R.; DWIVEDI, U.; TEWARI, S.; TRIPATHI, S. Machine learning applications in power system fault diagnosis: Research advancements and perspectives. **Engineering Applications of Artificial Intelligence**, 2021. Citado 1 vez na página 17.

VAN ENGELEN, J. E.; HOOS, H. H. A survey on semi-supervised learning. **Machine learning**, Springer, v. 109, n. 2, p. 373–440, 2020. Citado 1 vez na página 28.

WANG, F.; LI, Y.; ZHANG, Y. An empirical study on the search engine optimization technique and its outcomes, p. 2767–2770, 2011. Citado 1 vez na página 27.

YADAV, A.; DASH, Y. An Overview of Transmission Line Protection by Artificial Neural Network: Fault Detection, Fault Classification, Fault Location, and Fault Direction Discrimination. **Advances in Artificial Neural Systems**, 2014. Citado 1 vez na página 26.

APÊNDICE 1 – CÓDIGO DE MODELAGEM DO SISTEMA (ANATEM)

```
 CEPEL - Centro de Pesquisas de Energia Elétrica
Análise de Transitórios Eletromecânicos - ANATEM - Versão  12.0.0-x64

X-------------------------------------------------X
| INFORMAÇÕES SOBRE CÓPIA DO PROGRAMA             |
X-------------------------------------------------X
VERSÃO : 12.0.0-Fev21
LICENÇA: Academica
EMPRESA: Universidade Federal do Paraná
SIGLA  : UFPR

DADOS DE CONSTANTES

X----X------X
|Nome  Valor|
X----X------X

TEPQ    .01
TEMD  1.E-6
TETE  1.E-6
TABS  1.E-6

SUMÁRIO DE CONSTANTES

X-------X-------X-------X-------X-------X-------X-------X-------X
TETE    TEMD    TEPQ    TABS    TBID    PFFB    AMX1    AMX2
X-------X-------X-------X-------X-------X-------X-------X-------X

.1E-5   .1E-5     .01   .1E-5   .1E-8     .02    360.   1000.


X------X------X------X------X------X------X------X------X
LCRT   LPRT   IMDS   IACS   IACE   MRAC   MRDC   ITMR
X------X------X------X------X------X------X------X------X

24     60     10     10    300     30    100     20
```

```
X-------X---------X-------------------------------X
|UNIDADE    NOME           IDENTIFICAÇÃO          |
| LOGICA   LOGICO           DO ARQUIVO            |
X-------X---------X-------------------------------X

1   TEM$DADOS .\39bus_ANATEM_remocao_gerador_barra32.stb
2   TEM$SAVCA .\39BUS_SYSTEM_CARREGAMENTO_CASO_BASE.SAV
3   TEM$MODEL .\DATA_10GENERATOR.BLT
4   TEM$PRINT .\39BUS.OUT
5   TEM$INPUT ..\..\..\..\..\..\..\..\\\.\CONIN$
6   TEM$VIDEO ..\..\..\..\..\..\..\..\\\.\CONOUT$
7   TEM$PUNCH NUL
8   TEM$PLOTA .\39BUS.PLT
9   TEM$LOG   NUL
10   TEM$SNAPS NUL
11   TEM$ARQSN NUL

DADOS DE MODELO 2 DE GERADOR - POLOS SALIENTES

X----X----X-----X-----X-----X-----X-----X-----X-----X-----X-X
| MOD  SAT   Ld   L'd   L"d    Ll   T'do  T"do   Ra    H    MVA  C|
|            Lq                          T"qo         D    Freq  |
X----X----X-----X-----X-----X-----X-----X-----X-----X-----X-X

0001        2.0   0.6   .40   0.3   7.0   .05     0   500   100 N
1.9                          .035           0   60
0002       29.5  6.97  5.00   3.5  6.56   .05     0  30.3   100 N
28.2                         .035           0   60
0003      24.95  5.31  4.50  3.04   5.7   .05     0  35.8   100 N
23.7                         .035           0   60
0004       26.2  4.36  3.50  2.95  5.69   .05     0  28.6   100 N
25.8                         .035           0   60
0005         67  13.2  8.90   5.4   5.4   .05     0    26   100 N
62                          .035           0   60
0006       25.4   5.0  4.00  2.24   7.3  .050     0  34.8   100 N
24.1                         .035           0   60
0007       29.5   4.9  4.40  3.22  5.66  .050     0  26.4   100 N
```

```
29.2                          .035           0   60
0008         29   5.7  4.50  2.80   6.7  .050     0  24.3   100 N
28                          .035           0   60
0009      21.06   5.7  4.50  2.98  4.79  .050     0  34.5   100 N
20.5                          .035           0   60
0010         10   3.1  2.50  1.25  10.2  .050     0  42.0   100 N
6.9                           .035           0   60


X-------X---------X------------------------------X
|UNIDADE    NOME           IDENTIFICAÇÃO          |
| LOGICA   LOGICO           DO ARQUIVO            |
X-------X---------X------------------------------X

1   TEM$DADOS .\39bus_ANATEM_remocao_gerador_barra32.stb
2   TEM$SAVCA .\39BUS_SYSTEM_CARREGAMENTO_CASO_BASE.SAV
3   TEM$MODEL .\DATA_10GENERATOR_AVR_AND_PSS.CDU
4   TEM$PRINT .\39BUS.OUT
5   TEM$INPUT ..\..\..\..\..\..\..\..\..\\\.\CONIN$
6   TEM$VIDEO ..\..\..\..\..\..\..\..\..\\\.\CONOUT$
7   TEM$PUNCH NUL
8   TEM$PLOTA .\39BUS.PLT
9   TEM$LOG   NUL
10   TEM$SNAPS NUL
11   TEM$ARQSN NUL

DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X---------------X---------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1001 AVR_G1

DEFPAR                #TR                   0.01
DEFPAR                #TC                    1.0
DEFPAR                #TB                   10.0
DEFPAR                #KA                  200.0
```

```
DEFPAR                  #TA                     0.015
0001   ENTRAD                   VREF

0002   IMPORT VTR               VT

0003   IMPORT VSAD              VPSS

0010   LEDLAG           VT      X1         1.0    0.0    1.0 #TR

0020   SOMA           - X1      X2

VREF   X2

VPSS   X2

0030   LEDLAG           X2      X3         1.0 #TC        1.0 #TB

0040   LEDLAG           X3      X4      #KA        0.0    1.0 #TA

0050   LIMITA           X4      EFD                                VEMIN
VEMAX
0100   EXPORT EFD       EFD

DEFVAL                  VEMIN    -5.0

DEFVAL                  VEMAX     5.0

.....................................................................................
1002 AVR_G2

DEFPAR                  #TR                     0.01
DEFPAR                  #TC                      1.0
DEFPAR                  #TB                     10.0
DEFPAR                  #KA                    200.0
DEFPAR                  #TA                     0.015
0001   ENTRAD                   VREF

0002   IMPORT VTR               VT
```

```
0003   IMPORT VSAD            VPSS

0010   LEDLAG          VT     X1        1.0    0.0    1.0 #TR


DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X--------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1002 AVR_G2

0020   SOMA          - X1     X2

VREF   X2

VPSS   X2

0030   LEDLAG          X2     X3        1.0 #TC       1.0 #TB

0040   LEDLAG          X3     X4     #KA       0.0    1.0 #TA

0050   LIMITA          X4     EFD                                VEMIN
VEMAX
0100   EXPORT EFD      EFD

DEFVAL                 VEMIN    -5.0

DEFVAL                 VEMAX     5.0

.....................................................................................
1003 AVR_G3

DEFPAR                 #TR                    0.01
DEFPAR                 #TC                     1.0
DEFPAR                 #TB                    10.0
DEFPAR                 #KA                   200.0
```

```
DEFPAR                  #TA                     0.015
0001   ENTRAD                   VREF

0002   IMPORT VTR               VT

0003   IMPORT VSAD              VPSS

0010   LEDLAG           VT      X1         1.0     0.0     1.0 #TR

0020   SOMA           - X1      X2

VREF   X2

VPSS   X2

0030   LEDLAG           X2      X3         1.0 #TC         1.0 #TB

0040   LEDLAG           X3      X4      #KA        0.0     1.0 #TA

0050   LIMITA           X4      EFD                                VEMIN
VEMAX

DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X--------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1003 AVR_G3

0100   EXPORT EFD       EFD

DEFVAL                  VEMIN     -5.0

DEFVAL                  VEMAX      5.0

.....................................................................................
1004 AVR_G4
```

```
DEFPAR                  #TR                     0.01
DEFPAR                  #TC                      1.0
DEFPAR                  #TB                     10.0
DEFPAR                  #KA                    200.0
DEFPAR                  #TA                    0.015
0001   ENTRAD                   VREF

0002   IMPORT VTR               VT

0003   IMPORT VSAD              VPSS

0010   LEDLAG           VT      X1         1.0    0.0    1.0 #TR

0020   SOMA           - X1      X2

VREF   X2

VPSS   X2

0030   LEDLAG           X2      X3         1.0 #TC        1.0 #TB

0040   LEDLAG           X3      X4      #KA        0.0    1.0 #TA

0050   LIMITA           X4      EFD                                VEMIN
VEMAX
0100   EXPORT EFD       EFD

DEFVAL                  VEMIN    -5.0

DEFVAL                  VEMAX     5.0

.....................................................................................
1005 AVR_G5

DEFPAR                  #TR                     0.01
DEFPAR                  #TC                      1.0
DEFPAR                  #TB                     10.0
```

```
DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X---------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1005 AVR_G5

DEFPAR                 #KA                   200.0
DEFPAR                 #TA                   0.015
0001   ENTRAD                VREF

0002   IMPORT VTR            VT

0003   IMPORT VSAD           VPSS

0010   LEDLAG          VT    X1        1.0    0.0    1.0 #TR

0020   SOMA          - X1    X2

VREF   X2

VPSS   X2

0030   LEDLAG          X2    X3        1.0 #TC       1.0 #TB

0040   LEDLAG          X3    X4     #KA       0.0    1.0 #TA

0050   LIMITA          X4    EFD                              VEMIN
VEMAX
0100   EXPORT EFD      EFD

DEFVAL                 VEMIN    -5.0

DEFVAL                 VEMAX     5.0

.....................................................................................
1006 AVR_G6
```

```
DEFPAR                  #TR                     0.01
DEFPAR                  #TC                      1.0
DEFPAR                  #TB                     10.0
DEFPAR                  #KA                    200.0
DEFPAR                  #TA                    0.015
0001   ENTRAD                  VREF

0002   IMPORT VTR              VT

0003   IMPORT VSAD             VPSS

0010   LEDLAG           VT     X1        1.0    0.0    1.0 #TR

0020   SOMA           - X1     X2


DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X---------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1006 AVR_G6

VREF   X2

VPSS   X2

0030   LEDLAG           X2     X3        1.0 #TC       1.0 #TB

0040   LEDLAG           X3     X4     #KA       0.0    1.0 #TA

0050   LIMITA           X4     EFD                               VEMIN
VEMAX
0100   EXPORT EFD       EFD

DEFVAL                  VEMIN    -5.0
```

```
DEFVAL                  VEMAX     5.0

.....................................................................
1007 AVR_G7

DEFPAR                  #TR                     0.01
DEFPAR                  #TC                      1.0
DEFPAR                  #TB                     10.0
DEFPAR                  #KA                    200.0
DEFPAR                  #TA                    0.015
0001   ENTRAD                   VREF

0002   IMPORT VTR               VT

0003   IMPORT VSAD              VPSS

0010   LEDLAG           VT      X1        1.0    0.0    1.0 #TR

0020   SOMA           - X1      X2

VREF   X2

VPSS   X2

0030   LEDLAG           X2      X3        1.0 #TC       1.0 #TB

0040   LEDLAG           X3      X4      #KA       0.0    1.0 #TA

0050   LIMITA           X4      EFD                                VEMIN
VEMAX
0100   EXPORT EFD       EFD

DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X---------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
```

```
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1007 AVR_G7

DEFVAL                 VEMIN    -5.0

DEFVAL                 VEMAX     5.0

.....................................................................................
1008 AVR_G8

DEFPAR                 #TR                    0.01
DEFPAR                 #TC                     1.0
DEFPAR                 #TB                    10.0
DEFPAR                 #KA                   200.0
DEFPAR                 #TA                   0.015
0001   ENTRAD                 VREF

0002   IMPORT VTR             VT

0003   IMPORT VSAD            VPSS

0010   LEDLAG          VT     X1        1.0    0.0    1.0 #TR

0020   SOMA          - X1     X2

VREF   X2

VPSS   X2

0030   LEDLAG          X2     X3        1.0 #TC       1.0 #TB

0040   LEDLAG          X3     X4     #KA       0.0    1.0 #TA

0050   LIMITA          X4     EFD                              VEMIN
VEMAX
0100   EXPORT EFD      EFD

DEFVAL                 VEMIN    -5.0
```

```
DEFVAL                  VEMAX     5.0

.....................................................................
1009 AVR_G9

DEFPAR                  #TR                    0.01
DEFPAR                  #TC                     1.0
DEFPAR                  #TB                    10.0
DEFPAR                  #KA                   200.0
DEFPAR                  #TA                   0.015

DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X---------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1009 AVR_G9

0001   ENTRAD                 VREF

0002   IMPORT VTR             VT

0003   IMPORT VSAD            VPSS

0010   LEDLAG          VT     X1        1.0    0.0    1.0 #TR

0020   SOMA          - X1     X2

VREF   X2

VPSS   X2

0030   LEDLAG          X2     X3        1.0 #TC       1.0 #TB

0040   LEDLAG          X3     X4     #KA       0.0    1.0 #TA
```

```
0050   LIMITA          X4     EFD                                VEMIN
VEMAX
0100   EXPORT EFD      EFD

DEFVAL                 VEMIN    -5.0

DEFVAL                 VEMAX     5.0

.....................................................................
1010 AVR_G10

DEFPAR                 #TR                    0.01
DEFPAR                 #TC                     1.0
DEFPAR                 #TB                    10.0
DEFPAR                 #KA                   200.0
DEFPAR                 #TA                   0.015
0001   ENTRAD                 VREF

0002   IMPORT VTR             VT

0003   IMPORT VSAD            VPSS

0010   LEDLAG          VT     X1        1.0    0.0    1.0 #TR

0020   SOMA          - X1     X2

VREF   X2


DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X--------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1010 AVR_G10

VPSS   X2
```

```
0030   LEDLAG          X2     X3        1.0 #TC       1.0 #TB

0040   LEDLAG          X3     X4     #KA       0.0    1.0 #TA

0050   LIMITA          X4     EFD                              VEMIN
VEMAX
0100   EXPORT EFD      EFD

DEFVAL                 VEMIN    -5.0

DEFVAL                 VEMAX     5.0

.....................................................................
1100 PSS_G1

DEFPAR                 #K                    1.0
DEFPAR                 #TW                  10.0
DEFPAR                 #T1                   5.0
DEFPAR                 #T2                  0.60
DEFPAR                 #T3                   3.0
DEFPAR                 #T4                  0.50
0001   IMPORT DWMAQ           DW

0020   GANHO           DW     X1     #K

0030   WSHOUT          X1     X2     #TW       1.0 #TW

0040   LEDLAG          X2     X3        1.0 #T1       1.0 #T2

0050   LEDLAG          X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA          X4     VSAD                             VPMIN
VPMAX
0100   EXPORT VSAD     VSAD

DEFVAL                 VPMIN    -0.2

DEFVAL                 VPMAX     0.2
```

```
.....................................................................
1200 PSS_G2

DEFPAR                  #K                       0.5
DEFPAR                  #TW                     10.0

DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X---------------X---------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1200 PSS_G2

DEFPAR                  #T1                      5.0
DEFPAR                  #T2                     0.40
DEFPAR                  #T3                      1.0
DEFPAR                  #T4                     0.10
0001   IMPORT DWMAQ           DW

0020   GANHO           DW     X1     #K

0030   WSHOUT          X1     X2     #TW       1.0 #TW

0040   LEDLAG          X2     X3        1.0 #T1       1.0 #T2

0050   LEDLAG          X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA          X4     VSAD                              VPMIN
VPMAX
0100   EXPORT VSAD     VSAD

DEFVAL                  VPMIN    -0.2

DEFVAL                  VPMAX     0.2

.....................................................................
```

```
1300 PSS_G3

DEFPAR                  #K                       0.5
DEFPAR                  #TW                     10.0
DEFPAR                  #T1                      3.0
DEFPAR                  #T2                     0.20
DEFPAR                  #T3                      2.0
DEFPAR                  #T4                     0.20
0001   IMPORT DWMAQ           DW

0020   GANHO            DW     X1     #K

0030   WSHOUT           X1     X2     #TW       1.0 #TW

0040   LEDLAG           X2     X3        1.0 #T1       1.0 #T2

0050   LEDLAG           X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA           X4     VSAD                                VPMIN
VPMAX
0100   EXPORT VSAD      VSAD

DEFVAL                  VPMIN    -0.2


DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X--------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1300 PSS_G3

DEFVAL                  VPMAX     0.2

.................................................................................
1400 PSS_G4
```

```
DEFPAR                #K                      2.0
DEFPAR                #TW                    10.0
DEFPAR                #T1                     1.0
DEFPAR                #T2                    0.10
DEFPAR                #T3                     1.0
DEFPAR                #T4                    0.30
0001   IMPORT DWMAQ          DW

0020   GANHO          DW     X1     #K

0030   WSHOUT         X1     X2     #TW       1.0 #TW

0040   LEDLAG         X2     X3        1.0 #T1       1.0 #T2

0050   LEDLAG         X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA         X4     VSAD                            VPMIN
VPMAX
0100   EXPORT VSAD    VSAD

DEFVAL                VPMIN    -0.2

DEFVAL                VPMAX     0.2

.....................................................................................
1500 PSS_G5

DEFPAR                #K                      1.0
DEFPAR                #TW                    10.0
DEFPAR                #T1                     1.5
DEFPAR                #T2                    0.20
DEFPAR                #T3                     1.0
DEFPAR                #T4                    0.10
0001   IMPORT DWMAQ          DW

0020   GANHO          DW     X1     #K

0030   WSHOUT         X1     X2     #TW       1.0 #TW
```

```
0040   LEDLAG          X2     X3        1.0 #T1       1.0 #T2


DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X--------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1500 PSS_G5

0050   LEDLAG          X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA          X4     VSAD                              VPMIN
VPMAX
0100   EXPORT VSAD     VSAD

DEFVAL                 VPMIN    -0.2

DEFVAL                 VPMAX     0.2

.....................................................................................
1600 PSS_G6

DEFPAR                 #K                    4.0
DEFPAR                 #TW                  10.0
DEFPAR                 #T1                   0.5
DEFPAR                 #T2                  0.10
DEFPAR                 #T3                   0.5
DEFPAR                 #T4                  0.05
0001   IMPORT DWMAQ           DW

0020   GANHO           DW     X1     #K

0030   WSHOUT          X1     X2     #TW       1.0 #TW

0040   LEDLAG          X2     X3        1.0 #T1       1.0 #T2
```

```
0050   LEDLAG          X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA          X4     VSAD                              VPMIN
VPMAX
0100   EXPORT VSAD     VSAD

DEFVAL                 VPMIN    -0.2

DEFVAL                 VPMAX     0.2

.....................................................................
1700 PSS_G7

DEFPAR                 #K                    7.5
DEFPAR                 #TW                  10.0
DEFPAR                 #T1                   0.2
DEFPAR                 #T2                  0.02
DEFPAR                 #T3                   0.5
DEFPAR                 #T4                  0.10

DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X-------------------X--------------X--------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS          LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1700 PSS_G7

0001   IMPORT DWMAQ           DW

0020   GANHO           DW     X1     #K

0030   WSHOUT          X1     X2     #TW       1.0 #TW

0040   LEDLAG          X2     X3        1.0 #T1       1.0 #T2

0050   LEDLAG          X3     X4        1.0 #T3       1.0 #T4
```

```
0060   LIMITA          X4     VSAD                              VPMIN
VPMAX
0100   EXPORT VSAD     VSAD

DEFVAL                 VPMIN    -0.2

DEFVAL                 VPMAX     0.2

.....................................................................
1800 PSS_G8

DEFPAR                 #K                      2.0
DEFPAR                 #TW                    10.0
DEFPAR                 #T1                     1.0
DEFPAR                 #T2                    0.20
DEFPAR                 #T3                     1.0
DEFPAR                 #T4                    0.10
0001   IMPORT DWMAQ           DW

0020   GANHO           DW     X1     #K

0030   WSHOUT          X1     X2     #TW       1.0 #TW

0040   LEDLAG          X2     X3        1.0 #T1       1.0 #T2

0050   LEDLAG          X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA          X4     VSAD                              VPMIN
VPMAX
0100   EXPORT VSAD     VSAD

DEFVAL                 VPMIN    -0.2

DEFVAL                 VPMAX     0.2


DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X--------------------X--------------X--------------------------X------X
```

```
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

1900 PSS_G9

DEFPAR                #K                     2.0
DEFPAR                #TW                   10.0
DEFPAR                #T1                    1.0
DEFPAR                #T2                   0.50
DEFPAR                #T3                    2.0
DEFPAR                #T4                   0.10
0001   IMPORT DWMAQ          DW

0020   GANHO          DW     X1     #K

0030   WSHOUT         X1     X2     #TW       1.0 #TW

0040   LEDLAG         X2     X3        1.0 #T1       1.0 #T2

0050   LEDLAG         X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA         X4     VSAD                              VPMIN
VPMAX
0100   EXPORT VSAD    VSAD

DEFVAL                VPMIN    -0.2

DEFVAL                VPMAX     0.2

.....................................................................................
2000 PSS_G10

DEFPAR                #K                     1.0
DEFPAR                #TW                   10.0
DEFPAR                #T1                    1.0
DEFPAR                #T2                   0.05
DEFPAR                #T3                    3.0
DEFPAR                #T4                   0.50
```

```
0001   IMPORT DWMAQ          DW

0020   GANHO           DW     X1     #K

0030   WSHOUT          X1     X2     #TW       1.0 #TW

0040   LEDLAG          X2     X3        1.0 #T1       1.0 #T2

0050   LEDLAG          X3     X4        1.0 #T3       1.0 #T4

0060   LIMITA          X4     VSAD                              VPMIN
VPMAX
0100   EXPORT VSAD     VSAD


DADOS DE CONTROLADORES DEFINIDOS PELO USUÁRIO

X-----X-------------------X--------------X--------------------------X------X
| CDU           BLOCO           VARIÁVEL            PARÂMETROS           LMIN |
| NUM   NUM I  TIPO  SUBTIP S ENTRAD  SAÍDA   P1     P2     P3     P4    LMAX |
X-----X----X-X------X------X-X------X------X------X------X------X------X------X

2000 PSS_G10

DEFVAL                VPMIN    -0.2

DEFVAL                VPMAX     0.2

.....................................................................................

DADOS DE MÁQUINA SINCRONA

X------------------X--X-----X---X------X-------X-------X-------X-----X-----X
|     Barra         No Pg(%)  No   Maq     Exc     Vel     Est   Reat Barra|
|  Num     Nome     Gr Qg(%)  Un   Mod    Mod T   Mod T   Mod T  Xvd  Contr|
X-----X------------X--X-----X---X------X-------X-------X-------X-----X-----X

39 BARRA-039    10  100   1    0001   1001U           1100U          39
100
```

```
31 BARRA-031    10  100   1    0002   1002U          1200U          31
100

32 BARRA-032    10  100   1    0003   1003U          1300U          32
100

33 BARRA-033    10  100   1    0004   1004U          1400U          33
100

34 BARRA-034    10  100   1    0005   1005U          1500U          34
100

35 BARRA-035    10  100   1    0006   1006U          1600U          35
100

36 BARRA-036    10  100   1    0007   1007U          1700U          36
100

37 BARRA-037    10  100   1    0008   1008U          1800U          37
100

38 BARRA-038    10  100   1    0009   1009U          1900U          38
100

30 BARRA-030    10  100   1    0010   1010U          2000U          30
100


DADOS DE EVENTOS

X-------------X---------------X-----X------X--X---X----X-------X--------------X
|    Evento         Elemento             %   Gr Und  Blk  Resist     Nome 1     |
|Tipo   Tempo    Tip             Num    ABS     Uni       Reatan     Nome 2   D|
|                Num   Para Nc  Extr            C        Suscep               |
|                                               Unf       Defas.               |
X----X--------X------X-----X--X-----X------X--X---X----X-------X------------X-X

RMGR    0.000 Ger                                              BARRA-032
```

32

DADOS DE PLOTAGEM

Elemento			Refer		Num	Bloco	Identificação
Tipo	M Tip	Num	Para Nc Gr	Barra Gr	Extr	P	
FREQ	Bar	1					BARRA-001
FREQ	Bar	2					BARRA-002
FREQ	Bar	3					BARRA-003
FREQ	Bar	4					BARRA-004
FREQ	Bar	5					BARRA-005
FREQ	Bar	6					BARRA-006
FREQ	Bar	7					BARRA-007
FREQ	Bar	8					BARRA-008
FREQ	Bar	9					BARRA-009
FREQ	Bar	10					BARRA-010
FREQ	Bar	11					BARRA-011
FREQ	Bar	12					BARRA-012
FREQ	Bar	13					BARRA-013
FREQ	Bar	14					BARRA-014
FREQ	Bar	15					BARRA-015
FREQ	Bar	16					BARRA-016
FREQ	Bar	17					BARRA-017
FREQ	Bar	18					BARRA-018
FREQ	Bar	19					BARRA-019
FREQ	Bar	20					BARRA-020
FREQ	Bar	21					BARRA-021
FREQ	Bar	22					BARRA-022
FREQ	Bar	23					BARRA-023
FREQ	Bar	24					BARRA-024
FREQ	Bar	25					BARRA-025
FREQ	Bar	26					BARRA-026
FREQ	Bar	27					BARRA-027
FREQ	Bar	28					BARRA-028

```
FREQ    Bar     29                                  BARRA-029
FREQ    Bar     30                                  BARRA-030
FREQ    Bar     31                                  BARRA-031
FREQ    Bar     32                                  BARRA-032
FREQ    Bar     33                                  BARRA-033
FREQ    Bar     34                                  BARRA-034
FREQ    Bar     35                                  BARRA-035
FREQ    Bar     36                                  BARRA-036
FREQ    Bar     37                                  BARRA-037
FREQ    Bar     38                                  BARRA-038
FREQ    Bar     39                                  BARRA-039
VOLT    Bar      1                                  BARRA-001
VOLT    Bar      2                                  BARRA-002
VOLT    Bar      3                                  BARRA-003
VOLT    Bar      4                                  BARRA-004
VOLT    Bar      5                                  BARRA-005
VOLT    Bar      6                                  BARRA-006
VOLT    Bar      7                                  BARRA-007
VOLT    Bar      8                                  BARRA-008
VOLT    Bar      9                                  BARRA-009

DADOS DE PLOTAGEM

X-------X----------------------X--------X-----X-----X-------------------------X
|               Elemento          Refer    Num  Bloco       Identificação      |
|Tipo  M Tip   Num   Para Nc Gr Barra Gr  Extr P                                |
X-------X---X------X-----X--X--X-----X--X-----X-----X------------X------------X

VOLT    Bar     10                                  BARRA-010
VOLT    Bar     11                                  BARRA-011
VOLT    Bar     12                                  BARRA-012
VOLT    Bar     13                                  BARRA-013
VOLT    Bar     14                                  BARRA-014
VOLT    Bar     15                                  BARRA-015
VOLT    Bar     16                                  BARRA-016
VOLT    Bar     17                                  BARRA-017
VOLT    Bar     18                                  BARRA-018
VOLT    Bar     19                                  BARRA-019
VOLT    Bar     20                                  BARRA-020
```

VOLT	Bar	21	BARRA-021
VOLT	Bar	22	BARRA-022
VOLT	Bar	23	BARRA-023
VOLT	Bar	24	BARRA-024
VOLT	Bar	25	BARRA-025
VOLT	Bar	26	BARRA-026
VOLT	Bar	27	BARRA-027
VOLT	Bar	28	BARRA-028
VOLT	Bar	29	BARRA-029
VOLT	Bar	30	BARRA-030
VOLT	Bar	31	BARRA-031
VOLT	Bar	32	BARRA-032
VOLT	Bar	33	BARRA-033
VOLT	Bar	34	BARRA-034
VOLT	Bar	35	BARRA-035
VOLT	Bar	36	BARRA-036
VOLT	Bar	37	BARRA-037
VOLT	Bar	38	BARRA-038
VOLT	Bar	39	BARRA-039
ANGL	Bar	1	BARRA-001
ANGL	Bar	2	BARRA-002
ANGL	Bar	3	BARRA-003
ANGL	Bar	4	BARRA-004
ANGL	Bar	5	BARRA-005
ANGL	Bar	6	BARRA-006
ANGL	Bar	7	BARRA-007
ANGL	Bar	8	BARRA-008
ANGL	Bar	9	BARRA-009
ANGL	Bar	10	BARRA-010
ANGL	Bar	11	BARRA-011
ANGL	Bar	12	BARRA-012
ANGL	Bar	13	BARRA-013
ANGL	Bar	14	BARRA-014
ANGL	Bar	15	BARRA-015
ANGL	Bar	16	BARRA-016
ANGL	Bar	17	BARRA-017
ANGL	Bar	18	BARRA-018

DADOS DE PLOTAGEM

Elemento						Refer		Num	Bloco	Identificação	
Tipo	M Tip	Num	Para	Nc	Gr	Barra	Gr	Extr	P		
ANGL	Bar	19								BARRA-019	
ANGL	Bar	20								BARRA-020	
ANGL	Bar	21								BARRA-021	
ANGL	Bar	22								BARRA-022	
ANGL	Bar	23								BARRA-023	
ANGL	Bar	24								BARRA-024	
ANGL	Bar	25								BARRA-025	
ANGL	Bar	26								BARRA-026	
ANGL	Bar	27								BARRA-027	
ANGL	Bar	28								BARRA-028	
ANGL	Bar	29								BARRA-029	
ANGL	Bar	30								BARRA-030	
ANGL	Bar	31								BARRA-031	
ANGL	Bar	32								BARRA-032	
ANGL	Bar	33								BARRA-033	
ANGL	Bar	34								BARRA-034	
ANGL	Bar	35								BARRA-035	
ANGL	Bar	36								BARRA-036	
ANGL	Bar	37								BARRA-037	
ANGL	Bar	38								BARRA-038	
ANGL	Bar	39								BARRA-039	
FLXA	Cir	1	2	1				1		BARRA-001	BARRA-002
FLXR	Cir	1	2	1				1		BARRA-001	BARRA-002
FLXA	Cir	1	39	1				1		BARRA-001	BARRA-039
FLXR	Cir	1	39	1				1		BARRA-001	BARRA-039
FLXA	Cir	2	3	1				2		BARRA-002	BARRA-003
FLXR	Cir	2	3	1				2		BARRA-002	BARRA-003
FLXA	Cir	2	25	1				2		BARRA-002	BARRA-025
FLXR	Cir	2	25	1				2		BARRA-002	BARRA-025
FLXA	Cir	2	30	1				2		BARRA-002	BARRA-030
FLXR	Cir	2	30	1				2		BARRA-002	BARRA-030
FLXA	Cir	2	1	1				2		BARRA-002	BARRA-001
FLXR	Cir	2	1	1				2		BARRA-002	BARRA-001

```
PCAR    Bar      3                                  BARRA-003
QCAR    Bar      3                                  BARRA-003
FLXA    Cir      3     4  1                3       BARRA-003    BARRA-004
FLXR    Cir      3     4  1                3       BARRA-003    BARRA-004
FLXA    Cir      3    18  1                3       BARRA-003    BARRA-018
FLXR    Cir      3    18  1                3       BARRA-003    BARRA-018
FLXA    Cir      3     2  1                3       BARRA-003    BARRA-002
FLXR    Cir      3     2  1                3       BARRA-003    BARRA-002
PCAR    Bar      4                                  BARRA-004
QCAR    Bar      4                                  BARRA-004
FLXA    Cir      4     5  1                4       BARRA-004    BARRA-005
FLXR    Cir      4     5  1                4       BARRA-004    BARRA-005
FLXA    Cir      4    14  1                4       BARRA-004    BARRA-014
FLXR    Cir      4    14  1                4       BARRA-004    BARRA-014
FLXA    Cir      4     3  1                4       BARRA-004    BARRA-003
```

DADOS DE PLOTAGEM

```
X-------X----------------------X--------X-----X-----X-------------------------X
|               Elemento          Refer    Num  Bloco       Identificação      |
|Tipo  M Tip   Num   Para Nc Gr Barra Gr  Extr P                                |
X-------X---X------X-----X--X--X-----X--X-----X-----X------------X------------X

FLXR    Cir      4     3  1                4       BARRA-004    BARRA-003
FLXA    Cir      5     6  1                5       BARRA-005    BARRA-006
FLXR    Cir      5     6  1                5       BARRA-005    BARRA-006
FLXA    Cir      5     8  1                5       BARRA-005    BARRA-008
FLXR    Cir      5     8  1                5       BARRA-005    BARRA-008
FLXA    Cir      5     4  1                5       BARRA-005    BARRA-004
FLXR    Cir      5     4  1                5       BARRA-005    BARRA-004
FLXA    Cir      6     7  1                6       BARRA-006    BARRA-007
FLXR    Cir      6     7  1                6       BARRA-006    BARRA-007
FLXA    Cir      6    11  1                6       BARRA-006    BARRA-011
FLXR    Cir      6    11  1                6       BARRA-006    BARRA-011
FLXA    Cir      6    31  1                6       BARRA-006    BARRA-031
FLXR    Cir      6    31  1                6       BARRA-006    BARRA-031
FLXA    Cir      6     5  1                6       BARRA-006    BARRA-005
FLXR    Cir      6     5  1                6       BARRA-006    BARRA-005
PCAR    Bar      7                                  BARRA-007
```

```
QCAR    Bar      7                                  BARRA-007
FLXA    Cir      7     8  1                 7       BARRA-007    BARRA-008
FLXR    Cir      7     8  1                 7       BARRA-007    BARRA-008
FLXA    Cir      7     6  1                 7       BARRA-007    BARRA-006
FLXR    Cir      7     6  1                 7       BARRA-007    BARRA-006
PCAR    Bar      8                                  BARRA-008
QCAR    Bar      8                                  BARRA-008
FLXA    Cir      8     9  1                 8       BARRA-008    BARRA-009
FLXR    Cir      8     9  1                 8       BARRA-008    BARRA-009
FLXA    Cir      8     5  1                 8       BARRA-008    BARRA-005
FLXR    Cir      8     5  1                 8       BARRA-008    BARRA-005
FLXA    Cir      8     7  1                 8       BARRA-008    BARRA-007
FLXR    Cir      8     7  1                 8       BARRA-008    BARRA-007
FLXA    Cir      9    39  1                 9       BARRA-009    BARRA-039
FLXR    Cir      9    39  1                 9       BARRA-009    BARRA-039
FLXA    Cir      9     8  1                 9       BARRA-009    BARRA-008
FLXR    Cir      9     8  1                 9       BARRA-009    BARRA-008
FLXA    Cir     10    11  1                10       BARRA-010    BARRA-011
FLXR    Cir     10    11  1                10       BARRA-010    BARRA-011
FLXA    Cir     10    13  1                10       BARRA-010    BARRA-013
FLXR    Cir     10    13  1                10       BARRA-010    BARRA-013
FLXA    Cir     10    32  1                10       BARRA-010    BARRA-032
FLXR    Cir     10    32  1                10       BARRA-010    BARRA-032
FLXA    Cir     11    12  1                11       BARRA-011    BARRA-012
FLXR    Cir     11    12  1                11       BARRA-011    BARRA-012
FLXA    Cir     11     6  1                11       BARRA-011    BARRA-006
FLXR    Cir     11     6  1                11       BARRA-011    BARRA-006
FLXA    Cir     11    10  1                11       BARRA-011    BARRA-010
FLXR    Cir     11    10  1                11       BARRA-011    BARRA-010
PCAR    Bar     12                                 BARRA-012
QCAR    Bar     12                                 BARRA-012
FLXA    Cir     12    13  1                12       BARRA-012    BARRA-013
```

DADOS DE PLOTAGEM

```
X-------X----------------------X--------X-----X-----X-------------------------X
|               Elemento          Refer    Num  Bloco       Identificação      |
|Tipo  M Tip   Num   Para Nc Gr Barra Gr  Extr P                                |
X-------X---X------X-----X--X--X-----X--X-----X-----X------------X------------X
```

FLXR	Cir	12	13	1	12	BARRA-012	BARRA-013
FLXA	Cir	12	11	1	12	BARRA-012	BARRA-011
FLXR	Cir	12	11	1	12	BARRA-012	BARRA-011
FLXA	Cir	13	14	1	13	BARRA-013	BARRA-014
FLXR	Cir	13	14	1	13	BARRA-013	BARRA-014
FLXA	Cir	13	10	1	13	BARRA-013	BARRA-010
FLXR	Cir	13	10	1	13	BARRA-013	BARRA-010
FLXA	Cir	13	12	1	13	BARRA-013	BARRA-012
FLXR	Cir	13	12	1	13	BARRA-013	BARRA-012
FLXA	Cir	14	15	1	14	BARRA-014	BARRA-015
FLXR	Cir	14	15	1	14	BARRA-014	BARRA-015
FLXA	Cir	14	4	1	14	BARRA-014	BARRA-004
FLXR	Cir	14	4	1	14	BARRA-014	BARRA-004
FLXA	Cir	14	13	1	14	BARRA-014	BARRA-013
FLXR	Cir	14	13	1	14	BARRA-014	BARRA-013
PCAR	Bar	15				BARRA-015	
QCAR	Bar	15				BARRA-015	
FLXA	Cir	15	16	1	15	BARRA-015	BARRA-016
FLXR	Cir	15	16	1	15	BARRA-015	BARRA-016
FLXA	Cir	15	14	1	15	BARRA-015	BARRA-014
FLXR	Cir	15	14	1	15	BARRA-015	BARRA-014
PCAR	Bar	16				BARRA-016	
QCAR	Bar	16				BARRA-016	
FLXA	Cir	16	17	1	16	BARRA-016	BARRA-017
FLXR	Cir	16	17	1	16	BARRA-016	BARRA-017
FLXA	Cir	16	19	1	16	BARRA-016	BARRA-019
FLXR	Cir	16	19	1	16	BARRA-016	BARRA-019
FLXA	Cir	16	21	1	16	BARRA-016	BARRA-021
FLXR	Cir	16	21	1	16	BARRA-016	BARRA-021
FLXA	Cir	16	24	1	16	BARRA-016	BARRA-024
FLXR	Cir	16	24	1	16	BARRA-016	BARRA-024
FLXA	Cir	16	15	1	16	BARRA-016	BARRA-015
FLXR	Cir	16	15	1	16	BARRA-016	BARRA-015
FLXA	Cir	17	18	1	17	BARRA-017	BARRA-018
FLXR	Cir	17	18	1	17	BARRA-017	BARRA-018
FLXA	Cir	17	27	1	17	BARRA-017	BARRA-027
FLXR	Cir	17	27	1	17	BARRA-017	BARRA-027
FLXA	Cir	17	16	1	17	BARRA-017	BARRA-016

FLXR	Cir	17	16	1		17	BARRA-017	BARRA-016
PCAR	Bar	18					BARRA-018	
QCAR	Bar	18					BARRA-018	
FLXA	Cir	18	3	1		18	BARRA-018	BARRA-003
FLXR	Cir	18	3	1		18	BARRA-018	BARRA-003
FLXA	Cir	18	17	1		18	BARRA-018	BARRA-017
FLXR	Cir	18	17	1		18	BARRA-018	BARRA-017
FLXA	Cir	19	20	1		19	BARRA-019	BARRA-020
FLXR	Cir	19	20	1		19	BARRA-019	BARRA-020
FLXA	Cir	19	33	1		19	BARRA-019	BARRA-033

DADOS DE PLOTAGEM

Elemento						Refer	Num	Bloco	Identificação	
Tipo	M Tip	Num	Para	Nc	Gr	Barra Gr	Extr	P		
FLXR	Cir	19	33	1		19		BARRA-019	BARRA-033	
FLXA	Cir	19	16	1		19		BARRA-019	BARRA-016	
FLXR	Cir	19	16	1		19		BARRA-019	BARRA-016	
PCAR	Bar	20						BARRA-020		
QCAR	Bar	20						BARRA-020		
FLXA	Cir	20	34	1		20		BARRA-020	BARRA-034	
FLXR	Cir	20	34	1		20		BARRA-020	BARRA-034	
FLXA	Cir	20	19	1		20		BARRA-020	BARRA-019	
FLXR	Cir	20	19	1		20		BARRA-020	BARRA-019	
PCAR	Bar	21						BARRA-021		
QCAR	Bar	21						BARRA-021		
FLXA	Cir	21	22	1		21		BARRA-021	BARRA-022	
FLXR	Cir	21	22	1		21		BARRA-021	BARRA-022	
FLXA	Cir	21	16	1		21		BARRA-021	BARRA-016	
FLXR	Cir	21	16	1		21		BARRA-021	BARRA-016	
FLXA	Cir	22	23	1		22		BARRA-022	BARRA-023	
FLXR	Cir	22	23	1		22		BARRA-022	BARRA-023	
FLXA	Cir	22	35	1		22		BARRA-022	BARRA-035	
FLXR	Cir	22	35	1		22		BARRA-022	BARRA-035	
FLXA	Cir	22	21	1		22		BARRA-022	BARRA-021	
FLXR	Cir	22	21	1		22		BARRA-022	BARRA-021	

```
PCAR    Bar     23                                  BARRA-023
QCAR    Bar     23                                  BARRA-023
FLXA    Cir     23    24  1                 23      BARRA-023    BARRA-024
FLXR    Cir     23    24  1                 23      BARRA-023    BARRA-024
FLXA    Cir     23    36  1                 23      BARRA-023    BARRA-036
FLXR    Cir     23    36  1                 23      BARRA-023    BARRA-036
FLXA    Cir     23    22  1                 23      BARRA-023    BARRA-022
FLXR    Cir     23    22  1                 23      BARRA-023    BARRA-022
PCAR    Bar     24                                  BARRA-024
QCAR    Bar     24                                  BARRA-024
FLXA    Cir     24    16  1                 24      BARRA-024    BARRA-016
FLXR    Cir     24    16  1                 24      BARRA-024    BARRA-016
FLXA    Cir     24    23  1                 24      BARRA-024    BARRA-023
FLXR    Cir     24    23  1                 24      BARRA-024    BARRA-023
PCAR    Bar     25                                  BARRA-025
QCAR    Bar     25                                  BARRA-025
FLXA    Cir     25    26  1                 25      BARRA-025    BARRA-026
FLXR    Cir     25    26  1                 25      BARRA-025    BARRA-026
FLXA    Cir     25    37  1                 25      BARRA-025    BARRA-037
FLXR    Cir     25    37  1                 25      BARRA-025    BARRA-037
FLXA    Cir     25     2  1                 25      BARRA-025    BARRA-002
FLXR    Cir     25     2  1                 25      BARRA-025    BARRA-002
PCAR    Bar     26                                  BARRA-026
QCAR    Bar     26                                  BARRA-026
FLXA    Cir     26    27  1                 26      BARRA-026    BARRA-027
FLXR    Cir     26    27  1                 26      BARRA-026    BARRA-027
FLXA    Cir     26    28  1                 26      BARRA-026    BARRA-028

DADOS DE PLOTAGEM

X-------X----------------------X--------X-----X-----X------------------------X
|               Elemento          Refer    Num  Bloco       Identificação      |
|Tipo  M Tip   Num   Para Nc Gr Barra Gr  Extr P                                |
X-------X---X------X-----X--X--X-----X--X-----X-----X------------X------------X

FLXR    Cir     26    28  1                 26      BARRA-026    BARRA-028
FLXA    Cir     26    29  1                 26      BARRA-026    BARRA-029
FLXR    Cir     26    29  1                 26      BARRA-026    BARRA-029
FLXA    Cir     26    25  1                 26      BARRA-026    BARRA-025
```

FLXR	Cir	26	25	1		26	BARRA-026	BARRA-025
PCAR	Bar	27					BARRA-027	
QCAR	Bar	27					BARRA-027	
FLXA	Cir	27	17	1		27	BARRA-027	BARRA-017
FLXR	Cir	27	17	1		27	BARRA-027	BARRA-017
FLXA	Cir	27	26	1		27	BARRA-027	BARRA-026
FLXR	Cir	27	26	1		27	BARRA-027	BARRA-026
PCAR	Bar	28					BARRA-028	
QCAR	Bar	28					BARRA-028	
FLXA	Cir	28	29	1		28	BARRA-028	BARRA-029
FLXR	Cir	28	29	1		28	BARRA-028	BARRA-029
FLXA	Cir	28	26	1		28	BARRA-028	BARRA-026
FLXR	Cir	28	26	1		28	BARRA-028	BARRA-026
PCAR	Bar	29					BARRA-029	
QCAR	Bar	29					BARRA-029	
FLXA	Cir	29	38	1		29	BARRA-029	BARRA-038
FLXR	Cir	29	38	1		29	BARRA-029	BARRA-038
FLXA	Cir	29	26	1		29	BARRA-029	BARRA-026
FLXR	Cir	29	26	1		29	BARRA-029	BARRA-026
FLXA	Cir	29	28	1		29	BARRA-029	BARRA-028
FLXR	Cir	29	28	1		29	BARRA-029	BARRA-028
PGER	Bar	30					BARRA-030	
QGER	Bar	30					BARRA-030	
FLXA	Cir	30	2	1		30	BARRA-030	BARRA-002
FLXR	Cir	30	2	1		30	BARRA-030	BARRA-002
PGER	Bar	31					BARRA-031	
QGER	Bar	31					BARRA-031	
PCAR	Bar	31					BARRA-031	
QCAR	Bar	31					BARRA-031	
FLXA	Cir	31	6	1		31	BARRA-031	BARRA-006
FLXR	Cir	31	6	1		31	BARRA-031	BARRA-006
PGER	Bar	32					BARRA-032	
QGER	Bar	32					BARRA-032	
FLXA	Cir	32	10	1		32	BARRA-032	BARRA-010
FLXR	Cir	32	10	1		32	BARRA-032	BARRA-010
PGER	Bar	33					BARRA-033	
QGER	Bar	33					BARRA-033	
FLXA	Cir	33	19	1		33	BARRA-033	BARRA-019
FLXR	Cir	33	19	1		33	BARRA-033	BARRA-019

PGER	Bar	34					BARRA-034	
QGER	Bar	34					BARRA-034	
FLXA	Cir	34	20	1		34	BARRA-034	BARRA-020
FLXR	Cir	34	20	1		34	BARRA-034	BARRA-020
PGER	Bar	35					BARRA-035	

DADOS DE PLOTAGEM

Elemento						Refer	Num	Bloco	Identificação	
Tipo	M Tip	Num	Para	Nc	Gr	Barra Gr	Extr	P		
QGER	Bar	35							BARRA-035	
FLXA	Cir	35	22	1			35		BARRA-035	BARRA-022
FLXR	Cir	35	22	1			35		BARRA-035	BARRA-022
PGER	Bar	36							BARRA-036	
QGER	Bar	36							BARRA-036	
FLXA	Cir	36	23	1			36		BARRA-036	BARRA-023
FLXR	Cir	36	23	1			36		BARRA-036	BARRA-023
PGER	Bar	37							BARRA-037	
QGER	Bar	37							BARRA-037	
FLXA	Cir	37	25	1			37		BARRA-037	BARRA-025
FLXR	Cir	37	25	1			37		BARRA-037	BARRA-025
PGER	Bar	38							BARRA-038	
QGER	Bar	38							BARRA-038	
FLXA	Cir	38	29	1			38		BARRA-038	BARRA-029
FLXR	Cir	38	29	1			38		BARRA-038	BARRA-029
PGER	Bar	39							BARRA-039	
QGER	Bar	39							BARRA-039	
PCAR	Bar	39							BARRA-039	
QCAR	Bar	39							BARRA-039	
FLXA	Cir	39	1	1			39		BARRA-039	BARRA-001
FLXR	Cir	39	1	1			39		BARRA-039	BARRA-001
FLXA	Cir	39	9	1			39		BARRA-039	BARRA-009
FLXR	Cir	39	9	1			39		BARRA-039	BARRA-009
ILIN	Cir	39	1	1			39		BARRA-039	BARRA-001
ILIN	Cir	39	9	1			39		BARRA-039	BARRA-009

DADOS DE SIMULAÇÃO

Tempo simulação	Intervalo integração	Frequência plotagem	pontos impressão	fatoração
20	.01	1	1	1

CEPEL - CENTRO DE PESQUISAS DE ENERGIA ELÉTRICA - ANATEM - v12.0.0-x64

39bus_system_carregamento_caso_base
39 GENERATOR 10 BUS SYSTEM

T= 0.0s RMGR - Desligar usina da barra 32 BARRA-032

T= 0.0s Desligou grupo 10 de gerador da barra 32 BARRA-032

Tempo de CPU da simulação: 00:00:00.54
Média de soluções de rede CA por passo: 22.84
Média de iterações modelos CA - rede CA por passo: 11.89

APÊNDICE 2 – CÓDIGO DE CRIAÇÃO DOS MODELOS DE CLASSIFICAÇÃO EM 2 ÁREAS

```
# Importação das bibliotecas

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import h5py
import scipy
from PIL import Image
from scipy import ndimage
#from lr_utils import load_dataset

# %matplotlib inline

# Carregando a matriz de características
file = open('X.csv')
X    = np.loadtxt(file, delimiter=",")
# Cada linha denota um instante de tempo onde as PMUs fizeram medições.
# Cada coluna um sinal diferente medido pela PMU

print(X)
print(X.shape)

# Cada linha desempenhará o papel de uma amostra no nosso
processo de treinamento
# Cada coluna desempenhará o papel de uma característica no nosso
processo de treinamento

# Carregando vetor de classes (rótulos) correspondente

file = open('y.csv')
y    = np.loadtxt(file, delimiter=",")
y    = y.reshape((1,y.shape[0]))

print(y)
print(y.shape)
```

```
# Exemplo de uma caracteristica (todas as amostras)
index = 100

t  = np.arange(0, X.shape[0], 1)
print(t.shape)

plt.plot(t,X[:,index])

plt.plot(t,y[0,:])

X = X.T # Fazendo o transposto para que cada linha agora contenha uma
característica e cada coluna agora seja uma amostra
print(X)
print(X.shape)

n = X.shape[0] # número de características
print(n)

m = X.shape[1] # número total de amostras (treinamento + teste)
print(m)

# antes de separarmos em dados de treinamento e dados de teste, temos que
misturar as amostras. Ou seja, aleatoriamente trocar
# as posições das colunas da nossa matriz de dados.

np.random.seed(100) # definindo a semente do gerador de números aleatórios

n_col                       = X.shape[1]
novas_posicoes_das_colunas = np.random.permutation(n_col)
X                           = X[:,novas_posicoes_das_colunas]
y                           = y[:,novas_posicoes_das_colunas]

print(X.shape)
print(y.shape)

plt.plot(t,X[index,:])

plt.plot(t,y[0,:])
```

```
# agora sim podemos separar em dados de treinamento e dados de teste
print(X.shape)

m_treinamento = int(m*0.7) # pegando cerca de 70% das amostras para realizar
o treinamento
m_teste = m - m_treinamento # as demais amostras são usadas para fins de teste

print(m_treinamento)
print(m_teste)
print(m_treinamento + m_teste)

colunas = np.arange(0, m_treinamento, 1)
print(colunas)
dados_treinamento_x = X[:,colunas]
dados_treinamento_y = y[:,colunas]
print(dados_treinamento_x.shape)
print(dados_treinamento_y.shape)

colunas = np.arange(m_treinamento, m, 1)
print(colunas)
dados_teste_x = X[:,colunas]
dados_teste_y = y[:,colunas]
print(dados_teste_x.shape)
print(dados_teste_y.shape)
print(dados_treinamento_x.shape)

m_trein = dados_treinamento_x.shape[1]
m_teste =  dados_teste_x.shape[1]

print ("Numero de exemplos para treinamento: m_train = " + str(m_treinamento))
print ("Number de exemplos para teste: m_test = " + str(m_teste))
print ("dados_treinamento_x shape: " + str(dados_treinamento_x.shape))
print ("dados_treinamento_y shape: " + str(dados_treinamento_y.shape))
print ("dados_teste_x shape: " + str(dados_teste_x.shape))
print ("dados_teste_y shape: " + str(dados_teste_y.shape))

#definindo dados
```

```
train_set_x = dados_treinamento_x
test_set_x  = dados_teste_x
train_set_y = dados_treinamento_y
test_set_y  = dados_teste_y

from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler

scaler             = StandardScaler().fit(train_set_x.T)
train_set_x_scaled = scaler.transform(train_set_x.T)
test_set_x_scaled  = scaler.transform(test_set_x.T)
LogReg             = LogisticRegression(class_weight='balanced',solver='lbfgs'
max_iter=1000,tol=1e-6,multi_class='ovr')
print(train_set_x_scaled)
LogReg.fit(train_set_x_scaled, train_set_y.T)
print(LogReg.score(train_set_x_scaled,train_set_y.T))
print(LogReg.score(test_set_x_scaled,test_set_y.T))
print(LogReg.predict(test_set_x_scaled))
print(test_set_y)

from sklearn.metrics import confusion_matrix

y_verdadeiro_reglog = test_set_y.T
y_previsto_reglog   = LogReg.predict(test_set_x_scaled)

matriz_de_confusao_reglog = confusion_matrix(y_verdadeiro_reglog,
y_previsto_reglog)
print(matriz_de_confusao_reglog)

sns.heatmap(matriz_de_confusao_reglog, cmap="viridis", annot=True, fmt='d')
plt.xlabel("Rótulo Previsto")
plt.ylabel("Rótulo Verdadeiro")
matriz_reglog = plt.gcf()
#plt.show()
matriz_reglog.savefig("matrizconfusaoreglog.pdf", bbox_inches = "tight")

from sklearn.neural_network import MLPClassifier

NN_clf = MLPClassifier(solver='lbfgs', alpha=1e-5, random_state=1,
```

```
max_iter = 1000)
NN_clf.fit(train_set_x_scaled, train_set_y.T)

print(NN_clf.score(train_set_x_scaled,train_set_y.T))
print(NN_clf.score(test_set_x_scaled,test_set_y.T))
print(NN_clf.predict(test_set_x_scaled))
print(test_set_y)

y_chapeu   = NN_clf.predict(test_set_x_scaled).reshape(-1,)
test_set_y = test_set_y.reshape(-1,)
print(y_chapeu.shape, test_set_y.shape)

from sklearn.metrics import confusion_matrix
#from sklearn.metrics import ConfusionMatrixDisplay

#print(np.sum(test_set_y==0))

y_true             = test_set_y
y_pred             = y_chapeu
matriz_de_confusao_mlp = confusion_matrix(y_true, y_pred)
#rotulos            = ["y=0","y=1"]
#matriz_de_confusao_display = ConfusionMatrixDisplay(confusion_matrix =
matriz_de_confusao, display_labels=rotulos)

#matriz_de_confusao_display.plot()
#matriz_de_confusao_display.text_ = ["Rótulo verdadeiro","Rótulo estimado"]

tn, fp, fn, tp = matriz_de_confusao_mlp.ravel()

print(tn,fp,fn,tp)

sns.heatmap(matriz_de_confusao_mlp, cmap="viridis", annot=True, fmt='d')
plt.xlabel("Rótulo Previsto")
plt.ylabel("Rótulo Verdadeiro")
matriz_mlp = plt.gcf()
#plt.show()
matriz_mlp.savefig("matrizconfusaomlp.pdf", bbox_inches = "tight")
```

APÊNDICE 3 – CÓDIGO DE CRIAÇÃO DO MODELO DE CLASSIFICAÇÃO EM 4 ÁREAS

```
#importação das bibliotecas

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# importação do tensorflow

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
import logging
logging.getLogger("tensorflow").setLevel(logging.ERROR)
tf.autograph.set_verbosity(0)

arquivo = 0

print(arquivo)

# Carregando a matriz de características
file = open(f'X_est_arquivo{arquivo}.csv')
X = np.loadtxt(file, delimiter=",")
# Cada linha denota um instante de tempo onde as PMUs fizeram medições.
# Cada coluna um sinal diferente medido pela PMU

# Carregando vetor de classes (rótulos) correspondente
file = open(f'y_est_arquivo{arquivo}.csv')
y    = np.loadtxt(file, delimiter=",")
y    = y.reshape((-1,1))

print(X.shape, y.shape)

np.random.seed(100) # definindo a semente do gerador de números aleatórios
"""
```

```
np.random.seed(100) # definindo a semente do gerador de números aleatórios

n_linhas                    = X.shape[0]
novas_posicoes_das_linhas   = np.random.permutation(n_linhas)
X_est                       = X[novas_posicoes_das_linhas,:]
y_est                       = y[novas_posicoes_das_linhas,:]

print(X_est)
print(y_est)

# abaixo normalizamos as características
camada_norm = tf.keras.layers.Normalization(axis=-1)
camada_norm.adapt(X_est)  # calcula média e variância
X_est_scaled = camada_norm(X_est) # características normalizadas

fator_duplicacao = 1;
X_est_scaled     = np.tile(X_est_scaled,(fator_duplicacao,1))
y_est            = np.tile(y_est,(fator_duplicacao,1))
print(X_est_scaled.shape, y_est.shape)

# abaixo treinamos uma rede neural usando tensorflow
n      = X_est.shape[1]
modelo = Sequential(
[
tf.keras.Input(shape=(n,)),
Dense(60, activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.0000)),
Dense(20, activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.0000)),
Dense(4, activation="softmax")
], name = "meu_modelo"
)
#modelo.summary()
modelo.compile(
#loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),
#loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
loss=tf.keras.losses.SparseCategoricalCrossentropy(),
optimizer=tf.keras.optimizers.Adam(0.001),
```

```
)
historico = modelo.fit(
X_est_scaled,y_est,
epochs=20
)

fig,ax = plt.subplots(1,1, figsize = (4,3))
ax.plot(historico.history['loss'], label='perda')
ax.set_ylim([0, 2])
ax.set_xlabel('Época')
ax.set_ylabel('Perda (Custo)')
ax.legend()
ax.grid(True)
perda_x_epoca = plt.gcf()
#plt.show()
perda_x_epoca.savefig("perdaporepoca.pdf", bbox_inches = "tight")

# taxa de acerto para dados de estimação
logit       = modelo(X_est_scaled)
Previsao    = np.argmax(logit,axis=1)
#Previsao    = modelo(X_est_scaled)
#Yhat        = (Previsao.numpy() >= 0.5).astype(int)
taxa_acerto_est = np.mean((Previsao.reshape(-1,1)==y_est))
print(f"média da taxa de acerto para os eventos de estimação: {taxa_acerto_est

## Abaixo testamos outros dados de validação (dados de outros eventos)
file  = open(f'X_val_arquivo{arquivo}.csv')
X_val = np.loadtxt(file, delimiter=",")
file  = open(f'y_val_arquivo{arquivo}.csv')
y_val = np.loadtxt(file, delimiter=",").reshape((-1,))
X_val_scaled  = camada_norm(X_val)

print(X_val_scaled.shape)
print(y_val.shape)

# taxa de acerto para dados de validação (outros eventos não vistos pelo
modelo durante treinamento)
logit       = modelo(X_val_scaled)
Previsao    = np.argmax(logit, axis=1)
```

```
#Previsao    = modelo(X_val_scaled)
#Yhat        = (Previsao.numpy() >= 0.5).astype(int)
taxa_acerto_val = np.mean((Previsao.reshape(-1,1)==y_val.reshape(-1,1)))
print(f"média da taxa de acerto para os eventos de validação: {taxa_acerto_val

"""abaixo plotamos matriz de confusão"""

from sklearn.metrics import confusion_matrix
#from sklearn.metrics import ConfusionMatrixDisplay

y_true             = y_val.reshape(-1,1)
y_pred             = Previsao
rotulos            = [0,1,2,3]
matriz_de_confusao = confusion_matrix(y_true, y_pred, labels = rotulos)
#disp = ConfusionMatrixDisplay(confusion_matrix = matriz_de_confusao, display_

print(y_val.shape)
print(y_pred)
print(y_true)

sns.heatmap(matriz_de_confusao, cmap="viridis", annot=True, fmt='d')
plt.xlabel("Rótulo Previsto")
plt.ylabel("Rótulo Verdadeiro")
matriz = plt.gcf()
#plt.show()
matriz.savefig("matrizconfusaokeras1.pdf", bbox_inches = "tight")
```

APÊNDICE 4 – CÓDIGO DE APRIMORAMENTO DO MODELO UTILIZANDO OPTUNA

```
# Importação das bibliotecas

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import optuna
import tensorflow as tf
import logging

# Configuração do Tensorflow

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
logging.getLogger("tensorflow").setLevel(logging.ERROR)
tf.autograph.set_verbosity(0)

from sklearn.metrics import accuracy_score

# Carregando a matriz de características

arquivo = 0

file = open(f'X_est_arquivo{arquivo}.csv')
X = np.loadtxt(file, delimiter=",")
# Cada linha denota um instante de tempo onde as PMUs fizeram medições.
# Cada coluna um sinal diferente medido pela PMU

# Carregando vetor de classes (rótulos) correspondente
file = open(f'y_est_arquivo{arquivo}.csv')
y    = np.loadtxt(file, delimiter=",")
y    = y.reshape((-1,1))

# Mistura das amostras

np.random.seed(100) # definindo a semente do gerador de números aleatórios
```

```
n_linhas                    = X.shape[0]
novas_posicoes_das_linhas   = np.random.permutation(n_linhas)
X_est                       = X[novas_posicoes_das_linhas,:]
y_est                       = y[novas_posicoes_das_linhas,:]

# Normalização das características
camada_norm = tf.keras.layers.Normalization(axis=-1)
camada_norm.adapt(X_est)  # calcula média e variância
X_est_scaled = camada_norm(X_est) # características normalizadas

## Abaixo carregamos os dados de validação (dados de outros eventos)
file  = open(f'X_val_arquivo{arquivo}.csv')
X_val = np.loadtxt(file, delimiter=",")
file  = open(f'y_val_arquivo{arquivo}.csv')
y_val = np.loadtxt(file, delimiter=",").reshape((-1,))
X_val_scaled  = camada_norm(X_val)

fator_duplicacao = 1;
X_est_scaled     = np.tile(X_est_scaled,(fator_duplicacao,1))
y_est            = np.tile(y_est,(fator_duplicacao,1))
print(X_est_scaled.shape, y_est.shape)

# Treinamento de uma rede neural usando tensorflow
n      = X_est.shape[1]
modelo = Sequential(
[
tf.keras.Input(shape=(n,)),
Dense(140, activation="relu",
 kernel_regularizer=tf.keras.regularizers.l2(0.00)),
Dense(60, activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.00)),
Dense(30, activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.00)),
Dense(4, activation="softmax")
], name = "meu_modelo"
)
#modelo.summary()
modelo.compile(
```

```
#loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),
#loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
loss=tf.keras.losses.SparseCategoricalCrossentropy(),
optimizer=tf.keras.optimizers.Adam(0.001),
)

historico = modelo.fit(
X_est_scaled,y_est,
epochs=40
)

fig,ax = plt.subplots(1,1, figsize = (4,3))
ax.plot(historico.history['loss'], label='perda')
ax.set_ylim([0, 2])
ax.set_xlabel('Época')
ax.set_ylabel('Perda (Custo)')
ax.legend()
ax.grid(True)
#perda_x_epoca = plt.gcf()
plt.show()
#perda_x_epoca.savefig(f'perdaporepocalambda.pdf', bbox_inches = "tight")

# taxa de acerto para dados de estimação
logit       = modelo(X_est_scaled)
Previsao    = np.argmax(logit,axis=1)
#Previsao    = modelo(X_est_scaled)
#Yhat        = (Previsao.numpy() >= 0.5).astype(int)
taxa_acerto_est = np.mean((Previsao.reshape(-1,1)==y_est))
print(f"média da taxa de acerto para os eventos de estimação: {taxa_acerto_est

# taxa de acerto para dados de validação (outros eventos não vistos pelo
modelo durante treinamento)
logit       = modelo(X_val_scaled)
Previsao    = np.argmax(logit, axis=1)
#Previsao    = modelo(X_val_scaled)
#Yhat        = (Previsao.numpy() >= 0.5).astype(int)
taxa_acerto_val = np.mean((Previsao.reshape(-1,1)==y_val.reshape(-1,1)))
print(f"média da taxa de acerto para os eventos de validação: {taxa_acerto_val
```

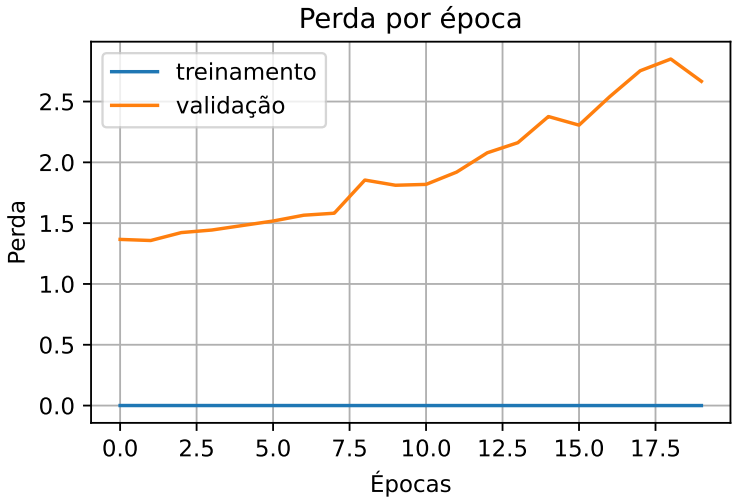
```
def objetivo(trial):
# Definindo hiperparametros para pesquisa
neuronios_camada1 = trial.suggest_int('neuronios_camada1', 50, 150)
neuronios_camada2 = trial.suggest_int('neuronios_camada2', 20, 100)
neuronios_camada3 = trial.suggest_int('neuronios_camada3', 10, 50)
#lambda_camada1 = trial.suggest_float('lambda_camada1', 0.0, 0.45)
#lambda_camada2 = trial.suggest_float('lambda_camada2', 0.0, 0.45)
#lambda_camada3 = trial.suggest_float('lambda_camada3', 0.0, 0.45)
epocas = trial.suggest_int('epocas', 10, 50)

# Treinando o modelo de rede neural
n      = X_est.shape[1]
modelo = Sequential(
[
tf.keras.Input(shape=(n,)),    # especificando a dimensão do vetor de entrada
Dense(neuronios_camada1, activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.0)), # note o termo de regulariz
Dense(neuronios_camada2, activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.0)), # note o termo de regulariz
Dense(neuronios_camada3, activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.0)),
Dense(4, activation="softmax") # Essa forma aqui é mais robusta contra erros
de arredondamente, entretanto, quando usamos linear aqui precisamos usar
BinaryCrossentropy(from_logits=True)
], name = "meu_modelo"   # Na definição do nome do modelo,
não pode haver espaços em branco
)

modelo.compile(
loss=tf.keras.losses.SparseCategoricalCrossentropy(),
optimizer=tf.keras.optimizers.Adam(0.001),
)

historico = modelo.fit(
X_est_scaled,y_est,
epochs=epocas
)

# Fazendo predições no conjunto de validação e calculando a precisão
```

```
logit       = modelo(X_val_scaled)
Previsao    = np.argmax(logit, axis=1)
precisao = accuracy_score(y_val, Previsao)

return precisao

# Encontrando os melhores hiperparâmetros

estudo = optuna.create_study(direction = 'maximize')
estudo.optimize(objetivo, n_trials=100)

# Treinando a rede neural com os melhores hiperparâmetros
melhores_parametros = estudo.best_params

modelo = Sequential(
[
tf.keras.Input(shape=(n,)),
Dense(melhores_parametros['neuronios_camada1'], activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.0)),
Dense(melhores_parametros['neuronios_camada2'], activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.0)),
Dense(melhores_parametros['neuronios_camada3'], activation="relu",
kernel_regularizer=tf.keras.regularizers.l2(0.0)),
Dense(4, activation="softmax")
], name = "meu_modelo"
)

modelo.compile(
loss=tf.keras.losses.SparseCategoricalCrossentropy(),
optimizer=tf.keras.optimizers.Adam(0.001),
)

historico = modelo.fit(
X_est_scaled,y_est,
epochs=melhores_parametros['epocas']
)

# Fazendo predições no conjunto de validação e calculando a precisão
logit       = modelo(X_val_scaled)
```

```
Previsao    = np.argmax(logit, axis=1)
precisao = accuracy_score(y_val, Previsao)
print(precisao)

# Plotando a matriz de confusão

from sklearn.metrics import confusion_matrix

y_true             = y_val.reshape(-1,1)
y_pred             = Previsao
rotulos            = [0,1,2,3]
matriz_de_confusao = confusion_matrix(y_true, y_pred, labels = rotulos)

sns.heatmap(matriz_de_confusao, cmap="viridis", annot=True, fmt='d')
plt.xlabel("Rótulo Previsto")
plt.ylabel("Rótulo Verdadeiro")
#matriz = plt.gcf()
plt.show()
#matriz.savefig("matrizconfusaokeras1.pdf", bbox_inches = "tight")
```

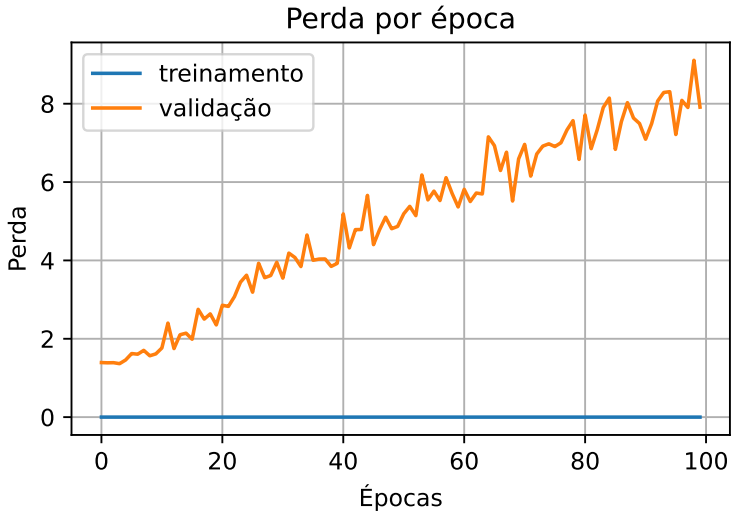
APÊNDICE 5 – GRÁFICOS DE PERDA POR ÉPOCA DO PROCESSO DE APRIMORAMENTO

FIGURA 20 – PERDA POR ÉPOCA (CAMADA 1 = 60, CAMADA 2 = 60, CAMADA 3 = 60, 20 ÉPOCAS)



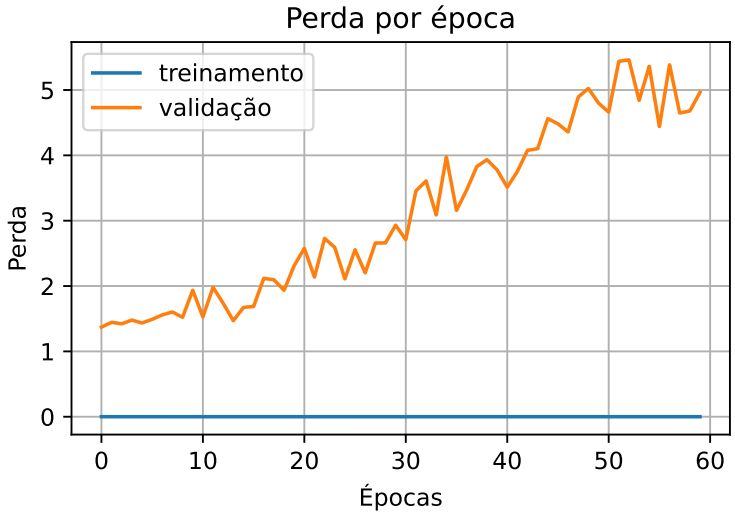
FONTE: O Autor, (2024).

FIGURA 21 – PERDA POR ÉPOCA (CAMADA 1 = 60, CAMADA 2 = 110, CAMADA 3 = 60, 100 ÉPOCAS)



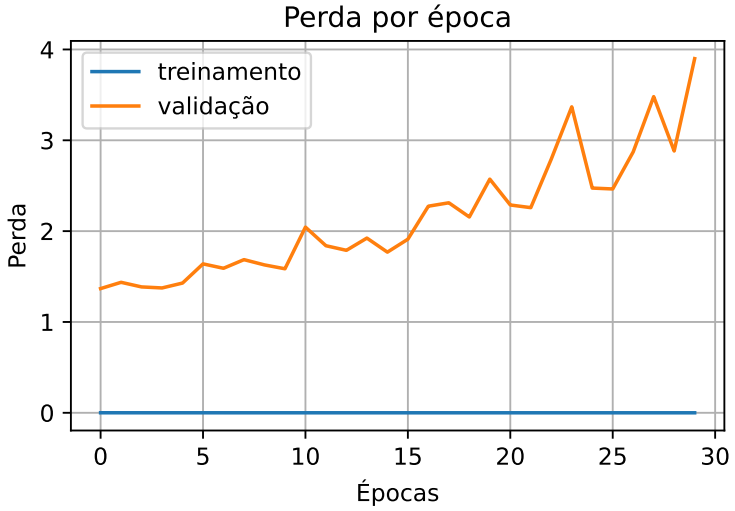
FONTE: O Autor, (2024).

FIGURA 22 – PERDA POR ÉPOCA (CAMADA 1 = 70, CAMADA 2 = 40, CAMADA 3 = 100, 60 ÉPOCAS)



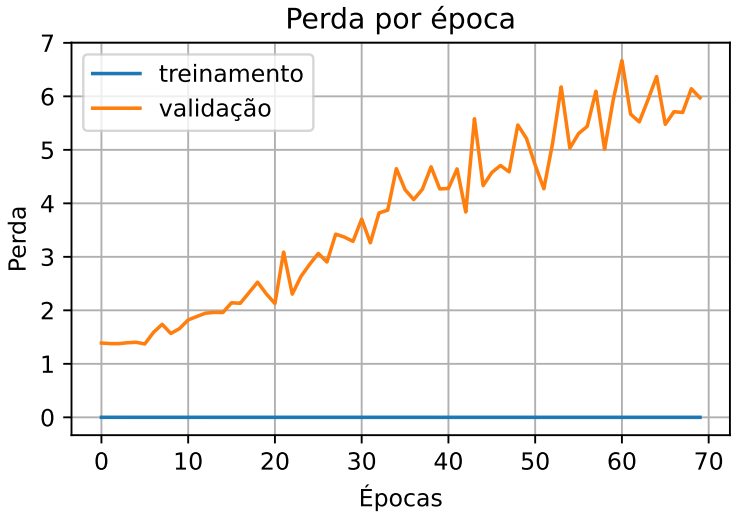
FONTE: O Autor, (2024).

FIGURA 23 – PERDA POR ÉPOCA (CAMADA 1 = 70, CAMADA 2 = 90, CAMADA 3 = 60, 30 ÉPOCAS)



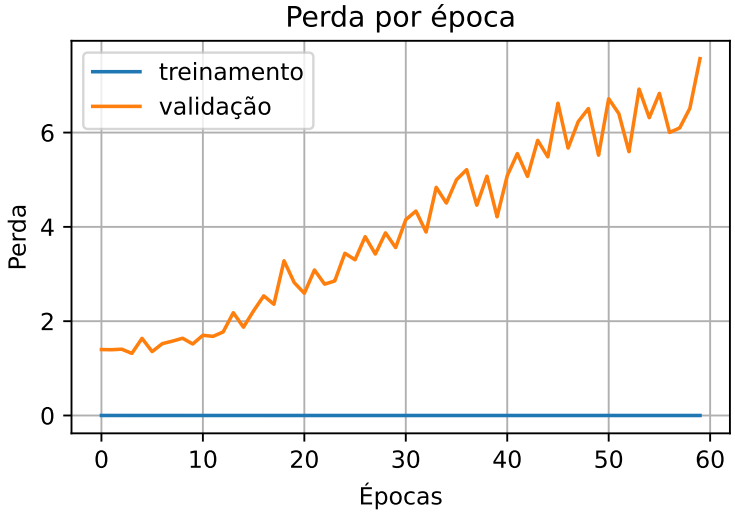
FONTE: O Autor, (2024).

FIGURA 24 – PERDA POR ÉPOCA (CAMADA 1 = 80, CAMADA 2 = 80, CAMADA 3 = 40, 70 ÉPOCAS)



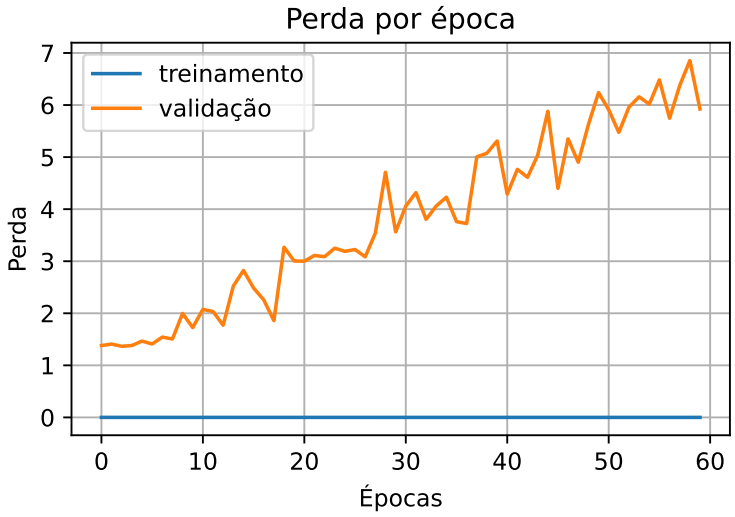
FONTE: O Autor, (2024).

FIGURA 25 – PERDA POR ÉPOCA (CAMADA 1 = 110, CAMADA 2 = 90, CAMADA 3 = 60, 60 ÉPOCAS)



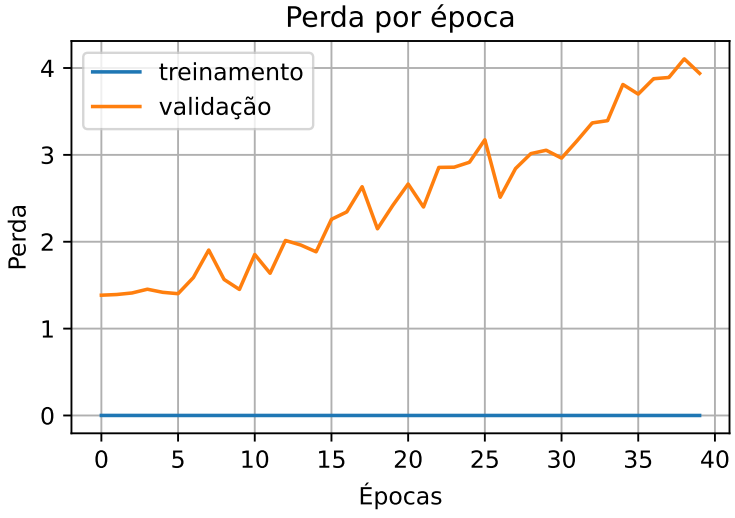
FONTE: O Autor, (2024).

FIGURA 26 – PERDA POR ÉPOCA (CAMADA 1 = 110, CAMADA 2 = 100, CAMADA 3 = 100, 60 ÉPOCAS)



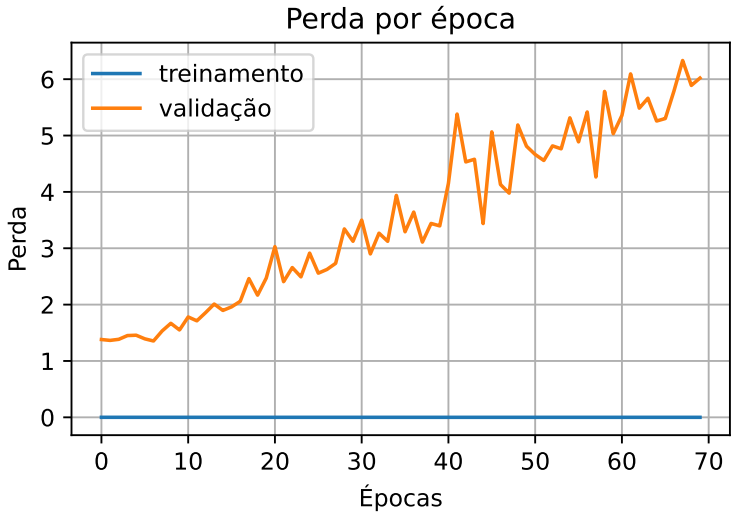
FONTE: O Autor, (2024).

FIGURA 27 – PERDA POR ÉPOCA (CAMADA 1 = 140, CAMADA 2 = 80, CAMADA 3 = 10, 40 ÉPOCAS)



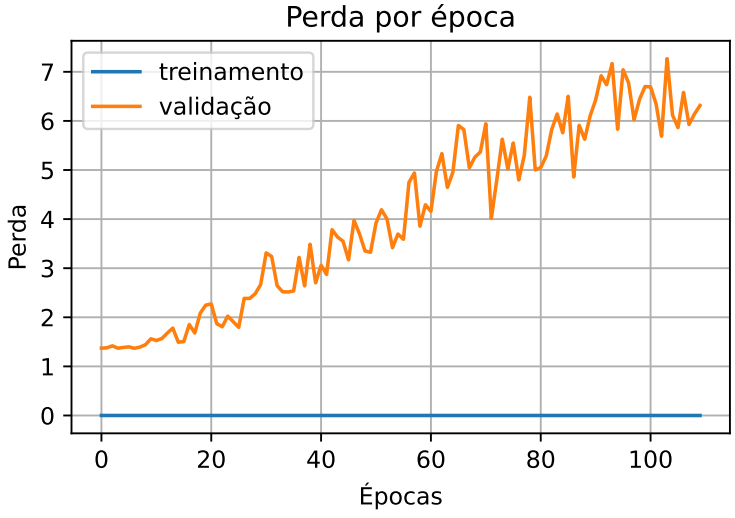
FONTE: O Autor, (2024).

FIGURA 28 – PERDA POR ÉPOCA (CAMADA 1 = 140, CAMADA 2 = 80, CAMADA 3 = 20, 70 ÉPOCAS)



FONTE: O Autor, (2024).

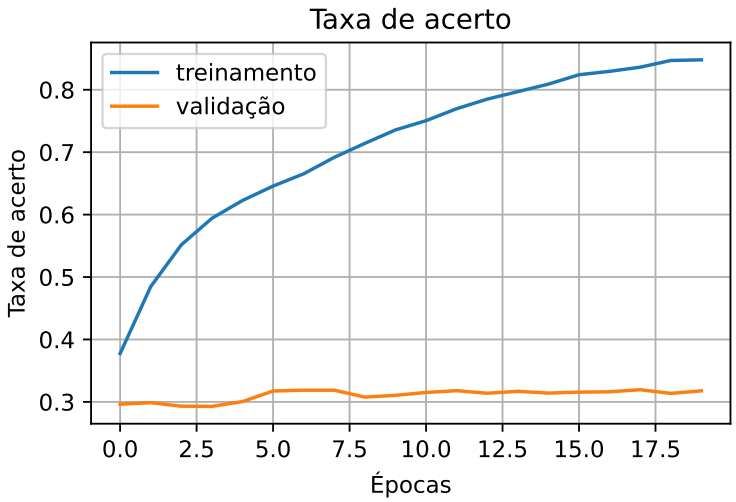
FIGURA 29 – PERDA POR ÉPOCA (CAMADA 1 = 150, CAMADA 2 = 30, CAMADA 3 = 30, 110 ÉPOCAS)



FONTE: O Autor, (2024).

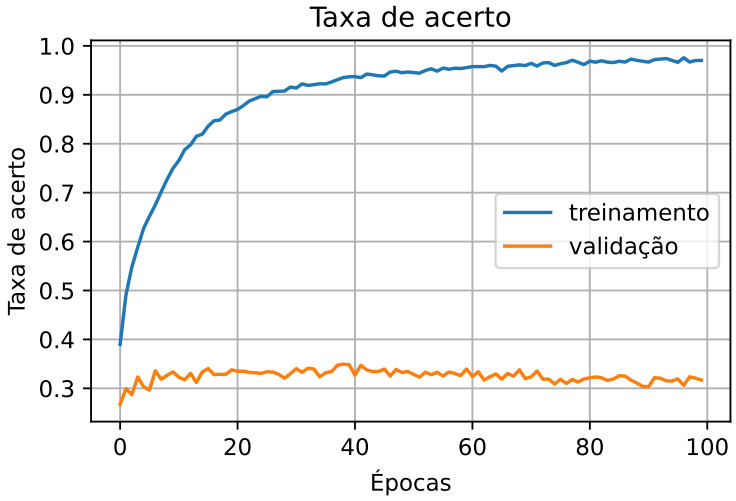
APÊNDICE 6 – GRÁFICOS DE ACERTO POR ÉPOCA DO PROCESSO DE APRIMORAMENTO

FIGURA 30 – ACERTO POR ÉPOCA (CAMADA 1 = 60, CAMADA 2 = 60, CAMADA 3 = 60, 20 ÉPOCAS)



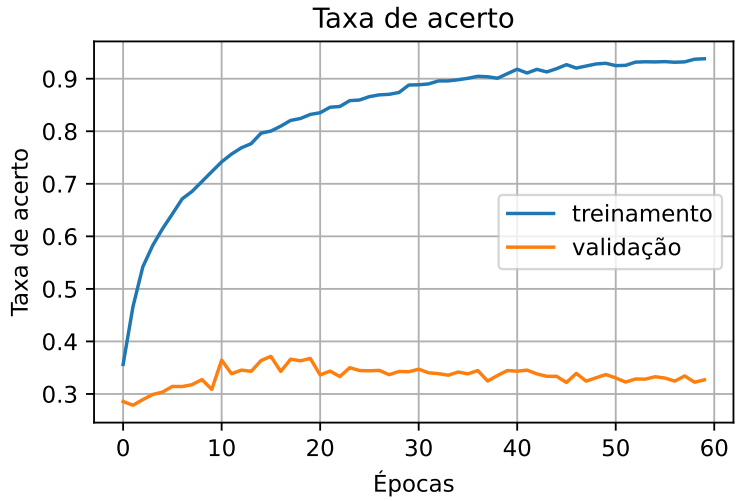
FONTE: O Autor, (2024).

FIGURA 31 – ACERTO POR ÉPOCA (CAMADA 1 = 60, CAMADA 2 = 110, CAMADA 3 = 60, 100 ÉPOCAS)



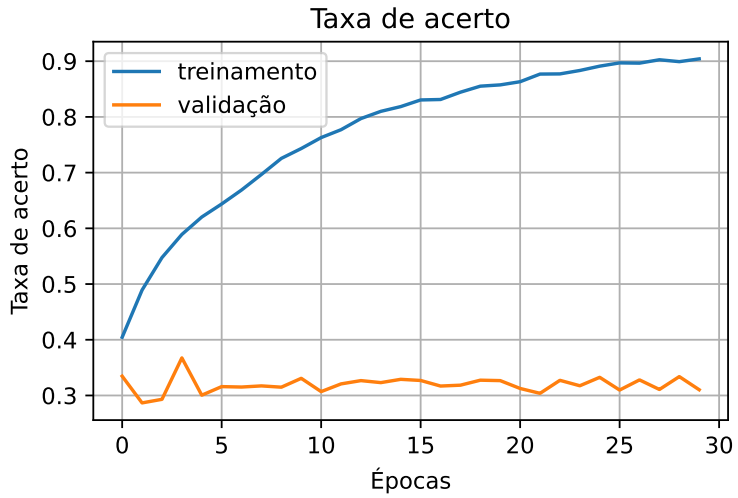
FONTE: O Autor, (2024).

FIGURA 32 – ACERTO POR ÉPOCA (CAMADA 1 = 70, CAMADA 2 = 40, CAMADA 3 = 100, 60 ÉPOCAS)



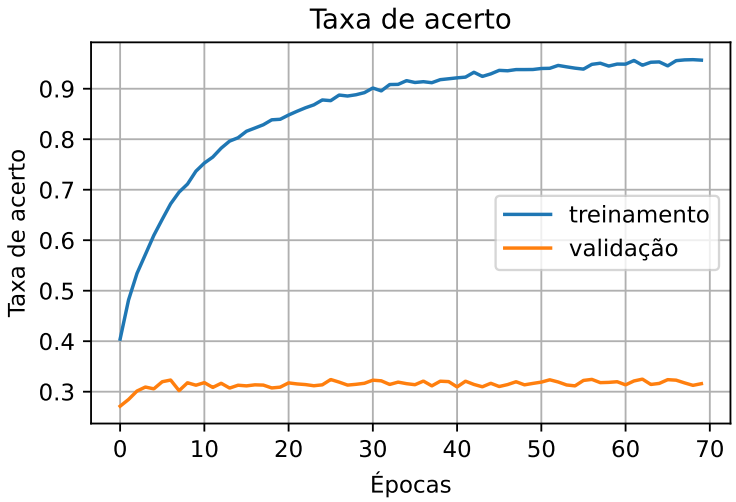
FONTE: O Autor, (2024).

FIGURA 33 – ACERTO POR ÉPOCA (CAMADA 1 = 70, CAMADA 2 = 90, CAMADA 3 = 60, 30 ÉPOCAS)



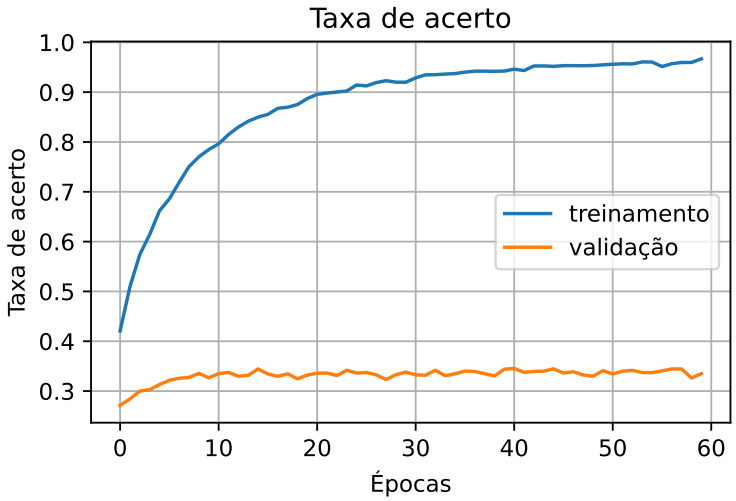
FONTE: O Autor, (2024).

FIGURA 34 – ACERTO POR ÉPOCA (CAMADA 1 = 80, CAMADA 2 = 80, CAMADA 3 = 40, 70 ÉPOCAS)



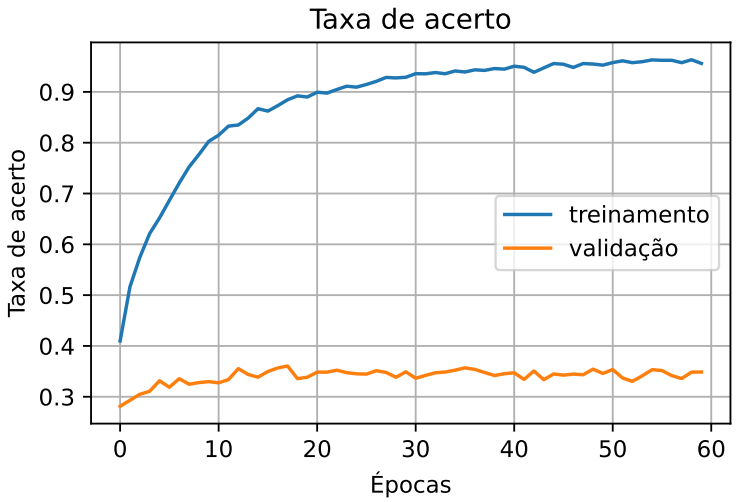
FONTE: O Autor, (2024).

FIGURA 35 – ACERTO POR ÉPOCA (CAMADA 1 = 110, CAMADA 2 = 90, CAMADA 3 = 60, 60 ÉPOCAS)



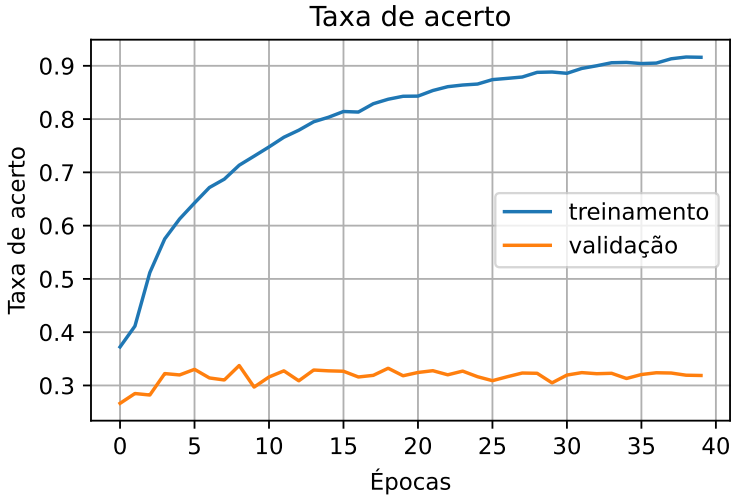
FONTE: O Autor, (2024).

FIGURA 36 – ACERTO POR ÉPOCA (CAMADA 1 = 110, CAMADA 2 = 100, CAMADA 3 = 100, 60 ÉPOCAS)



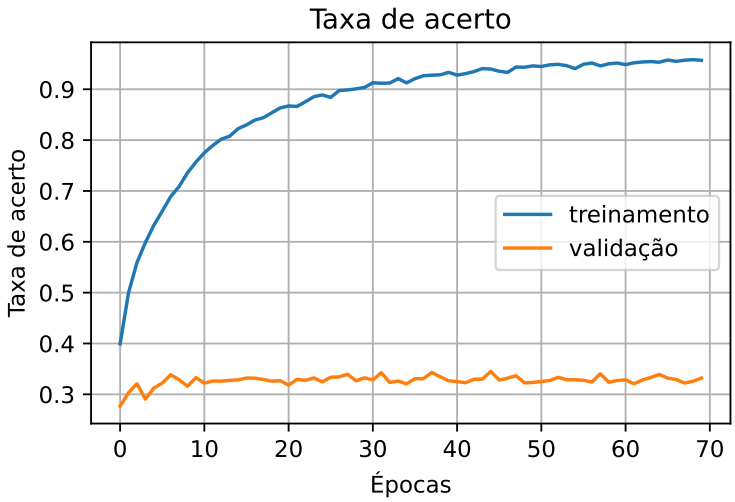
FONTE: O Autor, (2024).

FIGURA 37 – ACERTO POR ÉPOCA (CAMADA 1 = 140, CAMADA 2 = 80, CAMADA 3 = 10, 40 ÉPOCAS)



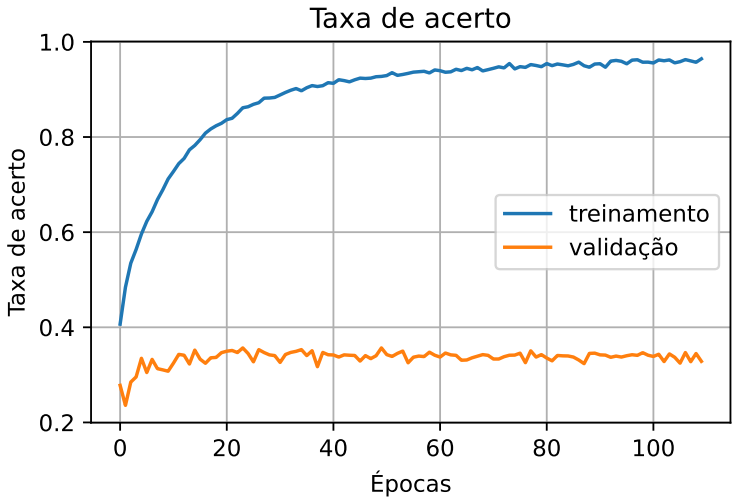
FONTE: O Autor, (2024).

FIGURA 38 – ACERTO POR ÉPOCA (CAMADA 1 = 140, CAMADA 2 = 80, CAMADA 3 = 20, 70 ÉPOCAS)



FONTE: O Autor, (2024).

FIGURA 39 – ACERTO POR ÉPOCA (CAMADA 1 = 150, CAMADA 2 = 30, CAMADA 3 = 30, 110 ÉPOCAS)



FONTE: O Autor, (2024).