REDES NEURAIS PROFUNDAS MULTITAREFAS APLICADAS À IDENTIFICAÇÃO DA LOCALIZAÇÃO E DO TAMANHO DE RUPTURAS EM CASO DE ACIDENTES DE PERDA DE REFRIGERANTE EM UMA PLANTA NUCLEAR

Filipe Santana Moreira do Desterro

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Nuclear, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Nuclear.

Orientadores: Roberto Schirru
Claudio Marcio do Nascimento Abreu Pereira

Rio de Janeiro
Março de 2024

REDES NEURAIS PROFUNDAS MULTITAREFAS APLICADAS À IDENTIFICAÇÃO DA LOCALIZAÇÃO E DO TAMANHO DE RUPTURAS EM CASO DE ACIDENTES DE PERDA DE REFRIGERANTE EM UMA PLANTA NUCLEAR

Filipe Santana Moreira do Desterro

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA NUCLEAR.

Orientadores: Roberto Schirru
Cláudio Márcio do Nascimento Abreu Pereira

Aprovada por: Prof. Roberto Schirru
Prof. Cláudio Márcio do Nascimento Abreu Pereira
Prof. Alan Miranda Monteiro de Lima
Prof. Andressa dos Santos Nicolau
Dr. Paulo Victor Rodrigues de Carvalho
Dr. César Marques Salgado

RIO DE JANEIRO, RJ - BRASIL
MARÇO DE 2024

Desterro, Filipe Santana Moreira do

Redes Neurais Profundas Multitarefas Aplicadas à Identificação da Localização e do Tamanho De Rupturas em Caso de Acidentes de Perda de Refrigerante em Uma Planta Nuclear/ Filipe Santana Moreira do Desterro. – Rio de Janeiro: UFRJ/COPPE, 2024.

XII, 87 p.: il.; 29,7 cm.

Orientadores: Roberto Schirru
Cláudio Márcio do Nascimento Abreu Pereira

Tese (doutorado) – UFRJ/ COPPE/ Programa de Engenharia Nuclear, 2024.

Referências Bibliográficas: p.80-86.

1. Identificação do LOCA. 2. Rede neural multitarefas. 3. Seleção de variáveis importantes. I. Schirru, Roberto *et al*. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Nuclear. III. Título.

DEDICATÓRIA

Deus, a quem dedico este Tese, obrigado por não inventar o botão 'soneca' para a minha consciência, me fazendo levantar e trabalhar quando tudo que eu queria era procrastinar. Esta dedicatória é um lembrete de que, mesmo nos momentos de dúvida, sua presença foi a certeza que eu precisava.

Autor desconhecido.

AGRADECIMENTOS

Primeiramente a Deus que foi meu maior orientador durante todo o curso.

Ao professor e orientador Roberto Schirru pela oportunidade concedida e por sua orientação.

Ao professor e orientador Claudio Marcio pela confiança depositada em meu trabalho desde a graduação e sua amizade ao longo desta jornada.

Ao meu grande amigo Marcelo Carvalho que meu auxiliou em grande parte desta caminhada.

A minha querida esposa Ana Carolina que vem me acompanhando nas minhas aventuras e desventuras durante todos esses anos.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

REDES NEURAIS PROFUNDAS MULTITAREFAS APLICADAS À IDENTIFICAÇÃO DA LOCALIZAÇÃO E DO TAMANHO DE RUPTURAS EM CASO DE ACIDENTES DE PERDA DE REFRIGERANTE EM UMA PLANTA NUCLEAR

Filipe Santana Moreira do Desterro

Março/2024

Orientadores: Roberto Schirru
Cláudio Márcio do Nascimento Abreu Pereira

Programa: Engenharia Nuclear

Neste trabalho, técnicas de redes neurais artificiais (RNA) são utilizadas para caracterização de acidentes de perda de refrigerante (LOCA) no sistema primário de uma usina nuclear. Para tal, dois problemas são considerados: i) a determinação da dimensão da ruptura (seu diâmetro equivalente) e ii) a identificação da localização desta ruptura. Este trabalho teve como objetivo construir, desenvolver e avaliar o desempenho de diversas arquiteturas de RNN que possam solucionar os dois problemas concomitantemente, resultando assim em um modelo que tenha a característica de resolver os problemas de identificação da localização e determinação da dimensão de uma ruptura de forma integrada, propiciando um diagnóstico mais realístico. Arquiteturas de redes neurais profundas multitarefas (*multitask deep neural network* - MTDNN) são avaliadas e comparadas com outras arquiteturas de RNA que utilizam poucas camadas ocultas e funções de ativações como Sigmóide e ReLU. Estas técnicas são amplamente utilizadas pela comunidade científica sendo encontradas na literatura. Os resultados obtidos exibiram desempenhos superiores quando comparados a outras abordagens propostas na literatura.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

DEEP MULTITASK NEURAL NETWORKS APPLIED TO IDENTIFYING THE LOCATION AND SIZE OF RUPTURES IN CASE OF REFRIGERANT LOSS ACCIDENTS IN A NUCLEAR PLANT

Filipe Santana Moreira do Desterro

March/2024

Advisors: Roberto Schirru
Cláudio Márcio do Nascimento Abreu Pereira

Department: Nuclear Engineering

In this work, techniques of Artificial Neural Networks (ANNs) are employed for characterizing Loss of Coolant Accidents (LOCA) in the primary system of a nuclear power plant. To do so, two issues are considered: i) determining the rupture size (its equivalent diameter), and ii) identifying the location of this rupture. The objective of this work was to construct, develop, and evaluate the performance of various ANN architectures capable of solving both problems simultaneously, resulting in a model that possesses the characteristic of addressing the location identification and rupture size determination issues in an integrated manner, thus enabling a more realistic diagnosis. Multitask Deep Neural Network (MTDNN) architectures are assessed and compared with other ANN architectures that employ few hidden layers and activation functions such as Sigmoid and ReLU. These techniques are widely used in the scientific community and can be found in the literature. The obtained results exhibited superior performance when compared to other approaches proposed in the literature.

Sumário

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

ÍNDICE DE SÍMBOLOS E ABREVIAÇÕES

CFNN *Cascaded fuzzy Neural Network*

CMC	Conjunto Mínimo de Centróides
CNAAA	Central Nuclear Almirante Álvaro Alberto
DRNN	*Deep Rectifier Neural Network*
DT	Árvores de Decisão
FNN	*Fuzzy-Neural Network*
ILR	Identificação da Localização da Tuptura
ITF	Identificação do Tamanho da Ruptura
LOCA	*Loss of Coolant Accident*
MAE	Erro Absoluto Médio
MLP	*Multilayer Perceptron*
MSE	*Erro Quadrático Médio*
MTDNN	*multitask deep neural network*
PDA	Diagnóstico de Acidentes em Usinas Nucleares
PNN	*Probabilistic Neural Network*, *Probabilistic Neural Network*
PSO	*Particle Swarm Optimization*
QEA	Algoritmo de Inspiração Quântica
ReLU	*Rectified Linear Unit*
RNA	Redes Neurais Artificiais
SGD	*Stochastic Gradient Descent*
TMI	*Three Mile Island*

CAPÍTULO 1. INTRODUÇÃO

1.1 APRESENTAÇÃO GERAL DO PROBLEMA

Usinas nucleares são complexos sistemas eletromecânicos monitorados contínua e ininterruptamente durante sua operação para que se mantenham elevados níveis de disponibilidade, confiabilidade e segurança. Ao se detectar algum evento anormal na usina, os operadores devem diagnosticar o ocorrido através de informações colhidas por sensores e equipamentos. Entretanto, esta tarefa pode tornar-se complexa e demorada diante do imenso número de informações exibidas aos operadores, podendo até mesmo causar interpretação errônea, devido a padrões complexos de excursões temporais das inúmeras variáveis monitoradas.

Por conta disso, sistemas computacionais de apoio ao diagnóstico de eventos anormais são de extrema importância para dar suporte a tomada de decisão a equipes de operadores (MÓL et al., 2003). Devido à grande relevância do tema, investigações científicas e desenvolvimentos tecnológicos vêm sendo continuamente evoluídos para que os sistemas possam ganhar maior robustez e eficiência no apoio ao diagnóstico de anomalias e acidentes na operação da usina. A exemplo disso, vêm se destacando modelos baseados em Inteligência Artificial, como Redes Neurais Artificiais (RNA) (BARTLETT, 1992; BASU et al., 1994; BARTAL et al., 1995; SANTOS et al., 2019, PINHEIRO et al., 2019, DESTERRO et al., 2020), Programação Genética (GP) (PINHEIRO, 2019), Computação Quântica (NICOLAU et al., 2011), Algoritmo Genético (ZHOU et al., 2000; ALMEIDA et al., 2001; VICTOR et al., 2019) entre outros.

Existe uma considerável quantidade de trabalhos focados em diagnosticar eventos anormais ou acidentes de uma forma geral, através de um rótulo ou classe, como por exemplo: i) “Blackout”, ii) “Grande LOCA”, iii) “Pequeno LOCA”, entre diversos outros (BARTLETT (1993), ALVARENGA et al., (1987), SANTOS et al., (2019), etc.). Entretanto, uma caracterização mais específica ou mesmo quantitativa do acidente pode ser de fundamental importância na avaliação de seu impacto e tomada de decisão.

O diagnóstico de acidentes mais refinado torna-se uma tarefa mais complexa, ainda pouco explorada em casos práticos reais. Buscando este tipo de refinamento e considerando que um acidente de perda de refrigerante do primário (LOCA) é um dos mais importantes com relação às suas possíveis consequências, o presente trabalho busca o desenvolvimento de um modelo/sistema capaz de caracterizar dois aspectos de um LOCA:

i) a área equivalente da ruptura;
ii) a localização da ruptura.

Para trazer mais robustez aos sistemas de identificação e diagnóstico de eventos anormais, o empenho no desenvolvimento de novas pesquisas é essencial na busca de melhorar a eficiência das usinas nucleares, aumentando a segurança e proteção, enquanto as usinas também são atualizadas. Entretanto, essas ferramentas não contemplam a identificação de detalhes mais apurados em situações anormais dentro de uma usina nuclear, onde a equipe precisaria realizar uma inspeção para realizar a coleta destes detalhes, como por exemplo identificar o tamanho de uma ruptura. Os problemas de identificação do tamanho da ruptura (ITR) em caso de perda de refrigerante e a identificação da localização da ruptura (ILR), são variantes do acidente de perda de refrigerante (*Loss of Coolant Accident* – LOCA) e identificá-los se torna mais difícil diante da complexidade do acidente e da própria usina nuclear em relação ao grande número de variáveis a serem monitoradas.

O ITR é um problema intrinsecamente não linear, pois requer a determinação da extensão da ruptura em um complexo sistema de tubulações de uma usina nuclear. Além disso, o ITR pode variar dependendo de diversos fatores, como pressão, temperatura, composição do refrigerante e condições operacionais. Portanto, encontrar uma solução precisa para o ITR é crucial para a segurança e operação eficaz de uma usina nuclear. As rupturas costumam ser classificadas segundo sua área equivalente e, neste trabalho, são considerados as seguintes categorias a partir do manual de operação da Central Nuclear Almirante Álvaro Alberto (CNAAA):

- Mini Rupturas: áreas de ruptura menores que 3 cm^2;
- Pequenas Rupturas: Áreas de rupturas de 3 até 50 cm^2;
- Médias Rupturas: Áreas de rupturas de 50 até 1100 cm^2;
- Grandes Rupturas: Áreas de ruptura maiores que 1100 cm^2.

O ILR, em particular, é um problema de classificação que pode ser ainda mais desafiador do que o ITR. Isso ocorre porque uma ruptura de determinado tamanho '*x*' em uma localização '*j*' pode ter efeitos semelhantes aos de outra ruptura em um ponto diferente da usina, com um tamanho de ruptura maior ou menor. Isso ressalta a complexidade das interações no sistema e a necessidade de técnicas avançadas para uma identificação precisa e eficaz.

1.2 TRABALHOS RELACIONADOS

Nesta seção, é apresentada uma revisão dos trabalhos relacionados com esta tese. Inicialmente são citados alguns dos importantes trabalhos sobre diagnósticos de acidentes e posteriormente trabalhos utilizando técnicas de IA para análise de séries temporais de variáveis monitoradas em usinas nucleares.

Em 1998, PEREIRA et al., (1998) desenvolveu um método para identificação de transientes usando algoritmos genéticos para otimizar um sistema de classificação baseado em medidas da distância euclidiana. O método denominado pelos autores de Conjunto Mínimo de Centróides (CMC), é utilizar o algoritmo genético para dividir o espaço de busca, de modo a encontrar subconjuntos que representem as classes.

Em 2000, ALMEIDA (2010) aperfeiçoou o método desenvolvido por PEREIRA substituindo parte do método que utiliza as medidas de distância euclidianas por outra medida utilizando conjuntos nebulosos. Este método trás o critério de pertinências possibilísticas que

permite uma avaliação das zonas dos centróides, assim estabelecendo um limiar para classificações dos transientes e classificações “não sei”.

Em 2005, MEDEIROS (2005) propôs um sistema de diagnóstico que categoriza transientes com base nas características de 17 variáveis de estado de 3 transientes nucleares, utilizando Enxames de Partículas como uma ferramenta de otimização. Neste sistema, é calculada a distância euclidiana entre o conjunto de variáveis do evento em um determinado momento e o centróide das variáveis do transiente de referência. O Enxame de Partículas é usado com o propósito de identificar a melhor posição dos centróides (representados pelos vetores de Voronoi) dos transientes de referência, resultando significativamente a precisão das classificações.

Em 2010, NICOLAU (2010) desenvolveu um método para identificar transientes em uma usina nuclear. Utilizando Algoritmos de Inspiração Quântica e Inteligência de Enxames com objetivo de classificar eventos anormais dentro de três cenários de acidentes nucleares. Assim o Algoritmo de Inspiração Quântica foi utilizado para determinar a posição ótima dos centróides representativos de cada transiente, buscando maximizar a precisão das classificações.

Em 2014, NICOLAU (2014) explorou a implementação do Algoritmo de Inspiração Quântica (QEA) em duas áreas: Identificação e Diagnóstico de Acidentes em Usinas Nucleares (PDA) e Recarga do Combustível Nuclear. A aplicação do QEA no contexto do PDA foi elaborada para aprimorar as investigações anteriores de NICOLAU (2010). O objetivo era desenvolver um método de classificação de acidentes que pudesse lidar com cenários desafiadores, incluindo situações em que a resposta "Não Sei" fosse apropriada, especialmente para eventos desconhecidos, sem depender de um evento iniciador predefinido.

A partir da década de 90 importantes trabalhos focados no diagnóstico de transientes em plantas nucleares foram desenvolvidos utilizando técnica de redes neurais artificiais (RNA). Em 1993, BARTLETT (1993) desenvolveu uma ferramenta baseada em RNA, onde alcançava a identificação de 7 cenários de acidentes da planta nuclear.

Em 1995, BARTAL et al., (1995) aprimorou o trabalho de Bartlett, utilizando um modelo de rede neural probabilística (*Probabilistic Neural Network* - PNN). Neste trabalho foi adicionado ao classificador um modo de responder "Não Sei" para cenários desconhecido pelo modelo. Este modelo apresentou bom desempenho na identificação de 13 cenários de acidentes da planta nuclear.

Em 1997, ALVARENGA et al., (1997) apresentou um trabalho que demonstrou o uso de RNA, lógica nebulosa e algoritmo genético, trabalhando em conjunto na identificação de 16 cenários de acidentes na planta nuclear Angra 2.

Em 2003, MÓL et al., 2003 apresentou o trabalho com RNA em estruturas de multicamadas onde trouxe a classificação de cinco eventos anormais dentro de uma planta de usina nuclear. Este trabalho também incluiu um procedimento de diagnósticos "Não sei", para eventos fora do treinamento, a fim de evitar classificações errôneas.

Em 2009, COSTA (2009) apresentou um trabalho de mestrado utilizando modelos de RNA para identificar qual o melhor procedimento executar na ocorrência de um transiente na planta nuclear de Angra 2.

Em 2019, SANTOS et al., (2019) estendeu o trabalho de MÓL et al., (2003), utilizando Redes Neurais Retificadas Profundas (*Deep Rectifier Neural Network* - DRNN), onde o número de camadas é bem superior ao da arquitetura de MÓL em 2003. Foram usados 12 eventos anormais neste trabalho.

Em 2020, PINHEIRO et al., (2020) demonstrou a capacidade de sistemas inteligentes responder "Não sei" utilizando uma arquitetura de Rede Neural AutoEncoder. Esse modelo demonstrou desempenhos superiores em testes de classificações de eventos anormais desconhecidos.

Em 2004, MAN GYUN NA et al., (2004), apresentou dois modelos para resolver os problemas ITR e ILR de forma separada. Para resolver o ILR ele utilizou uma arquitetura de rede neural probabilística (*Probabilistic Neural Network* - PNN). Já para o ITR foi utilizado uma

arquitetura de rede neural difusa (*Fuzzy-Neural Network* - FNN). A arquitetura neste trabalho desenvolvida tinha o objetivo de identificar a localização de 3 áreas da usina e identificar o tamanho de ruptura variando de 0 a 10 cm2 de área.

Outro trabalho em 2017, apresentado por GEON PIL CHOI (2017), aprimorando o trabalho desenvolvido por MAN GYUN NA (2004), utilizou uma arquitetura rede neural difusa em cascata *(Cascaded fuzzy neural network* - CFNN). Este trabalho aumentou a precisão em casos de ITR, conseguindo identificar áreas de ruptura de até um metro quadrado.

Em 2019, MAHDI et al., (2019), desenvolveu uma arquitetura baseada em Redes Neurais NARX para predizer qual seria o tamanho de uma ruptura caso ocorresse um LOCA na planta nuclear. Este trabalho utilizou somente uma variável para treinar a rede neural.

Em 2021, TING-HAN LIN et al., (2021) desenvolveu um sistema baseado em árvores de decisão (DT) e RNN para prever o tamanho e a localização de um acidente de perda de refrigerante (LOCA). O conjunto de dados foi baseado em simulações realizadas a partir de 18 localizações diferentes e três tipos diferentes de rupturas (LOCA, MSLB e SGTR). Nos casos SGTR, os tamanhos das rupturas foram definidos entre 5 cm² e 50 cm², enquanto nas outras simulações os tamanhos das rupturas variaram de 5 cm² a 2000 cm², com variações de 2 cm^2 até 20 cm^2 e o restante dos tamanhos com variações de100 cm^2 até 2000 cm². No total, foram realizadas 336 simulações para gerar dados de treinamento e 550 foram usadas para validar o modelo. Neste trabalho, os dados são submetidos a duas fases diferentes para prever a localização e, em seguida, o tamanho da ruptura. Na primeira fase, o método MUSIC (J. Lin et al., 2006) é dedicado a determinar a localização da ruptura. Uma vez que a localização é determinada, na segunda fase, os dados são alimentados em um modelo de rede neural convolucional para prever o tamanho. O método MUSIC alcançou uma taxa de sucesso de pelo menos 96,67% para a localização da ruptura. Na previsão do tamanho, é mencionado que em 95% dos dados avaliados, o erro relativo foi inferior a 8%.

Em 2023, SANTOS et al., (2023) realizaram um estudo comparativo entre técnicas de Multilayer Perceptron (MLP) e o Algoritmo de Otimização por Enxame de Partículas (Particle

Swarm Optimization - PSO) para prever o tamanho da ruptura em caso de perda de refrigerante. O PSO alcançou uma acurácia de 0,972, ligeiramente inferior à MLP, que obteve 0,996. Embora o PSO tenha uma acurácia menor, não utilizou todas as variáveis disponíveis. Concluindo, portanto, que para cenários com poucos dados disponíveis, o PSO pode proporcionar uma resposta melhor.

Em 2023, como parte dos resultados desta tese, DESTERRO et al., (2023) apresentaram uma abordagem para treinar redes neurais multitarefa a fim de resolver os desafios ligados à Identificação do Tamanho da Ruptura (ITR) e Identificação da Localização da Ruptura (ILR). Assim, esta tese continua o desenvolvimento desse trabalho, buscando aprimorar e expandir ainda mais as investigações apresentadas em DESTERRO et al., (2023).

1.3 OBJETIVOS

Dado o vasto conjunto de variáveis monitoradas em uma instalação nuclear, o primeiro objetivo é utilizar dados minerados por meio de técnicas de seleção de variáveis. Esta técnica é essencial para a selecionar as variáveis mais relevantes e adequadas à resolução dos problemas ITR e ILR.

O segundo objetivo deste estudo é desenvolver uma arquitetura de Rede Neural de aprendizado profundo multitarefas (*multitask deep neural network* - MTDNN) capaz de superar o problema de caracterização do LOCA com relação à identificação do tamanho da ruptura (ITR) e à localização da ruptura (ILR).

Combinando a técnica de aprendizado profundo multitarefa e a seleção de variáveis importantes, é proposto fornecer uma solução eficaz e robusta para o problema da caracterização do LOCA.

1.4 JUSTIFICATIVA

Para garantir a segurança operacional e proteger o ambiente, no caso de um LOCA uma das tarefas mais crítica é descobrir exatamente o tamanho e onde ocorre a ruptura na tubulação do primário. Com a precisão de onde a ruptura ocorreu e qual a sua dimensão, é possível oferecer respostas mais eficazes para as emergências, permitindo que as equipes da sala de controle atuem rapidamente para conter um escape do refrigerante para fora do prédio da contenção e minimize danos. No caso do LOCA com rupturas muito pequenas, ainda maior é a busca de ferramentas que respondam com a localização precisa do LOCA, pela razão destas rupturas passarem despercebidas por um longo período, evoluindo em um estado mais grave como ocorrido no acidente Three Mile Island - TMI, ocorrido em 1979.

Quando se tem a estimativa do tamanho e local da ruptura, as equipes da usina nuclear podem fazer contramedidas mais precisas para mitigar a fuga de refrigerante e prevenir a difusão de substâncias radioativas. Isso inclui desde o isolamento imediato da área afetada, como também realocando sistemas de refrigeração alternativos para esfriar o reator, caso os de emergência venham falhar, e até mesmo a construção de barreiras a fim de controlar o início da dispersão de materiais radioativos.

Além disso, a uma estimativa mais precisa do tamanho da ruptura permite uma avaliação mais precisa dos riscos associados, tanto para o pessoal da usina quanto para o público nas proximidades. Isto facilita tomadas de decisões para rotas de evacuação e outras medidas de proteção que são convenientes para reduzir exposição à radiação ou outro risco.

A fim de tratar o problema do LOCA, estudos com o treinamento de RNA tem demonstrado resultados significativos na resolução de problemas. Os problemas de ITR e ILR em específico são tratados na literatura e em outros estudos (MAN GYUN NA (2004), GEON PIL CHOI (2017), MAHDI (2019)). Estes trabalhos utilizam técnicas de redes neurais diferentes, mas com uma característica em comum, arquiteturas com poucas camadas ocultas, sendo assim estes modelos

possuem baixa capacidade de alcançar melhores resultados à medida que a dificuldade dos problemas aumentam.

MAN GYUN NA (2004) e TING-HAN LIN et al., (2021) utilizaram um algoritmo para inicialmente prever a localização do LOCA e, em seguida, utilizaram essa informação como entrada para uma RNA a fim de prever o tamanho da ruptura. Observando o aumento na eficiência ao resolver esses problemas quando as informações sobre eles são conhecidas, surgiu a hipótese de desenvolver uma arquitetura única de rede neural multitarefa, que permite ao modelo produzir e compartilhar informações entre os problemas, a fim de aumentar a capacidade do modelo de fornecer informações mais precisas sobre o ITR e ILF, tornando-o um sistema mais realista.

Um outro aspecto a se destacar é que estes trabalhos não apresentam um critério de seleção das variáveis relevantes, dentro do universo de uma planta nuclear, que possam solucionar os problemas de ITR e ILR de forma mais eficiente, como é apontado por MAHDI (2019) em seu trabalho.

A partir de bons resultados apresentados em trabalhos que utilizam técnicas de RNA, vislumbrou-se que uma arquitetura de rede neural profunda com multitarefas, associada a técnicas de mineração de dados, poderia oferecer ganhos com solução integrada dos problemas ITR e ILR, gerando assim grande impacto na tomada de decisão pelas equipes no controle da usina e tornando os sistemas da usina mais robustos.

1.5 INOVAÇÃO E RELEVÂNCIA

Trabalhos anteriores na área nuclear, como os estudos de PINHEIRO et al., (2020), SANTOS et al., (2020) e DESTERRO et al., (2020), exploraram o uso de técnicas de redes neurais artificiais com aprendizado profundo. Essas pesquisas pioneiras serviram como motivação para o início deste estudo.

No âmbito deste trabalho, foram desenvolvidos modelos de MTDDN com o propósito de abordar o problema de identificar a localização e o tamanho de rupturas em de uma planta nuclear

de forma integrada. Notavelmente, não foram encontradas soluções equivalentes na literatura que apresentem e abordem de forma integrada os dois problemas (ITR e ILR), como as técnicas investigadas neste estudo.

A relevância deste trabalho se destaca na demonstração da eficácia dos modelos de redes neurais multitarefas com aprendizado profundo que foram desenvolvidos. Esses modelos apresentam soluções sólidas para problemas complexos no campo da energia nuclear. Além disso, os estudos relacionados à seleção de variáveis importantes também produziram bons resultados, contribuindo para a compreensão e resolução eficiente desses desafios críticos.

CAPÍTULO 2. FUNDAMENTAÇÃO TEÓRICA

2.1 REDES NEURAIS ARTIFICIAIS

RNAs são modelos computacionais que buscam imitar o funcionamento do cérebro humano, simulando o aprendizado de padrões através do treinamento. Seus primeiros experimentos foram iniciados por volta da metade do século XX, pelo psiquiatra e neuroanatomista Warren McCuIloch e pelo matemático Walter Pitts em 1943 (MCCULOCH et al., 1943), que conceberam o primeiro modelo artificial de um neurônio biológico. Nesse modelo inicial, as suas respostas resolviam problemas binários (zero ou um).

Quinze anos mais tarde, em 1958, ROSENBLATT et al., (1958) apresentou um modelo de aprendizado chamado *Perceptron*. Esse método incluiu o conceito de camada de neurônios à arquitetura de redes neurais e demonstrou a possibilidade de se obter bons resultados no reconhecimento de padrões com a arquitetura *Perceptron*. Minsky e Papert (MINSKY et al., 1969) incluíram o conceito de camadas ocultas nas arquiteturas de redes neurais artificiais criando o modelo de *Multilayer Perceptron* (MLP).

Em 1986, RUMELHART et al., (1986) desenvolveu um algoritmo de treinamento baseado na retropropagação do erro, conhecido como "*backpropagation*". Com esse algoritmo foi possível treinar arquiteturas MLPs de forma sistemática. A solução de problemas com complexidade mais alta passou ser possível com utilização desse algoritmo de treinamento.

Apesar deste algoritmo desenvolvido por Rumelhart ter trazido melhorias significativas ao aprendizado de máquina por redes neurais, o aumento da quantidade de neurônios e camadas ocultas, em busca de melhores resultados em problemas mais complexos, esbarrava em um problema conhecido como perda de gradiente (GLOROT, 2010). O problema de perda de gradiente prejudica o aprendizado da RNA, acarretando assim resultados pouco satisfatórios. Hochreiter (1991) apresentou esta problemática de forma teórica, tendo como causa o uso de funções de

ativação do tipo sigmoide em arquiteturas de redes neurais (HOCHREITER et al., 1991). Quando esta função de ativação era empregada em arquiteturas com muitas camadas ocultas, o problema se agravava.

Durante os anos 90, a evolução das unidades de processamento gráfico (GPU), fez com que sua utilização passasse a ter um escopo mais amplo, extrapolando os limites do processamento gráfico, sendo utilizadas como um co-processadores paralelos em problemas numéricos em geral (JEZEQUEL et al., 2015), resultando em um aumento significativo no poder computacional para a solução de diversos problemas complexos. A programação paralela que utilizava os diversos núcleos de processamento contidos nas GPUs permitia que os cálculos matemáticos pudessem ser feitos de forma muito mais rápida em comparação às arquiteturas computacionais existentes. Seguindo esta tendência, os desenvolvedores de arquiteturas de redes neurais também utilizavam esse modelo de programação para treinamento das arquiteturas de rede neural artificial.

Em 2006, GEOFFREY et al., (2006) demonstrou que uma rede neural poderia ser eficientemente treinada usando uma estratégia chamada pré-treinamento não supervisionado por camadas utilizando as GPUs para acelerar o treinamento. Nos anos seguintes com o uso de GPU, os métodos de treinamento mais antigos, o aprendizado supervisionado, começou a demonstrar melhor desempenho que o pré-treinamento, com a diferença de haver arquiteturas com muito mais camadas ocultas, aumentando eficiência no treinamento de modelos mais complexos. Desta forma, essa onda de pesquisas em redes neurais popularizou o uso do termo aprendizagem profunda ou *deep learning* (GEOFFREY et al., 2006) para enfatizar que os pesquisadores agora seriam capazes de treinar redes neurais mais profundas do que antes. Apesar do treinamento ser muito mais rápido com o advento das GPUs, Geoffrey Hinton não resolveu o problema de perda de gradiente, problema este que impedia o avanço das redes neurais de forma mais impactante.

Em 2011, GLOROT et al., (2011) apresentou à comunidade científica a função de ativação ReLU como solução para o problema de desaparecimento de gradiente. A ReLU permitiu que as redes neurais com muitas camadas ocultas alcançassem a capacidade de generalizar bem os dados em modelos profundos.

O termo aprendizagem profundo, quando relacionado aos modelos de redes neurais, geram muitas discussões, sendo uma delas o número de camadas para definir uma arquitetura como profunda. O pesquisador Jürgen Schmidhuber, em 2015, sugeriu que quando uma arquitetura de rede neural tem mais que três camadas é uma arquitetura profunda, e quando tem mais de dez camadas essa é uma arquitetura muito profunda (JÜRGEN, 2015). Apesar desta discussão ainda existir, foi selecionada a priori esta definição para descrever as arquiteturas deste trabalho.

2.1.1 Arquitetura da Rede Neural Artificial Tipo Perceptron Multicamadas

O funcionamento do neurônio biológico (Figura 1) é feito da seguinte maneira: O neurônio biológico recebe sinais elétricos de entrada através de seus dendritos; O corpo da célula combina esses sinais e caso a corrente elétrica gerada seja forte o suficiente, sinais elétricos de saída serão gerados e passados através das sinapses para os dendritos de outro neurônio.

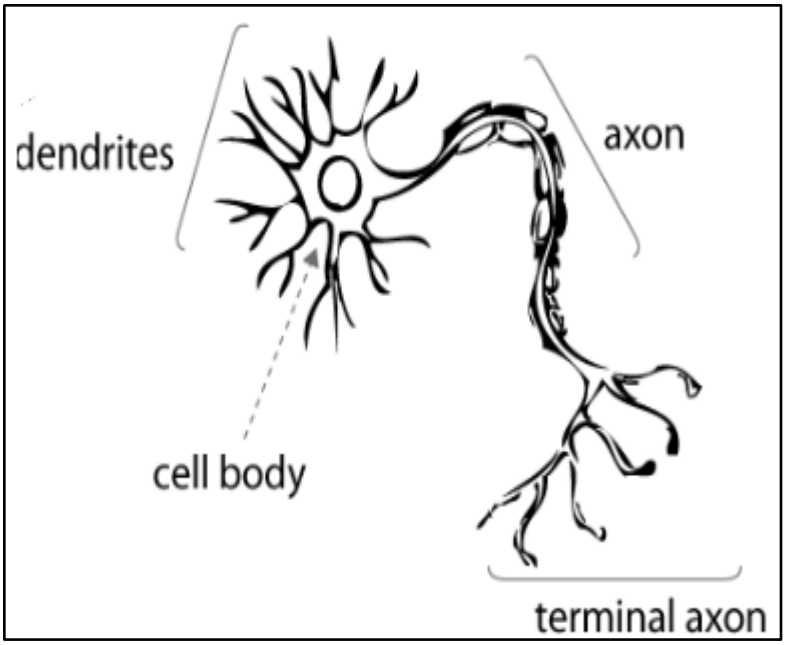
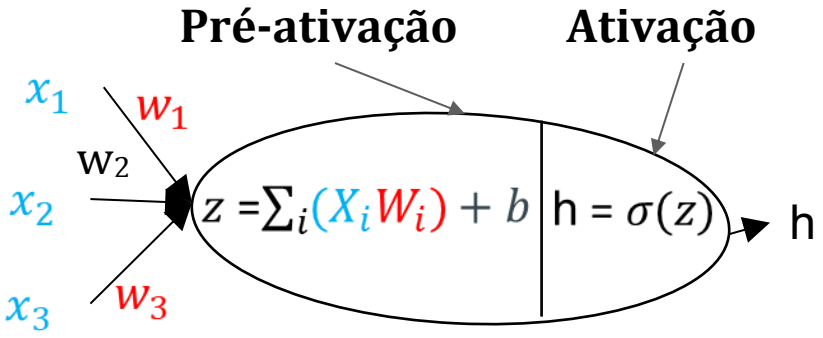


Figura 1: Representação do neurônio biológico

Na ausência de fortes sinais elétricos de entrada, os neurônios biológicos não são ativados, ou seja, quanto maior a corrente elétrica, maior a taxa de ativação do neurônio. Os neurônios artificiais (Figura 2) simulam o funcionamento dos neurônios biológicos. O comportamento é de maneira semelhante, onde o neurônio artificial computa a "soma ponderada" de suas entradas (pré-ativação), acrescentando um viés e através de uma função de ativação decide se o mesmo deve ser “ativado" ou não.



$$\text{Pré-Ativação do Neuronio} = z = \sum_i X_i W_i$$

$$\text{Ativação do Neuronio} = h = \sigma(z)$$

Figura 2: Representação do neurônio artificial (X_n: representa as entradas; W_n: representa os pesos).

O neurônio artificial é a menor estrutura dentro da arquitetura da RNA, em uma rede tipo MLP, um conjunto de neurônios forma uma estrutura nomeada de camada. Uma camada é formada por diversos neurônios não conectados entre si. Quando é criada uma estrutura com uma camada de entrada, uma ou mais camadas intermediárias (camadas ocultas), e uma camada de saída, tem-se uma arquitetura do tipo Multilayers Perceptron (MLP). Na representação da Figura 3 é demonstrado um exemplo de arquitetura MLP.

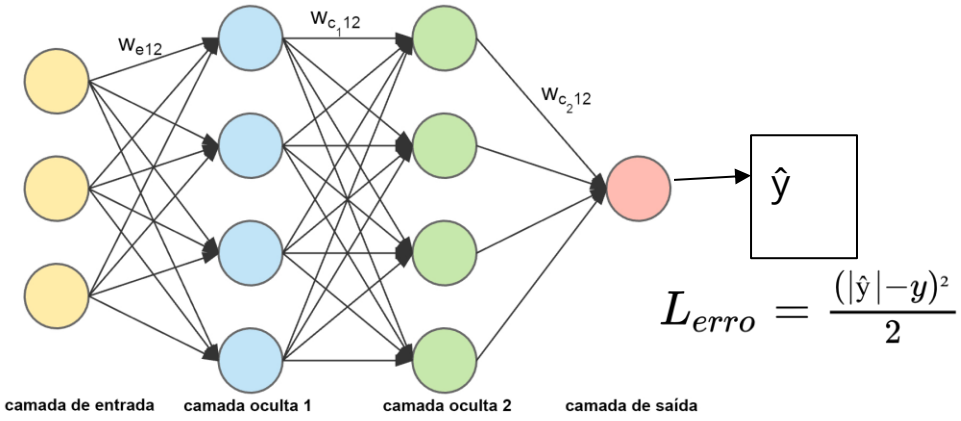


Figura 3: Representação da Arquitetura Multilayers Perceptron

Esta arquitetura funciona recebendo os sinais iniciais através da camada de entrada, propagando esses sinais para as camadas seguintes através das conexões (sinapses) entre os neurônios. Em um neurônio artificial a intensidade da sinapse é simulada por um fator de ponderação chamado peso da sinapse (W_n), ou simplesmente peso, conexões são as setas em preto na Figura 3. Ao final, depois de realizado todas as somas e multiplicações nas camadas ocultas, gera-se uma saída (resultado) na última camada $\hat{y}$. A função de erro L é utilizada para avaliar o quão bom está a saída da rede, em comparação com o valor conhecido y.

Encontrar a combinação de pesos de maneira manual em uma arquitetura de MLP onde resulte no menor erro possível é complexa de realizá-la sem um mecanismo inteligente. Uma maneira de encontrar os pesos ideais é utilizando algoritmos de otimização (RUMELHART et al., 1986) em função que o ajuste nos pesos minimize o erro da saída da RNA.

2.1.2 Função de Erro

No contexto do treinamento de Redes Neurais Artificiais (RNAs), a função de erro desempenha um papel de importante. Sua principal função é quantificar a discrepância entre as previsões geradas e os valores reais. A escolha coerente da função de erro é essencial para garantir que a RNA realize uma aprendizagem eficaz a partir dos dados de treinamento.

O Erro Quadrático Médio (MSE) calcula a média dos quadrados das diferenças entre um valor predito e os valores reais do conjunto de dados. Em um contexto real o que o MSE faz é calcular a diferença entre a saída $\hat{y}$ de uma RNA e o valor real y, depois eleva essa diferença ao quadrado, em seguida calcula a média desses quadrados. Isto representa o qual perto as prediçoes estao do valor real.

$$MSE = \frac{1}{n} \sum_{i=1n}^{n} (\hat{y}i - yi)^2 \qquad (1)$$

Outra função de erro é Erro Absoluto Médio (MAE) que avalia o quão semelhante estão as previsões em relação aos valores reais nos dados. Ao contrário do MSE, o MAE não realiza os quadrados das diferenças, ele apenas calcula a média das diferenças absolutas entre as $\hat{y}$ de uma RNA e o valor real y.

$$MAE = \frac{1}{n} \sum_{i=1n}^{n} (\hat{y}i - yi) \qquad (2)$$

A Entropia Cruzada é uma função fundamental no contexto da avaliação de classificadores. Ao contrário de métricas como MSE ou MAE, a Entropia Cruzada lida com a classe probabilística das previsões, oferecendo uma avaliação mais apropriada quando a incerteza é uma consideração importante. A Entropia Cruzada quantifica a diferença entre a distribuição de probabilidade prevista pelo modelo e a distribuição real dos dados. Cada previsão é avaliada em termos do logaritmo negativo da probabilidade atribuída à classe correta. Minimizar a Entropia Cruzada é o

objetivo, indicando que as previsões não apenas identificam corretamente a classe, mas também expressam confiança nessa predição.

$$\text{Entropia Cruzada}(y, p) = -\frac{1}{n}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j} \cdot \log(p_{i,j}) \quad (3)$$

2.1.3 Aprendizado Supervisionado

O problema central em redes neurais está em descobrir os valores dos pesos das sinapses entre os neurônios, de tal forma que a rede tenha baixo erro e alta capacidade de generalização. O processo de ajuste destes pesos permite que o modelo de RNA acerte mais, isso é também chamado de processo de aprendizado. Durante o processo de ajuste o modelo pode melhorar ou piorar definindo uma aprendizagem ruim ou boa.

Tradicionalmente tais pesos são descobertos através de algoritmos de retropropagação do erro, que utilizam métodos como Gradiente Descendente (*Stochastic Gradient Descent* – SGD). O aprendizado supervisionado utiliza os algoritmos realizando a retropropagação (*backpropagation*) do erro, utilizando uma medida de erro calculada a partir da saída da rede em relação à função objetivo, para calcular o quanto os pesos serão ajustados com objetivo de diminuir o erro.

2.1.3.1 Gradiente Descendente - SGD

O SGD (RUMELHART et al., 1986) foi o primeiro método criado para ajuste de pesos de forma interativa e tem sido usado até hoje em diversos modelos de aprendizado de máquina baseado em RNAs.

A atualização dos pesos $w^{(t+1)}$, é realizada através da equação 1, a seguir:

$$w^{(t+1)} = w^{(t)} - n \frac{\partial L}{\partial w} \tag{4}$$

Onde n é a taxa de aprendizado, usada para suavizar o incremento nos pesos durante o processo de ajuste. L é a função de erro. E $\frac{\partial L}{\partial w}$ é o gradiente local.

2.1.3.2 *Adaptive Moment Estimation* - ADAM

Outro algoritmo utilizado para ajuste dos pesos é o ADAM (DIEDERIK, 2014). Este propõe o controle da velocidade do gradiente com os termos $v_w^{(t+1)}$ e $\hat{v}_w$. Este algoritmo também utiliza outro mecanismo chamado de "momento" nos termos $m_w^{(t+1)}$ e $\hat{m}_w$. Este segundo mecanismo auxilia na aceleração dos vetores de gradientes nas direções corretas, levando assim a uma convergência mais rápida.

$$\begin{aligned} m_w^{(t+1)} &\leftarrow \beta_1 m_w^{(t)} + (1 - \beta_1) \frac{\partial L}{\partial w} L \\ v_w^{(t+1)} &\leftarrow \beta_2 v_w^{(t)} + (1 - \beta_2) \left(\frac{\partial L}{\partial w} L^{(t)}\right)^2 \end{aligned}$$

$$\hat{m}_w = \frac{m_w^{(t+1)}}{1 - \beta_1^{(t+1)}} \tag{5}$$

$$\hat{v}_w = \frac{v_w^{(t+1)}}{1 - \beta_2^{(t+1)}}$$

$$w^{(t+1)} = w^{(t)} - n \frac{\hat{m}}{\sqrt{\hat{v}} + \varepsilon}$$

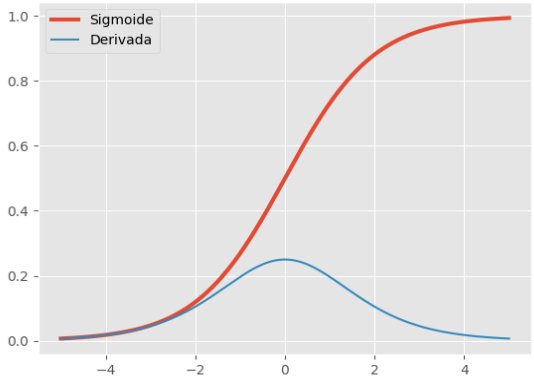
onde, ε é um pequeno escalar usado para evitar a divisão por 0, e β_1 e β_2 são os fatores usados para controlar a velocidade dos gradientes ($v_w^{(t+1)}$ e $\hat{v}_w$) e os momentos de gradientes ($m_w^{(t+1)}$ e $\hat{m}_w$), respectivamente.

2.1.4 Função de Ativação

A função de ativação desempenha um papel crítico no sucesso do algoritmo de minimização de erros em redes neurais. Ela determina como a informação flui pela rede e é essencial para a capacidade de um neurônio artificial aprender e generalizar a partir dos dados de treinamento. A função de ativação emula o processo de ativação dos neurônios biológicos e permite a aplicação de técnicas de treinamento baseadas em gradiente. Essa escolha influencia diretamente o desempenho do modelo. Portanto, a seleção cuidadosa da função de ativação é crucial para o treinamento eficaz de redes neurais.

2.1.4.1 Função de Ativação Sigmoide Logística

Esta função por muito tempo foi a função de ativação mais utilizadas em RNAs. Seu funcionamento começa quando ela recebe um valor real $\boldsymbol{x}$ como entrada e gera como saída um valor entre o intervalo 0 e 1 (Figura 4). Em particular, grandes números negativos se tornam 0 e grandes números positivos se tornam 1. Devido seu uso constante se tornou uma função tradicional nas arquiteturas de RNA.



$$\sigma(x) = \frac{1}{1 + e^{x}}$$

Figura 4: Representação da função de ativação sigmóide

A Sigmoide logística é uma função não negativa, ou seja, caso a sua entrada não seja zero, está sempre produzirá uma saída positiva. Nesta situação (Equação 6), devido $y_i > 0$, o gradiente $\partial L/\partial W_{ij}$ terá sempre o mesmo sinal que $\partial L/(\partial x_j)$. Como resultado durante a retro propagação todos os pesos W_{ij} de uma camada só podem ter seus valores aumentados ou diminuídos.

$$\frac{\partial L}{\partial wij} = \frac{\partial Xj}{\partial wij}\frac{\partial L}{\partial Xj} = yi\ \frac{\partial L}{\partial Xj} \tag{6}$$

Segundo LeCun (LECUN et al., 2012) esse comportamento das redes utilizando sigmoide logística faz com que a convergência no treinamento seja mais lenta. E o gradiente gerado durante a fase de retropropagação pode ser muito pequeno, pois a derivada das funções sigmoide para

valores de entrada muito alto ou muito baixo é quase 0. Nessa situação, o impacto do gradiente na atualização dos pesos é muito pequeno.

As funções sigmoides quando usadas em redes neurais profundas sofrem de um problema chamado de desaparecimento de gradiente ou gradiente mitigante (GLOROT et al., 2010). O problema de desaparecimento de gradiente é observado quando o treinamento da rede neural não gera resultados melhores (erros menores), devido à saturação dos resultados nas saídas de cada camada da rede neural.

Em 2010, Glorot (GLOROT et al., 2010) demonstrou que seria necessário aguardar muitas épocas nas camadas mais profundas (Figura 5), para que os pesos conseguissem ter uma atualização através da correção por retropropagação, proporcionando uma saída adequada para sua ativação.

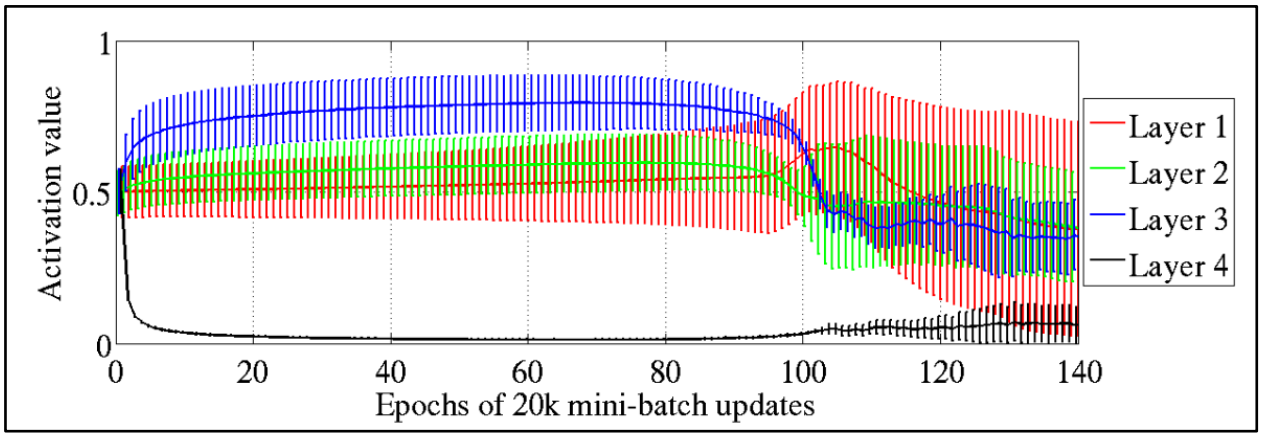


Figura 5: Dificuldade de aprendizado utilizando função de ativação sigmóide (X. Glorot 2010)

É observado na Figura 5, que os resultados da ativação por sigmoide da última camada oculta é saturado para zero quase que imediatamente, por outro lado, as primeiras camadas têm um resultado saturação mais alto. Esse tipo de saturação pode durar por muito tempo (na imagem durou por 100 épocas), ou em outros casos nunca podem mudar. Isto demonstra a dificuldade das

arquiteturas que utilizam a ativação sigmoide em suas camadas mais profundas tem no momento de atualizar os pesos.

2.1.4.2 Função de Ativação ReLU

Atualmente a função de ativação *Rectified Linear Unit* (ReLU) é a mais utilizada em redes neurais. A utilização da ReLU em redes neurais retificadas foi proposta em 2011 por GLOROT et al., no artigo *Deep Sparse Rectifier Neural Networks*. GLOROT aponta que as redes neurais procuram modelar o modelo biológico de forma hierárquica com características semelhante ao córtex visual dos mamíferos.

Estudos mais recentes, (ATTWELL e LAUGHLIN, 2001), sugerem que os neurônios codificam informações de maneira esparsa e distribuída. Estima-se que a porcentagem de neurônios ativos ao mesmo tempo, esteja entre 1 e 4%. As redes neurais profundas ativadas com funções sigmóides não conseguem simular essa propriedade dos neurônios biológicos. Por exemplo, uma rede com ativação sigmóide tem uma representação densa, ou seja, após a inicialização randômica dos pesos quase 100% dos neurônios são ativados e utilizados para processar a saída da rede.

Glorot (GLOROT, 2011) propôs o uso de neurônios retificados (ativados com a função ReLU) em redes neurais artificiais, de forma a modelar de maneira mais fiel à operação do modelo biológico. A ReLU possui a fórmula $\mathbf{f(x) = max\ (0,\ x)}$. De modo que, para valores $x < 0$ os neurônios retificados se tornam inativos (produzem zero como saída) e para $x > 0$ os neurônios retificados operam em um regime linear. Esse comportamento dos neurônios faz com que a rede tenha um grande número de neurônios inativos durante as passagens para frente. Isso permite que a rede obtenha uma representação esparsa, assim como no modelo biológico.

A ReLU em seu domínio positivo ($\mathbf{x} > 0$) apresenta um comportamento linear. Devido a essa linearidade, os gradientes fluem bem no caminho de neurônios ativos, sendo que a derivada para $\mathbf{x} > 0$ é sempre 1. Sendo assim, não há o efeito de desaparecimento de gradiente devido à saturação do neurônio, como acontece em redes ativadas com funções sigmoides.

A Figura 6 apresenta a propagação esparsa de ativações em uma rede com neurônios retificados. Cada entrada da rede é representada por um subconjunto interno de neurônios ativos e o cálculo é linear nesses subconjuntos. Em uma rede com neurônios retificados, após a inicialização uniforme dos pesos, cerca de 50% dos neurônios da rede estão inativos (produzindo zero com saída). Glorot et al., afirmam que além de ser biologicamente plausível a expansividade em redes neurais tem vantagens computacionais.

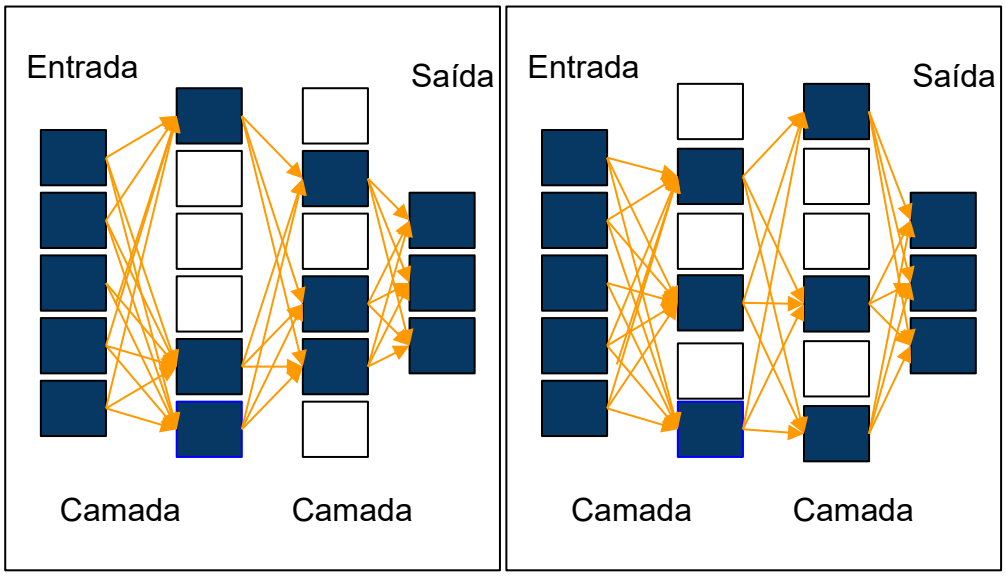


Figura 6: Ilustração da ativação esparsa em camadas ativadas com ReLu. A ativação dos neurônios com ReLu permite que cada entrada tenha seu próprio conjunto de neurônios ativados (GLOROT, 2011)

A ReLU demonstra algumas vantagens:

- **Desembaraçamento de Informações**: A expansividade reduz a complexidade das conexões entre as camadas de neurônios, isolando os neurônios principais de fatores de variação (características) irrelevantes, criando assim caminhos robustos de informação

- **Representações de tamanho variado**: Cada entrada ativa um caminho de informação próprio. O número de neurônios ativos varia para cada input permitindo que a rede controle a dimensionalidade efetiva e precisão necessária para representar a informação de entrada

- **Eficiência Computacional**: A representação esparsa (RE) é mais eficiente que a representação densa (RD). Já que na RE para cada entrada apenas neurônios chave se ativam. Enquanto para RD todos os neurônios são ativados para cada entrada.

2.1.4.3 Função de Ativação ELU

A função de ativação *Exponential Linear Unit* (ELU) foi proposta por Clevert em 2016 (CLEVERT et al., 2016). Esta é uma escolha alternativa em comparação com as funções de ativação convencionais, como a ReLU. Sua formulação matemática é definida como:

$$\begin{cases} x & se \;\; x > 0 \\ \alpha \,.\, (e^x) & se \;\; x \leq 0 \end{cases} \tag{7}$$

Aqui, o parâmetro α é um valor positivo que influencia a inclinação da função para $x \leq 0$. Uma característica diferente da ELU é sua capacidade de suavizar a transição na região negativa, mitigando possíveis problemas associados à ReLU, como o "problema dos neurônios mortos", onde alguns neurônios podem ficar inativos. Este problema surge quando acontece a etapa de correção dos pesos, que precisa propagar o erro para trás, já que o gradiente que passa por estes neurônios é sempre zero, logos os pesos destes neurônios não serão atualizados (CLEVERT et al., 2016).

Uma vantagem adicional da ELU é que ela permite que as ativações assumam valores negativos, o que pode ser benéfico em determinados contextos. A flexibilidade oferecida pela ELU na representação de ativações tanto positivas quanto negativas contribui para a eficácia da função em diversas arquiteturas de redes neurais.

2.1.4.4 Função de Ativação SOFTMAX

A função *Softmax* desempenha um papel crucial em problemas de classificação com múltiplas classes, onde a tarefa é atribuir uma observação a uma entre diversas classes possíveis. Ela transforma um vetor de números reais, frequentemente chamados de logaritmos ou pontuações, em uma distribuição de probabilidade.

A expressão matemática da função *Softmax* para o i-ésimo elemento do vetor de entrada z é dada por:

$$Softmax(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad (8)$$

Aqui, K representa o número total de classes. A função de maximização suave exponencia cada elemento do vetor z_i e normaliza essas exponenciais pela soma de todas as exponenciais, garantindo que a soma das probabilidades resultantes seja igual a um. Essa equalização é crucial para interpretar os valores resultantes como probabilidades. A classe com a maior probabilidade após a aplicação da função de maximização suave é escolhida como a predição final do modelo.

Em resumo, a função de maximização suave é uma ferramenta fundamental para converter pontuações ou *logits* em uma forma que seja adequada para representar distribuições de probabilidade, tornando-a essencial para a fase de saída de modelos de redes neurais em tarefas de classificação de múltiplas classes.

2.1.5 Arquitetura e Aprendizado em Arquiteturas de Rede Neural Multitarefa

Em geral, é comum utilizar modelos únicos de rede neural para tarefas específicas, ajustando seus parâmetros até que seu desempenho não melhore mais. Embora isso permita

alcançar um desempenho aceitável na tarefa em questão, informações que poderiam aprimorar o resultado de outras tarefas correlacionadas são potencialmente ignoradas e perdidas (DESTERRO et al., 2023). A fim de tentar aproveitar essas informações que poderiam se perdidas, Caruana (CARUANA, 1993) propôs Aprendizado Multitarefa (*Multitasking learning* - MTL) em uma arquitetura de rede neural multitarefa. A ideia é que exista uma única rede neural que seja capaz de aprender e executar várias tarefas relacionadas simultaneamente, compartilhando informações e camadas de processamento entre elas. A CARUANA (1993) defende que o compartilhamento de conhecimento entre tarefas em uma rede neural multitarefa pode resultar em uma melhoria substancial no desempenho global do modelo tarefas.

Considerando um cenário de processamento de linguagem natural. Um Modelo de Rede Neural Profunda com MTL pode ser treinado para executar diversas tarefas, como análise de sentimentos, tradução de idiomas e reconhecimento de entidades nomeadas. O compartilhamento das camadas iniciais da rede permite que o modelo aprenda representações de informações genéricas, beneficiando todas as tarefas. Essa abordagem não apenas economiza recursos computacionais, mas também acelera o processo de treinamento e aumenta a eficiência.

2.1.5.1 Motivação para utilização de MTL

No artigo de Sebastian Ruder (RUDER, 2017) fornece algumas razões pelas quais o aprendizado multitarefa é uma abordagem útil em aprendizado de máquina e redes neurais profundas.

1. Compartilhamento de conhecimento: Ao treinar um modelo em várias tarefas relacionadas, a rede neural pode aprender representações compartilhadas que são úteis para todas as tarefas. Isso pode levar a um aprendizado mais eficiente e a uma melhoria no desempenho em todas as tarefas.

2. Regularização: O aprendizado multitarefa pode atuar como uma forma de regularização, tornando o modelo mais robusto e reduzindo o risco de *overfitting*. Ao aprender várias tarefas simultaneamente, o modelo é forçado a aprender

representações mais genéricas, o que pode impedir que ele se ajuste em excesso aos dados (BAXTER, 1997).

3. Dados escassos: Em muitos cenários, os dados de treinamento podem ser escassos para uma tarefa específica. No entanto, ao combinar várias tarefas em um único modelo, os dados de diferentes tarefas podem ser usados para melhorar o aprendizado, mesmo que as amostras de treinamento individuais sejam limitadas.

4. Transferência de conhecimento: O aprendizado multitarefa permite a transferência de conhecimento entre tarefas. Uma tarefa que possui um grande conjunto de dados de treinamento pode ajudar a melhorar o desempenho em tarefas relacionadas com menos dados.

5. Eficiência computacional: Treinar um único modelo para várias tarefas pode ser mais eficiente em termos de recursos computacionais e tempo de treinamento do que treinar modelos separados para cada tarefa. Isso é especialmente benéfico em cenários onde recursos são limitados.

6. Melhorias no desempenho geral: Em muitos casos, o aprendizado com multitarefas pode levar a um melhor desempenho geral, pois as representações compartilhadas podem capturar características relevantes comuns a todas as tarefas.

Entretanto, é fundamental ressaltar que o sucesso do MTL depende da seleção criteriosa das tarefas para compartilhamento e do design atencioso da arquitetura da rede neural. Tarefas muito diferentes podem não se beneficiar do compartilhamento, e a escolha das tarefas adequadas é essencial para atingir os melhores resultados.

Na indústria nuclear, o MTL oferece um potencial notável. A aplicação do MTL na análise de eventos críticos, como a ocorrência de LOCA (*Loss of Coolant Accident*), pode ser de extrema importância. Através do desenvolvimento de uma arquitetura *Multitask deep Neural Network*

(MTDNN) específica para essas tarefas, a análise de LOCA pode se tornar mais precisa e eficaz, contribuindo para a segurança na indústria nuclear.

2.1.5.2 Arquitetura MLT

A arquitetura MTL (*Multi-Task Learning*) é um tipo de arquitetura de rede neural projetada para aprender simultaneamente várias tarefas (ou seja, múltiplas tarefas) em um único modelo. Essa abordagem permite que o modelo compartilhe representações e conhecimento entre as tarefas, o que pode levar a um aprendizado mais eficiente e a um melhor desempenho geral em comparação com treinar modelos separados para cada tarefa (CARUANA, 1993).

A abordagem com MLT envolve a inclusão de camadas compartilhadas e estas camadas capturam representações gerais dos dados de entrada e são usadas por todas as tarefas. As camadas específicas geram representações específicas da tarefa para atender aos requisitos individuais de cada tarefa que é prever a saída da tarefa. Essa arquitetura é ilustrada na Figura 7.

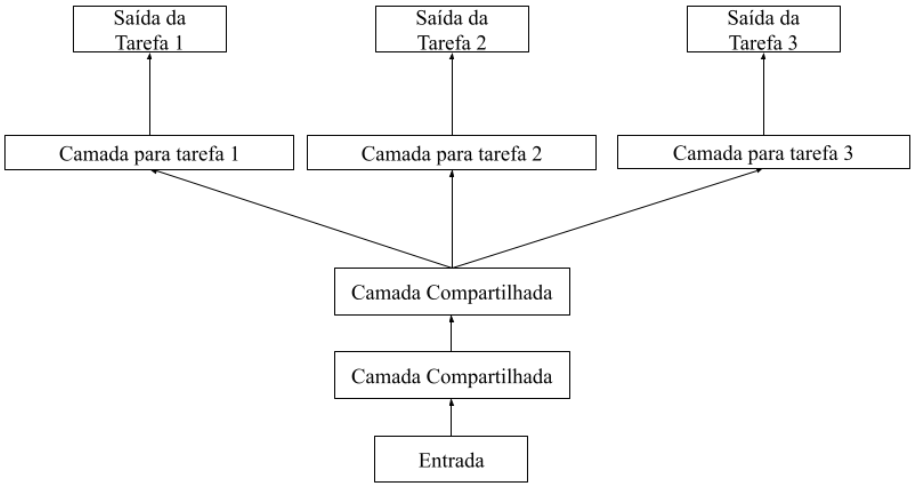


Figura 7: Representação da Arquitetura Multitarefas

Uma função de erro multitarefa é usada para combinar os erros individuais de cada tarefa em uma única função de erro. A função de erro multitarefa é projetada de forma a ponderar a contribuição de cada tarefa de acordo com sua importância relativa, permitindo que o modelo otimize todas as tarefas simultaneamente (RUDER, 2017).

$$Erro\ Multitarefa = \frac{1}{n} \sum_{n=1}^{n} P\ T_n \tag{9}$$

Durante o treinamento, o modelo é atualizado com base na função de erro multitarefa, de modo que os parâmetros da rede sejam ajustados para minimizar essa função. Onde *n* é o número de tarefas, *P* é o peso associado a tarefa e T_n é o erro calculado em cada tarefa ou saída da rede neural. Desta maneira o treinamento em conjunto de todas as tarefas permite que o modelo aprenda representações compartilhadas e específicas para cada tarefa ao mesmo tempo.

2.1.5.3 MultiTarefas

No contexto dinâmico da aprendizagem de máquina, a aplicação do aprendizado de multitarefas emerge como uma escolha natural em situações que demandam previsões para diversas tarefas simultaneamente. Essas circunstâncias são recorrentes em cenários financeiros ou de previsão econômica, onde a predição de muitos indicadores possivelmente correlacionados é essencial. De maneira análoga, na bioinformática, busca-se prever sintomas para várias doenças simultaneamente. Em áreas como a descoberta de medicamentos, onde a previsão de dezenas ou centenas de compostos ativos é necessária, a precisão do MTL demonstra uma melhoria contínua com o aumento do número de tarefas (RAMSUNDAR et al., 2015).

Entretanto, na maioria dos cenários, o foco recai exclusivamente no desempenho de uma única tarefa. Nesta seção, serão exploradas estratégias, demonstradas por Ruder (2017), para identificar tarefas secundárias ou correlacionadas adequadas, que permitem atenuar as qualidades do aprendizado multitarefa.

Tarefas Correlacionadas: A escolha clássica para uma tarefa secundária no MLT é optar por tarefas correlacionadas. Para ilustrar possíveis tarefas correlacionadas, alguns exemplos podem ser mostrados. Caruana (1998) emprega tarefas que predizem diferentes características da estrada como tarefas auxiliares para prever a direção do volante em um veículo autônomo (CARUANA, 1998). Zhang (2014) utiliza a estimativa de pose da cabeça e a inferência de atributos faciais como tarefas auxiliares para a detecção de marcos faciais (ZHANG, 2014); Girshick (2015) prediz conjuntamente a classe e as coordenadas de um objeto em uma imagem (GIRSHICK, 2015); por fim, Arik (2017) realiza a predição conjunta da duração fonética e do perfil de frequência para a síntese de texto em fala (ARIK et al, 2017).

Tarefas Adversárias: Frequentemente, dados rotulados para uma tarefa correlata não estão disponíveis. Contudo, em certas circunstâncias, é possível o acesso a uma tarefa oposta a tarefa principal. Esses dados podem ser aproveitados utilizando uma perda adversária, que não busca minimizar, mas maximizar o erro de treinamento usando uma camada de inversão de gradiente. Essa configuração tem obtido sucesso recentemente em adaptação de domínio (GANIN et al.,

2015). A tarefa adversária, nesse caso, consiste na predição do domínio da entrada; ao reverter o gradiente da tarefa adversária, a perda da tarefa adversária é maximizada, o que é benéfico para a tarefa principal, pois força o modelo a aprender representações que não conseguem distinguir entre domínios (RUDER, 2017).

Dicas: Como mencionado anteriormente, o MLT pode ser utilizado para aprender características que podem não ser fáceis de aprender apenas usando a tarefa original. Uma maneira eficaz de alcançar isso é usar dicas, ou seja, prever as características como uma tarefa auxiliar (RUDER, 2017). Exemplos dessa estratégia no contexto do processamento de linguagem natural incluem, previsão se uma sentença de entrada contém uma palavra de sentimento positivo ou negativo como tarefas auxiliares para a análise de sentimento. Outro exemplo onde prevê se um nome está presente em uma sentença como tarefa auxiliar para a detecção de erros de nome.

Foco de Atenção: Da mesma forma, a tarefa auxiliar pode ser usada para direcionar a atenção para partes da imagem que uma rede normalmente ignoraria (RUDER, 2017). Por exemplo, ao aprender a direção (Caruana, 1998), um modelo de única tarefa pode tipicamente ignorar as marcações de faixa, já que estas constituem apenas uma pequena parte da imagem e nem sempre estão presentes. No entanto, prever as marcações de faixa como tarefa auxiliar força o modelo a aprender a representá-las; esse conhecimento pode então ser usado também para a tarefa principal. Analogamente, para o reconhecimento facial, pode-se aprender a prever a localização de marcos faciais como tarefas auxiliares, uma vez que esses são frequentemente distintivos.

Suavização de Quantificação: Para muitas tarefas, o objetivo de treinamento é quantificado, ou seja, embora uma escala contínua seja mais plausível, os rótulos estão disponíveis como um conjunto discreto. Isso ocorre em muitos cenários que exigem avaliação humana para a coleta de dados, como prever o risco de uma doença (por exemplo, baixo/médio/alto) ou análise de sentimento (positivo/neutro/negativo). O uso de tarefas auxiliares menos quantificadas pode ser útil nesses casos, pois podem ser aprendidas mais facilmente devido à sua objetividade mais suave (RUDER, 2017).

Previsão de Entradas: Em alguns cenários, é impraticável usar algumas características como entradas, pois são inúteis para prever o objetivo desejado. No entanto, elas ainda podem orientar a aprendizagem da tarefa (RUDER, 2017). Nesses casos, as características podem ser usadas como saídas em vez de entradas.

Usando o Futuro para Prever o Presente: Em muitas situações, algumas características só se tornam disponíveis após as previsões serem feitas. Por exemplo, para carros autônomos, medições mais precisas de obstáculos e marcações de faixa podem ser feitas quando o carro está passando por eles (RUDER, 2017). Caruana (1998) também dá o exemplo de previsão de pneumonia, após o qual os resultados de ensaios médicos adicionais estarão disponíveis. Para esses exemplos, os dados adicionais não podem ser usados como características, pois não estarão disponíveis como entrada em tempo de execução. No entanto, eles podem ser usados como uma tarefa auxiliar para transmitir conhecimentos adicionais ao modelo durante o treinamento.

Diversas tarefas podem ser exploradas para potencializar o MLT, mesmo quando o foco é apenas em uma tarefa específica. Contudo, ainda é preciso de um entendimento sólido sobre quais tarefas auxiliares seriam verdadeiramente úteis na prática. A escolha de uma tarefa auxiliar depende, em grande parte, da suposição de que essa tarefa deve guardar alguma relação com a tarefa principal e ser benéfica para prever o resultado da tarefa principal ou ambas.

Caruana (1998) estabelece que duas tarefas são semelhantes se utilizarem as mesmas características para tomar decisões. De forma predominantemente teórica, Baxter (2000) argumenta que tarefas correlacionadas compartilham uma classe comum de hipóteses ótimas, ou seja, apresentam o mesmo viés indutivo. Embora essa perspectiva possibilite a análise de tarefas em que diferentes sensores coletam dados para o mesmo problema de classificação, sua aplicabilidade não se estende a tarefas que não tratam do mesmo problema. Xue et al., (2007) resume que duas tarefas são consideradas semelhantes quando as fronteiras de classificação delas, ou seja, os vetores de parâmetros, são próximas.

Apesar dos avanços teóricos iniciais na compreensão da relação entre tarefas, progressos recentes em direção a esse objetivo são limitados. A similaridade entre tarefas não se resume a uma

dicotomia, mas sim a um espectro (RUDER, 2017). Tarefas mais similares têm potencial para contribuir de maneira mais expressiva para o MLT, enquanto tarefas menos similares oferecem contribuições mais modestas. Permitir que os modelos aprendam quais elementos compartilhar com cada tarefa pode temporariamente contornar a falta de uma teoria consolidada e proporcionar a extração máxima de benefícios, mesmo em relação a tarefas apenas vagamente relacionadas.

2.2 SELEÇÃO DE VARIÁVEIS IMPORTANTES

Em grandes processos industriais, a coleta de dados por meio de sensores e sistemas é uma prática comum. No entanto, a abundância de variáveis pode complicar a tarefa de controle e monitoramento desses processos por parte dos operadores. Com o objetivo de otimizar o controle e o monitoramento desses processos, a área da ciência de dados busca técnicas capazes de selecionar subconjuntos de variáveis relevantes.

A escolha criteriosa das variáveis mais importantes para caracterizar um problema específico pode aprimorar a eficiência dos sistemas de identificação. Além disso, essa abordagem pode reduzir o tempo de processamento, evitando a necessidade de analisar todas as variáveis disponíveis. Nos últimos anos, os pesquisadores têm explorado a técnica conhecida como Floresta Aleatória (FA) na mineração de dados, e essa técnica tem demonstrado alto desempenho em estudos acadêmicos (SANTOS, 2019; PINHEIRO, 2019).

A Floresta Aleatória é um modelo de aprendizado de máquina amplamente utilizado para resolver problemas de classificação e previsão. Embora tenha apresentado resultados sólidos em sua aplicação original, seu uso tem ganhado destaque na área de ciência de dados, especialmente nos últimos anos, devido ao excelente desempenho que demonstra na mineração de dados[1]. Isso ocorre porque a FA é composta por várias Árvores de Decisão (AD), que desempenham um papel

[1] Mineração de dados: *é o processo de explorar grandes quantidades de dados à procura de padrões consistentes, como regras de associação ou sequências temporais, para detectar relacionamentos sistemáticos entre variáveis, detectando assim novos subconjuntos de dados.* DEFINIDO EM: https://pt.wikipedia.org/wiki/Mineração_de_dados

fundamental em sua construção. Por meio das Árvores de Decisão, é possível atribuir notas de importância a cada variável, refletindo o grau de sua relação com o objetivo definido.

2.2.1 Árvore de Decisão

A árvore de decisão (TOM, 1997) é um método que tenta aproximar uma função objetivo de forma discreta. Está função é representada em forma de árvore composta de nós e folhas. Outra forma de entender a representação da árvore de decisão pode ser apresentada como um conjunto de regras.

As árvores de decisão funcionam descendo de um nó raiz até um nó folha onde é encontra-se o objetivo ou a resposta para o conjunto de dados fornecido (Figura 8). Cada nó encontrado na árvore é representado como um atributo do objetivo e cada ramificação descendente deste nó são valores possíveis para esse atributo. O nó raiz realiza a primeira avaliação e com o resultado move-se para baixo, atingindo o próximo ramo da árvore. Esse processo é repetido para o próximo nó da árvore até nó folha. (uma chamada de figura, deve sempre vim no texto antes da figura aparecer.)

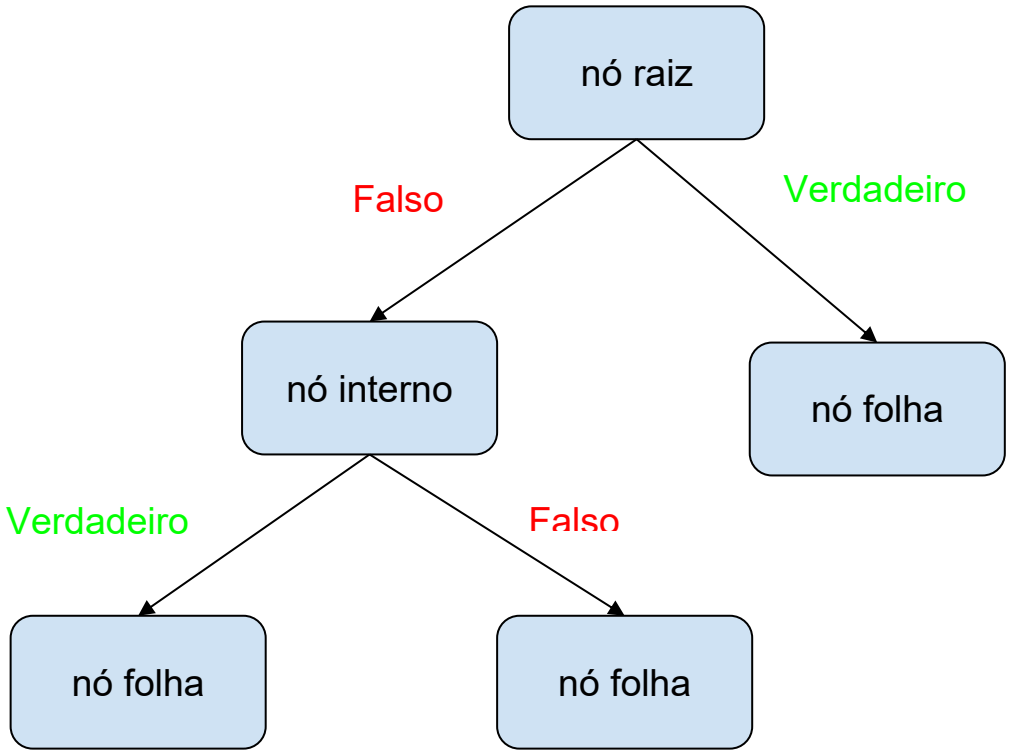


Figura 8: Representação da estrutura básica de uma árvore de decisão

Durante o processo de aprendizado é necessária uma pesquisa dos melhores atributos, onde estes serão alocados de acordo com sua importância de cima para baixo em uma possível árvore de decisão. Um exemplo é o algoritmo ID3 desenvolvido por Quinlan em 1989 (QUINLAN et al., 1989), e depois atualizado para o C4.5 em 1993 (QUINLAN et al., 1993). O algoritmo ID3 tenta construir uma árvore de decisão a partir da raiz, avaliando todos os atributos disponíveis através de um método estatístico. A partir da avaliação dos atributos, o melhor atributo é selecionado e usado para realizar o primeiro teste na raiz da árvore. Um nó descendente é criado para cada valor possível. Os atributos remanescentes serão separados de acordo com cada valor dos novos nós e será realizada uma nova avaliação dos melhores atributos.

2.2.1.1 Mensurando a Importância de Cada Atributo

A avaliação dos atributos é feita sobre uma propriedade estatística chamada de ganho de informação ou em inglês *information gain* (IG) (TOM, 1997). Essa propriedade permite mensurar o quão importante é um atributo associado ao valor do objetivo. A avaliação de um atributo *A* em relação ao atributo alvo *S* em um conjunto de dados aleatório é utilizado a equação 4:

$$IG(S,A) \equiv E(S) - \sum_{v \in Value(A)}^{v} \frac{|S_v|}{|S|} E(S_v) \quad (4)$$

onde Value(A), são todos os valores encontrados no atributo A. S_v é um subconjunto de *S* onde o valor *v* é do atributo A associado com *S*. É possível notar que o primeiro termo da Equação 2 é a entropia do conjunto original *S*, já o segundo termo é a soma das entropias de cada subconjunto S_v, ponderadas pela fração $\frac{|Sv|}{|S|}$que pertencem aos exemplos de S_v.

A entropia é uma medida oriunda da Teoria da Informação, onde esta classifica a pureza de um grupo aleatório de dados. A entropia $E(S)$ codifica o tamanho dos dados em bits, possibilitando identificar os dados pela quantidade de bits.

$$E(S) \equiv \sum_{i \in I} -p_i \ log_2 (p_i) \quad (5)$$

Um exemplo usado na literatura para ilustrar o uso da propriedade IG é tentar predizer se o dia é bom para jogar tênis (TOM, 1997). Em um conjunto que tenha o registro de 14 de dias para treinamento, contém o atributo Vento que pode ter valores Forte e Fraco. Suponha que no conjunto tenha 14 valores, sendo nove positivos e cinco negativos para jogar tênis. Seis valores positivos e dois negativos são associados ao atributo Vento preenchido com valor Fraco, os demais valores

são preenchidos ao atributo Vento com valor Forte. O ganho de informação pode ser calculado com a sequência dos 14 exemplos.

$$Valores(Vento) = Fraco, Forte$$
$$JogarTenis = [9+, 5-]$$
$$JogarTenis_{Fraco} = [6+, 2-]$$
$$JogarTenis_{Forte} = [3+, 3-]$$

$$IG(JogarTenis, Vento) = E(JogarTenis) - \sum_{v \in \{Fraco, Forte\}} \frac{|JogarTenis_v|}{|JogarTenis|} E(JogarTenis_v)$$

$$= E(JogarTenis) - (9/14)\, E(JogarTenis_{Fraco}) - (5/14)\, E(JogarTenis_{Forte})$$
$$= 0{,}940 - (9/14)0{,}811 - (5/14)\, 1$$
$$= 0{,}0615$$

O atributo "vento" teve sua importância calculada e sua medida de importância é 0,048. O algoritmo ID3 faz uso do IG como métrica para medir os atributos associados a cada nó durante o crescimento da árvore.

2.2.1.2 Sobreajuste (*Overfitting*) e Sub-ajuste (*Underfitting*)

O objetivo da aprendizagem de máquina é realizar a classificação ou predição com dados não treinados, o que é chamado de generalização. Quando isso não acontece pode surgir dois problemas o *Overfitting* e *Underfitting*. O *Overfitting* acontece quando os modelos são ajustados de maneira ótima aos dados de treinamento, porém não apresentam um comportamento bom diante de dados não treinados. Ou seja, o modelo memorizou os dados de treinamento, e não aprendeu os relacionamentos dos dados de entrada com os dados de saída. Diz se que quando um modelo apresenta *Overfitting* ele é tendencioso ou inflexível, pois só tem bons resultados para os dados treinados. O *Underfitting* acontece quando o modelo não consegue ter aprendizado bom para os dados de treinamento e dados de testes.

A Árvore de Decisão é propensa de maneira muito forte ao *Overfitting*, devido a grande flexibilidade de crescimento ilimitada, significando que pode continuar crescendo até o último nó de folha para cada objetivo. A limitação do crescimento da Árvore de Decisão permite que o modelo seja mais flexível, entretanto suas respostas não serão com 100% de certeza, podendo surgir o *Underfitting*.

2.2.2 Floresta Aleatória

A Floresta Aleatória (FA) é um conjunto de métodos preditores, no caso Árvores de Decisão. Durante o treinamento da FA cada árvore é inicializada com uma parte menor e aleatória dos atributos disponíveis. Essa construção de um conjunto de árvores tem como objetivo que a FA funcione como um sistema de votação, cabendo a cada árvore dar uma resposta a partir de um subconjunto de dados disponível a ela. Ao final são computadas todas as respostas e a que mais se repetir é selecionada como resposta do modelo. Isso pode ser entendido como um grupo de pessoas que esteja diante de um cenário, entretanto cada pessoa está em uma posição diferente. Estas pessoas podem perceber detalhes diferentes, mediante sua posição desse mesmo cenário e ainda conseguir dar respostas semelhantes.

Em 2001 Breiman (BREIMAN, 2001) propôs avaliar a importância de uma variável para prever Y adicionando a pureza ponderada "p (t) Δi (st, t)" para todos os nós *t* em que a variável é avaliada, com média de todas as árvores na floresta:

$$\mathrm{Imp}(X_j) = \frac{1}{M} \sum_{m=1}^{M} \sum_{t \in \varphi_m} 1(j_t = j) \left[p(t) \Delta i(s_t, t) \right] \tag{9}$$

Onde *M* é o total de Árvores de Decisão criadas no modelo, *t* são todos os nós onde a

variável está sendo avaliada na Árvore de Decisão, p(t) é a proporção de amostras que atingem *t*, j_t indica o identificador da variável usada para dividir o nó *t*.

CAPÍTULO 3. METODOLOGIA DE DESENVOLVIMENTO DOS MODELOS DE RNA

3.1. OBTENÇÃO E PRÉ-PROCESSAMENTO DOS DADOS

Para que um modelo de rede neural possa ser construído há a exigência da disponibilidade de dados, os quais são usados para realização do treinamento e validação do modelo. Entretanto, os dados das plantas nucleares são de difícil acesso ou não existem para determinados cenários de acidente. Devido a isso, se utilizam simuladores para geração de dados. Estes podem ser aplicados no desenvolvimento e melhoria de novas técnicas para sistemas que auxiliam a tomada de decisão.

3.1.1 O Laboratório de Interface Humano-Sistema

Neste trabalho o simulador do Laboratório de Interfaces Homem-Sistema (LABIHS) foi usado para gerar os dados. O simulador está localizado no Instituto de Engenharia Nuclear (IEN), na Ilha do Fundão - RJ onde encontram-se muitas ferramentas para pesquisa e desenvolvimento de novas tecnologias nucleares. Uma delas é o LABIHS, usado pela Divisão de Instrumentação e Confiabilidade Humana (DICH). Ele simula o comportamento de operação de um reator de água pressurizada (PWR) de uma planta nuclear de 930 Megawatts ilustrado na Figura 9. O simulador nuclear emula os circuitos principais da usina nuclear e é operado por painéis digitais de fácil acesso em um conjunto de estações de computador.

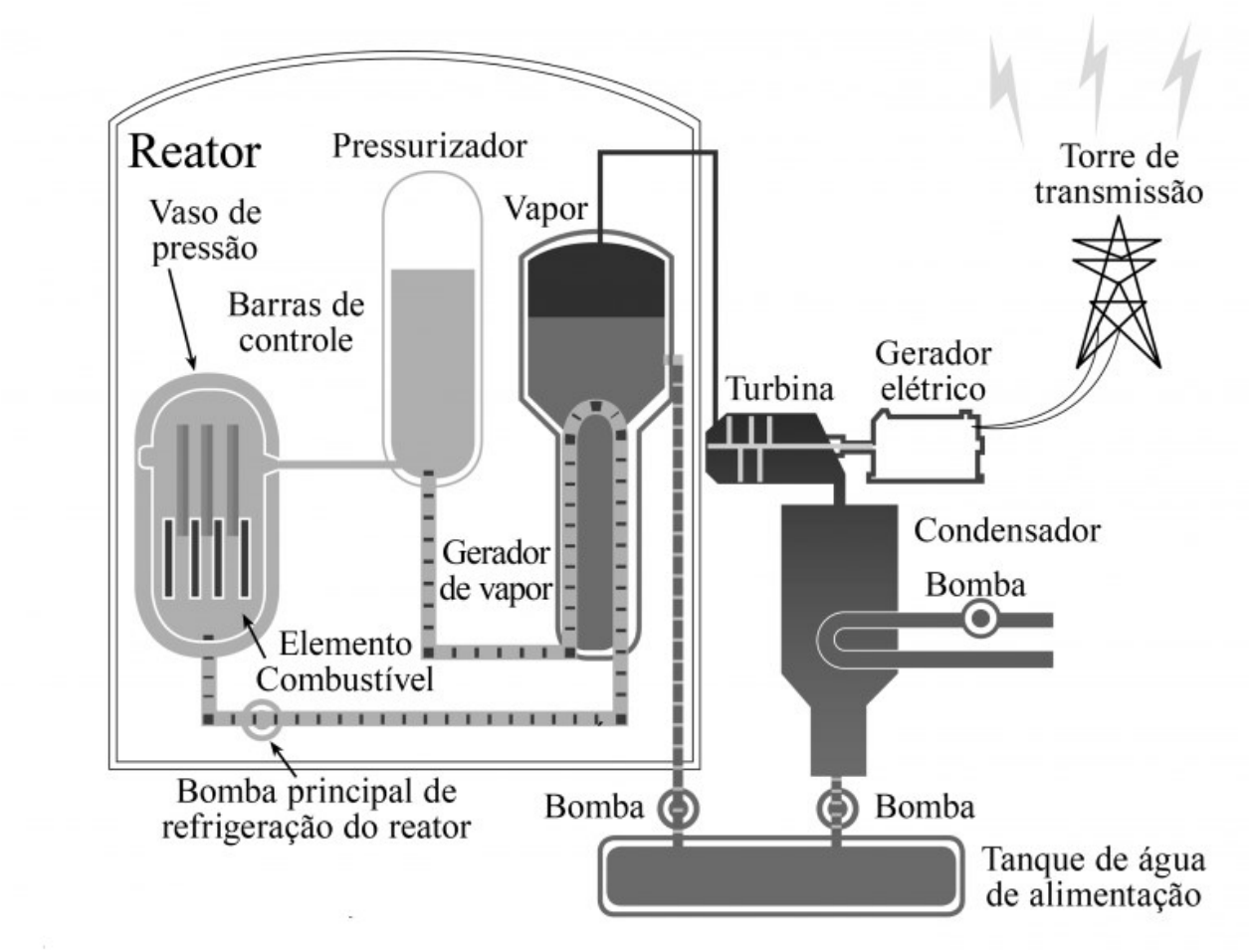


Figura 9: Ilustração do funcionamento da planta nuclear

O simulador usado no LABIHS foi desenvolvido pelo laboratório de pesquisa em fatores humanos do Instituto Kaeri. Este simulador foi baseado em um reator real da Westhinghouse, nomeado de Kori 3&4 e situado na República da Coréia do Sul.

O simulador é formado por cinco partes essenciais, sendo elas:

1. Um modelo matemático com a lógica do funcionamento de uma usina nuclear, programado em Fortran.
2. Memória compartilhada programada em C/C++ para realizar a leitura e escrita das variáveis utilizada no modelo matemático em Fortran.
3. Interface gráfica programa em C/C++ com suporte de bibliotecas do programa ILOG Views Studio.
4. Programa em C/C++ que controla a execução ou interrupção do modelo matemático em Fortran. Com ele também é possível inserir falhas na operação em tempos agendados.
5. Base de dados com informações para inicialização da operação no ambiente simulado.

3.1.2 Simulações Realizadas

As simulações foram iniciadas em um ponto em que a usina se encontra com toda potência de operação ativa (100% de operação). As simulações também tiveram como objetivo simular uma falha na usina após um tempo de 60 segundos do início do simulador. Foram selecionadas 100 variáveis e salvas em arquivo texto em cada simulação. Devido às limitações do sistema o número de variáveis que podem ser armazenadas é de 100 por arquivo de log descritas no Anexo I.

Inicialmente cinco pontos da usina são escolhidos, e rupturas são inseridas durante a operação simulada. Ao total foram realizadas 170 simulações, sendo que nos primeiros 60 segundos a usina se encontravam em estado de normalidade, depois iniciavam-se as rupturas obtendo um tempo de oito a dez minutos de simulação. Em todas as simulações realizadas a planta se encontrava em operação a 100% de potência.

As posições das rupturas na planta nuclear, ilustrado na Figura 10, usadas durante as simulações foram as seguintes:

- Perna Fria do Gerador de Vapor;
- Perna Quente do Gerador de Vapor;
- Pressurizador parte superior;
- Vaso do reator parte inferior;
- Vaso do reator parte superior;

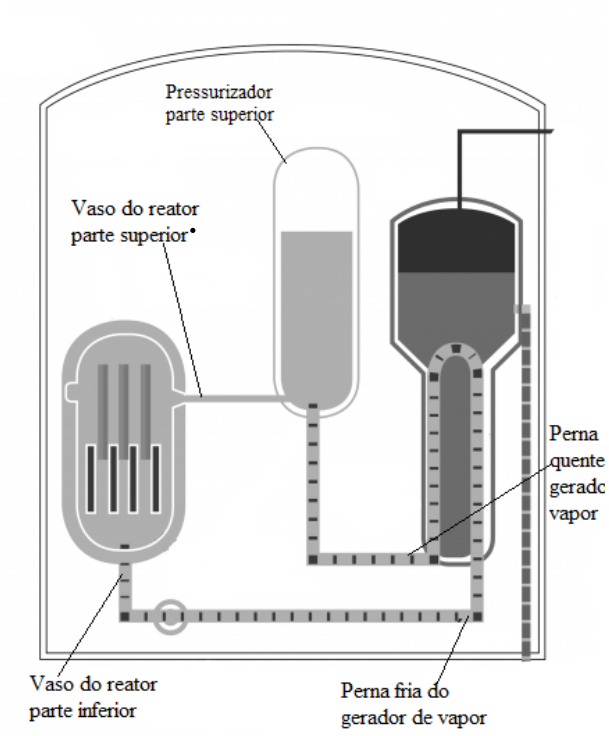


Figura 10: Localização das rupturas causadas durante as simulações

Os intervalos de tamanhos selecionados para causar rupturas simuladas foram escolhidos a partir da definição dos tipos de tamanho de ruptura, sendo estes:

- 0 cm² para operação normal
- de 1 a 10 cm² com incrementos de 1 cm²;
- de 10 a 50 cm² com incrementos de 5 cm²;
- de 50 a 500 cm² com incrementos de 50 cm²;
- de 500 a 1200 cm² com incrementos de 100 cm².

Os locais selecionados para introduzir rupturas no processo de simulação foram escolhidos com base na importância destas no funcionamento seguro e eficiente do reator. Tanto a perna fria (*cold leg*) e a perna quente (*hot leg*) do Gerador de Vapor (GV) são críticas para a transferência de calor do reator para o sistema de produção de energia. Uma falha nestes locais poderia superaquecer o dispositivo e danificar severamente a usina. O pressurizador é crucial para manter a pressão adequada dentro do sistema primário do reator, uma falha nesta localização pode levar a uma perda de controle sobre a pressão, com potencial para causar operação insegura ou acidentes nucleares. Rupturas no vaso do reator, tanto na parte inferior quanto superior, poderiam levar a vazamentos de radiação ao meio ambiente, potencialmente criando um risco para o público e o meio ambiente. A escolha dessas localizações para simulação é essencial para avaliar a capacidade de resposta do sistema de segurança da usina nuclear diante de situações adversas e para garantir a segurança contínua das operações nucleares.

3.1.3 Pré-processamento dos dados

Após as simulações, os dados contidos nos arquivos de *logs* passaram por um processo de tratamento a fim de serem formatados de maneira adequada para uso futuro. Inicialmente, procedeu-se à identificação de linhas com erros na obtenção dos dados. Este problema ocorre quando o simulador, ao lidar com valores muito pequenos, não consegue fornecer uma resposta, resultando em valores nulos. Diante dessa situação, optou-se por remover todas as linhas subsequentes que continham esses valores nulos.

Em seguida, durante o processo de tratamento, realizou-se a tradução dos códigos dos sensores para os nomes que melhor descreviam cada sensor em questão.

Finalmente, efetuou-se a conversão do arquivo de *log* para o formato CSV, com o intuito de facilitar a manipulação e a análise dos dados. Essa medida visa proporcionar uma gestão mais eficiente e acessível das informações obtidas durante as simulações.

3.2. SELEÇÃO DOS CONJUNTOS DE TREINAMENTO, VALIDAÇÃO E TESTE

Nos experimentos iniciais foram usados conjuntos de dados para realizar o treinamento e validação das arquiteturas de redes neurais com aprendizado profundo. Os dados utilizados foram coletados a partir das simulações realizadas no simulador do LABIHS. Cada simulação durou de dois a dez minutos, onde cada segundo é considerado como um exemplo. Determina-se que um "exemplo" é o conjunto de variáveis coletadas em um determinado segundo da simulação.

Três conjuntos de dados foram organizados, o primeiro é o Conjunto de Treinamento cujos dados serão usados para construção dos modelos em forma de treinamento da rede neural. O segundo é o Conjunto de Validação cujos dados serão reservados para avaliar o modelo durante sua construção. Os dados de validação não são usados para treinar o modelo, eles são usados somente para avaliação a cada passo que os pesos do modelo são atualizados. O conjunto de validação é utilizado para verificação da generalização do modelo durante o treinamento, a fim de evitar *overfitting*. O terceiro é o Conjunto de Teste utilizado ao final do treinamento para uma avaliação livre de dados conhecidos do modelo, a fim de avaliar se a rede foi treinada de maneira correta e se é capaz de generalizar sua resposta.

Os dados correspondentes as simulações de tamanhos de 7cm², 40cm², 350cm² e 900cm² de ruptura foram separados na íntegra (100% desses tamanhos simulados) para o conjunto de teste. Os dados restantes (outros tamanhos) foram separados em 80% para treinamento e 20% para validação durante o treinamento. A Tabela 1 apresenta a divisão dos dados tanto em relação a localização, quanto em relação os tamanhos das rupturas, para cada caso: treinamento, teste e validação.

Tabela 1: Simulações realizadas para cada conjunto de dados

Conjunto de dados	Parte da simulação	Perna fria	Perna quente	Pressurizador parte superior	Vaso do Reator parte inferior	Vaso do Reator parte superior
Treinamento	80% das simulações	Tamanhos de rupturas no conjunto de dados de treinamento e validação: 0 cm^2, 1 cm^2, 2cm^2, 3 cm^2, 4 cm^2, 5 cm^2, 6cm^2, 8cm^2, 9cm^2, 10cm^2, 15cm^2, 20cm^2, 25cm^2, 30cm^2, 35cm^2, 45cm^2, 50cm^2, 100cm^2, 150cm^2, 200cm^2, 250cm^2, 300cm^2, 400cm^2, 450cm^2, 500cm^2, 600cm^2, 700cm^2, 800cm^2, 1000cm^2, 1100cm^2, 1200cm^2.				
Validação	20% das simulações					
Teste	100% das simulações	0 cm^2, 7cm^2, 40cm^2, 350cm^2, 900cm^2				

Os dados referentes à descrição da localização onde se encontra a ruptura foram descritos em forma de texto e depois codificados utilizando a técnica de *one-hot encoding*. Está técnica permite que os dados categorizados sejam mais bem representados em colunas. A Tabela 2 demonstra como são os dados de localização e tamanho das rupturas. Na Tabela 3 é demonstrado a descrição do tamanho codificado.

Tabela 2: Dados de localização utilizando *one-hot-encoding*

Pressurizador parte superior	Perna Fria do GV	Perna Quente do GV	Vaso do reator parte inferior	Vaso do reator parte superior	Sem Ruptura	Localização
1	0	0	0	0	0	Ruptura Pressurizador parte superior
0	**1**	0	0	0	0	Ruptura Perna Fria do GV
0	0	**1**	0	0	0	Ruptura Perna Quente do GV
0	0	0	**1**	0	0	Ruptura no vaso do reator parte superior
0	0	0	0	**1**	0	Ruptura no vaso do reator parte inferior
0	0	0	0	0	**1**	Sem Ruptura

Tabela 3: Dados da descrição do tamanho em *one-hot-encoding*

Muito pequeno	Pequeno	Médio	Grande	Sem Ruptura	Descrição
1	0	0	0	0	ruptura muito pequena
0	**1**	0	0	0	ruptura pequena
0	0	**1**	0	0	ruptura média
0	0	0	**1**	0	ruptura grande
0	0	0	0	**1**	Sem ruptura

Os dados de entrada quanto os dados de saída referentes somente ao tamanho da ruptura foram normalizados de acordo com a equação 10. Onde X_N é o valor normalizado, X é o valor original, $\bar{X}$ é a média e S é o desvio padrão. A normalização dos dados é feita para melhorar a eficiência no treinamento da rede neural.

$$X_N = \frac{(X - \bar{X})}{S} \tag{10}$$

Por fim, tem-se na Tabela 4, demonstrando a configuração dos conjuntos de dados de treinamento, validação e teste com todas as variáveis. Em cada célula da tabela é mostrado, entre parênteses um par de números, representando o número de exemplos e depois o número de variáveis de cada exemplo.

Tabela 4: Disposição dos conjuntos de dados

Treinamento	Validação	Teste
(54350, 100)	(13660, 100)	(8197, 100)

3.2. SELEÇÃO DE VARIÁVEIS

Em uma planta de energia nuclear (PNE) existe uma grande gama de variáveis que demonstra o estado de funcionamento da PNE. Estas variáveis são provenientes de sensores e sistemas que realizam cálculos matemáticos a partir dos sensores. A observação destas variáveis

de estado pode auxiliar na identificação prematura de eventos anormais, evitando que estes eventos anormais possam se tornar acidentes.

Entretanto, esta grande quantidade de dados fornecida pela PNE contém muitos recursos redundantes e desnecessários para serem usados na identificação de eventos anormais. Uma grande quantidade de dados também resulta em maior tempo de processamento e baixa taxa de detecção. Diante da grande variedade possível de eventos anormais dentro de uma PNE, que geram grande número de dados que necessitam ser observados para identificação destes eventos, o uso de técnicas para selecionar as variáveis mais importantes tem grande relevância a fim de minimizar o conjunto de dados que precisam ser monitorados (XUAN et al., 2021).

Procurando aprofundar a pesquisa devido a uma grande quantidade de variáveis disponibilizada pelo simulador LABIHS, sendo 100 variáveis ao total, foi realizada a seleção de variáveis mais importantes em uma fase do pré-processamento de dados para modelagem da arquitetura de redes neurais. Diante da grande quantidade de dados gerados pelas simulações, a seleção das variáveis mais importantes se torna interessante e resulta em duas grandes vantagens: Primeiro tem-se um modelo de identificação com menor custo computacional; segundo o modelo pode conseguir interpretar os dados de maneira mais simples tendo uma compreensão melhor dos processos descritos pelos dados e assim realizar seu objetivo com mais precisão (SAEEY et al., 2008).

CAPÍTULO 4. EXPERIMENTOS, RESULTADOS OBTIDOS E ANÁLISE

4.1. MODELOS DE REDE NEURAIS ARTIFICIAIS

Dentro do escopo dos experimentos realizados, a meta central consistiu em identificar um modelo de rede neural otimizado para responder com a melhor previsão possível do problema do LOCA. Para atingir esse objetivo, foram avaliadas diversas arquiteturas de redes neurais profundas com multitarefas (*multi-task deep neural networks* - MTDNN). Uma ampla variedade de hiperparâmetros foram minuciosamente investigados em numerosos experimentos computacionais. Essas investigações abrangeram uma série de variações, tais como:

i) Número de camadas, explorando intervalos entre 2 e 10;

ii) Quantidade de neurônios por camada, variando de 100 a 900;

iii) Utilização de funções de ativação, incluindo *sigmoid*, *softmax*, ReLU e ELU;

iv) Emprego de diversas funções de perda, como *Mean Squared Error* (MSE), *Mean Absolute Error* (MAE) e Entropia Cruzada Categórica;

v) Emprego de algoritmos de otimização, utilizando os métodos *Stochastic Gradient Descent* (SGD) e ADAM;

vi) Variação do número de épocas, com intervalos entre 1000 e 5000;

vii) Modificação do tamanho do lote (*batch*), considerando [32, 64 e 128].

Os intervalos de números de camadas, número de neurônios, e tamanho do lote foram escolhidos de modo empírico. Quanto as funções de ativação, esta foram usadas para avaliar como o aprendizado iria se comporta, visto que em redes com poucas camadas ocultas o uso da função sigmoide poderia trazer melhores resultados, já com uso da função ReLU os melhores resultados estariam em redes com muitas camadas ocultas. Essa extensa exploração permitiu uma análise abrangente do desempenho do MTDNN em uma variedade de cenários, contribuindo para uma compreensão mais profunda da influência de cada parâmetro na eficácia do modelo.

Outro elemento crucial para o aprimoramento do aprendizado da arquitetura MTDNN foi a introdução de uma nova tarefa: a descrição da ruptura para auxiliar como uma dica ao aprendizado do tamanho da ruptura. Essa nova tarefa foi incorporada visando proporcionar à rede neural um entendimento mais abrangente. A lógica subjacente é que, ao aprender essa descrição específica da ruptura, a rede poderá aprimorar significativamente sua capacidade de prever com precisão o tamanho da ruptura. Essa estratégia busca enriquecer o contexto e os elementos considerados pelo modelo, promovendo um refinamento mais eficaz das previsões dos tamanhos, categorias de rupturas apresentadas nas descrições, variam desde operações normais até cenários de ruptura de diferentes magnitudes:

- Normal: Ausência de ruptura (0 cm^2).
- Ruptura Muito Pequena: De 1 a 10 cm^2.
- Ruptura Pequena: De 10 a 50 cm^2.
- Ruptura Média: De 50 a 500 cm^2.
- Ruptura Grande: De 500 a 1200 cm^2.

Essa descrição de magnitude é utilizada exclusivamente para facilitar o aprendizado da rede neural de maneira mais eficaz, não sendo destinada a ser utilizada posteriormente para qualquer análise.

Desta maneira, os experimentos desenvolvidos prosseguiram com uma arquitetura MTDDN básica de uma camada de entrada, camadas ocultas compartilhadas entre as tarefas, camadas ocultas para as tarefas especificas e por fim camadas de saídas para cada uma das três tarefas, demonstrado na Figura 11.

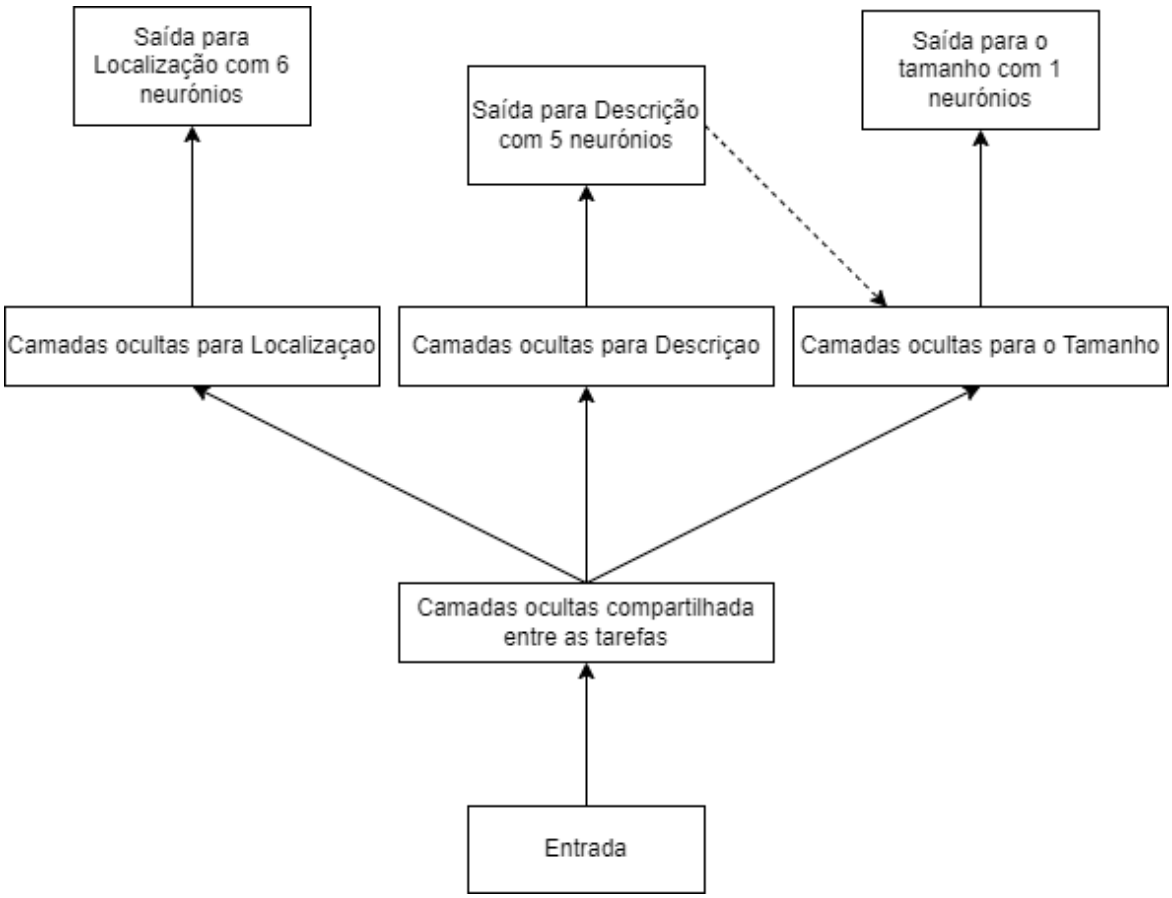


Figura 11: Arquitetura básica da MTDDN. Nesta figura é apresentada a entrada de dados, a camada oculta compartilhada entre todas as tarefas, camadas ocultas para as tarefas específicas, e camadas de saída para cada tarefa.

4.2 RESULTADOS DA SELEÇÃO DE VARIÁVEIS

No estudo conduzido por ALHOWAIDE et al. (2020), foi realizada uma análise comparativa entre vários métodos de seleção de variáveis importantes. Os resultados revelaram que a técnica de Floresta Aleatória (FA) obteve um desempenho melhor em selecionar um conjunto mínimo de variáveis para resolver o problema abordado. Baseado neste trabalho, a Floresta Aleatória foi aplicada aos dados deste estudo com o objetivo de encontrar um conjunto mínimo de variáveis para resolver os problemas ITR e ILF.

O uso da FA envolveu a configuração de parâmetros como a quantidade inicial de árvores aleatórias a serem geradas e a quantidade das variáveis a serem acessadas por essas árvores. No processo, optou-se por criar um total de 50 árvores de decisão, cada uma delas inicializada com 50% das variáveis do conjunto de dados original. Esses parâmetros foram selecionados empiricamente.

Após a aplicação do algoritmo de FA e as pontuações definidas, foi necessário definir qual a pontuação mínima e assim selecionar as variáveis importantes. A média das pontuações e usada como pontuação mínima, demonstrado na Figura 12.

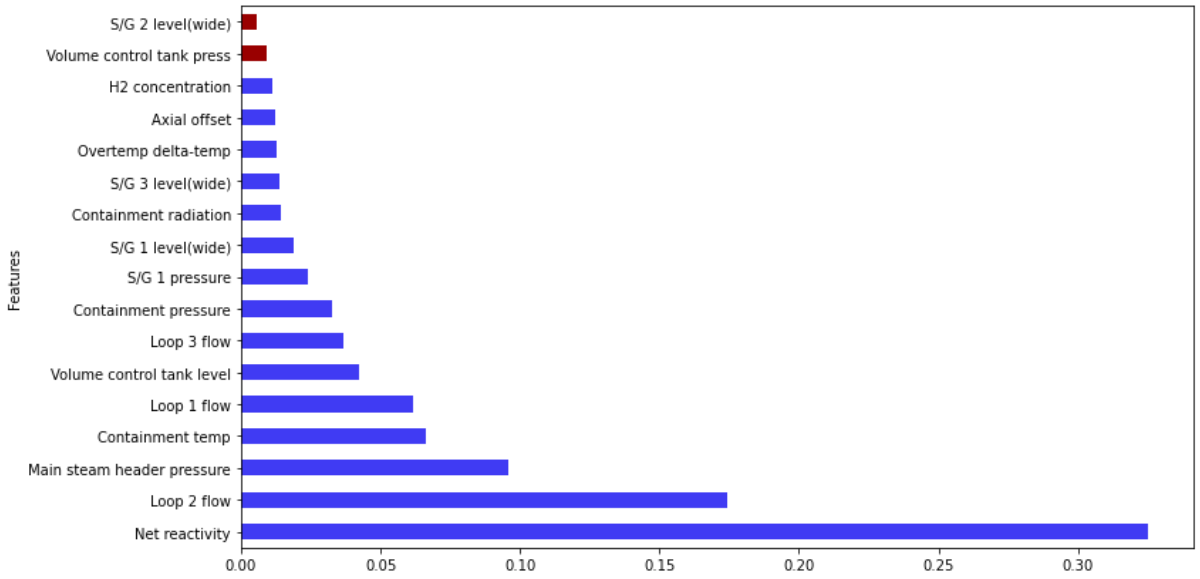


Figura 12: Imagem demonstrando as variáveis que obtiveram maior pontuação. Com a barra em azul estão a variáveis com a pontuação acima da média, e em vermelho as com pontuação abaixo da média (média das pontuações 0,01).

A partir do ranqueamento feito pela pontuação gerada através da FA foi formado um conjunto de 15 variáveis que será usado para formar um exemplo, demonstrado na Tabela 5. O ponto de corte é a média da pontuação dada pela FA. Esse conjunto de variáveis colhido de segundo a segunda durante a simulação formara o que é definido de exemplo.

Tabela 5: Conjunto de variáveis selecionadas com Random Forest

ID	Variable	Unit
1	Reatividade	dk/k(%)
2	Fluxo do setor 2	(%)
3	Pressão principal do coletor de vapor	(kg/cm2)
4	Temperatura na contenção	(°C)
5	Fluxo do setor 1	(%)
6	Nível de volume do tanque	(%)
7	Fluxo do setor 3	(%)
8	Pressão na contenção	(kg/cm2)
9	Pressão no gerador de vapor 1	(kg/cm2)
10	Nível no gerador de vapor 1 (*wide*)	(%)
11	Radiação na contenção	(mrem/hr)
12	Nível no gerador de vapor 3 (*wide*)	(%)
13	Delta da Temperatura	(°C)
14	Diferença de Fluxo Axial	(%)
15	Concentração H_2	(%)

Por fim, tem-se na Tabela 6, demonstrando a configuração dos conjuntos de dados de treinamento, teste e produção utilizando apenas as variáveis importantes pré-selecionas pela FA. Em cada célula da tabela é mostrado, entre parênteses um par de números, representando o número de exemplos e depois o número de variáveis de entrada de cada exemplo.

Tabela 6: Disposição dos Dados de Entrada da RNA

	Treinamento	**Validação**	**Teste**
Entrada	(54350, 15)	(13660, 15)	(8197, 15)

4.3 TREINAMENTO DE ARQUITETURAS DE REDES NEURAIS QUE UTILIZAM TODAS AS VARIÁVEIS

Na primeira fase, foi adotado uma abordagem abrangente, empregando todas as cem variáveis (V) disponíveis durante o processo de treinamento.

Descrição da entrada da arquitetura:

$\{ V_1, V_2, V_3, V_4, V_5 \ \ldots \ V_{96}, V_{97}, V_{98}, V_{100} \}$

Descrição da saída (S) da arquitetura:

$\{ S_1, S_2, S_3, \ldots S_9, S_{10}, S_{12} \}$

Após a execução de mil épocas de treinamento em diversas arquiteturas, foi selecionado e destacado os resultados mais promissores, os quais foram posteriormente organizados na **Tabela *7***.

Tabela 7: Os 10 melhores resultados que os modelos treinados apresentaram no conjunto de teste

Número de camadas internas	Número de neurônio por camada	Função de ativação	Acertos na localização (%)	Acertos na descrição da ruptura (%)	De 0% a 10% de erro na área da ruptura
6	500	relu	83,66%	98,85%	88,68%
3	400	relu	81,13%	98,98%	86,45%
3	300	relu	83,43%	99,86%	84,93%
3	500	relu	81,39%	98,14%	80,88%
6	300	relu	83,02%	95,47%	79,52%
6	400	relu	81,26%	99,51%	78,63%
5	300	relu	82,05%	98,86%	76,97%
4	300	relu	80,81%	99,77%	74,76%
5	400	relu	81,67%	99,38%	73,63%
4	400	relu	81,54%	98,12%	72,02%

A **Tabela** *7* foi compilada com base no desempenho na predição da ruptura no conjunto de teste, e os resultados foram ordenados destacando o modelo líder. Nesta avaliação todos os modelos que obtiveram os melhores resultados usaram a função de ativação ReLU.

Com resultado de 88,68% dos dados com menos de 10% de erro na predição do tamanho de ruptura e 86,66% de acerto na identificação da localização o primeiro modelo na tabela exibiu a mais alta capacidade de aprendizado ao utilizar todas as variáveis disponíveis. Sua arquitetura é composta por uma camada de entrada, seguida por uma camada compartilhada com as camadas particulares aos objetivos, depois segue as camadas ocultas dedicadas a cada objetivo. As camadas pertinentes a predição da localização é ativada pela função de ativação *relu*, as camadas pertinentes pela predição do tamanho da ruptura são ativadas com *elu*, conforme ilustrado de maneira gráfica na Figura 13 e descrito na Tabela 8.

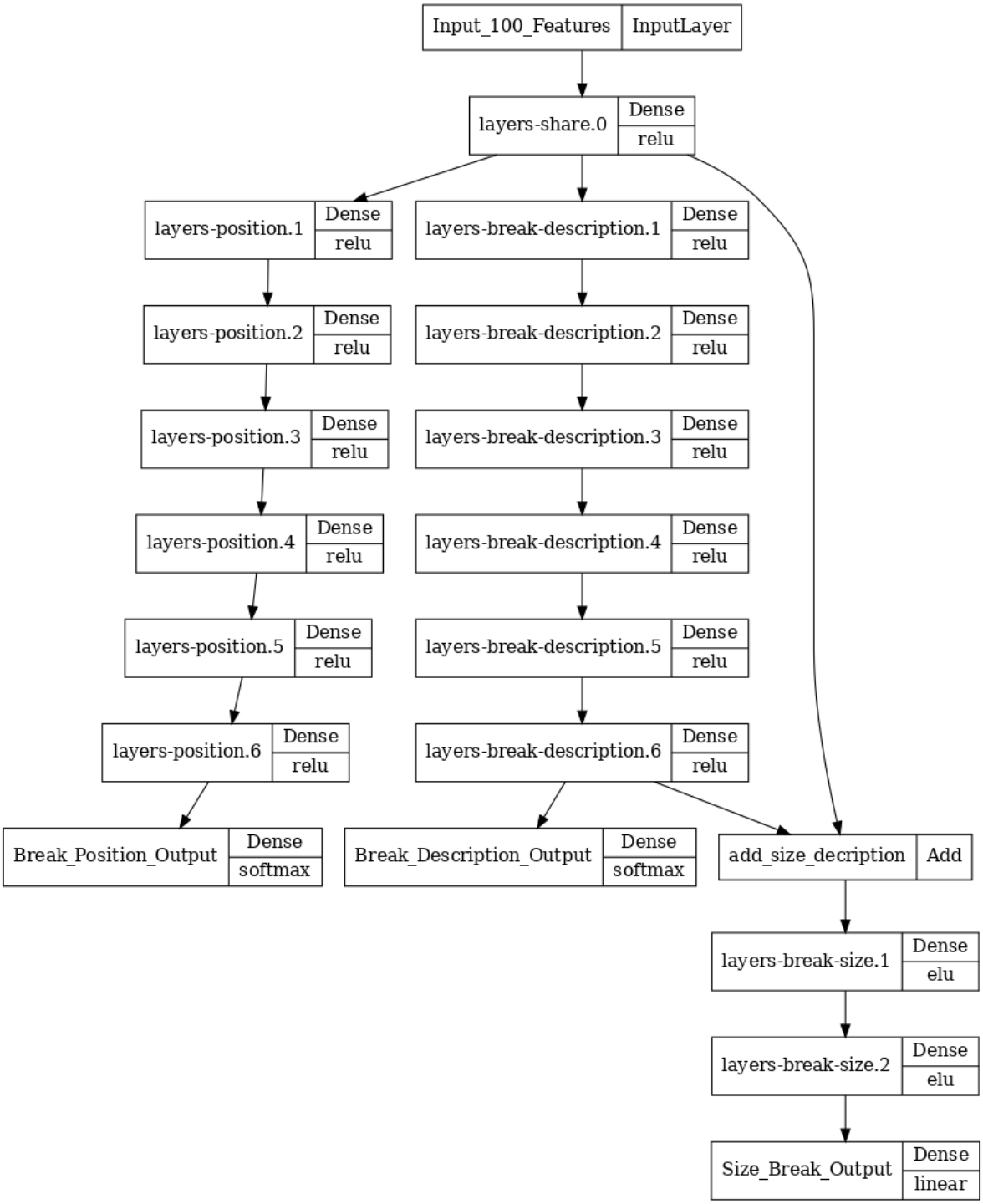


Figura 13: Arquitetura que demostrou melhor aprendizado durante os experimentos, utilizando todos os dados disponíveis.

Tabela 8: Descrição da arquitetura do melhor modelo treinado na fase 1

Camadas compartilhadas	1
Camadas da Localização	6
Camadas da Descrição	6
Camadas para o tamanho	2
Número de camadas ocultas	15
Função de Ativação	RELU/ELU
Algoritmo de correção dos pesos	ADAM
Números de épocas	Duas mil

Após identificar a arquitetura que apresentou o melhor desempenho, prosseguiu-se a um novo treinamento com um número expandido de épocas, buscando avaliar mais a fundo a capacidade de aprendizado dessa arquitetura. Entretanto, os testes realizados nessa fase revelaram um desempenho insatisfatório após atingir a marca de duas mil épocas de treinamento. Observou-se que, após esse ponto, o modelo desenvolvido enfrentava dificuldades em generalizar padrões para os dados de teste. Essa limitação evidenciou a necessidade de uma reavaliação e refinamento da estratégia adotada.

Diante deste cenário, foi estabelecido um critério rigoroso para monitorar o progresso da rede neural durante o treinamento. Em casos nos quais a rede não apresentava um aprimoramento contínuo nos dados de validação, interrompia-se o treinamento após um período de 300 épocas consecutivas sem observar melhorias significativas, com um limite de duas mil épocas, como mostra a Figura *14* e os resultados na Tabela 9.

Essa intervenção foi crucial para mitigar os riscos de *overfitting* e garantir que a rede neural não apenas aprendesse com os dados de treinamento, mas também pudesse generalizar eficazmente para novos conjuntos de dados, como os provenientes do conjunto de teste. A abordagem de interromper o treinamento após um período sem melhorias tangíveis destaca o comprometimento com a qualidade da generalização do modelo, um aspecto fundamental em aplicações práticas.

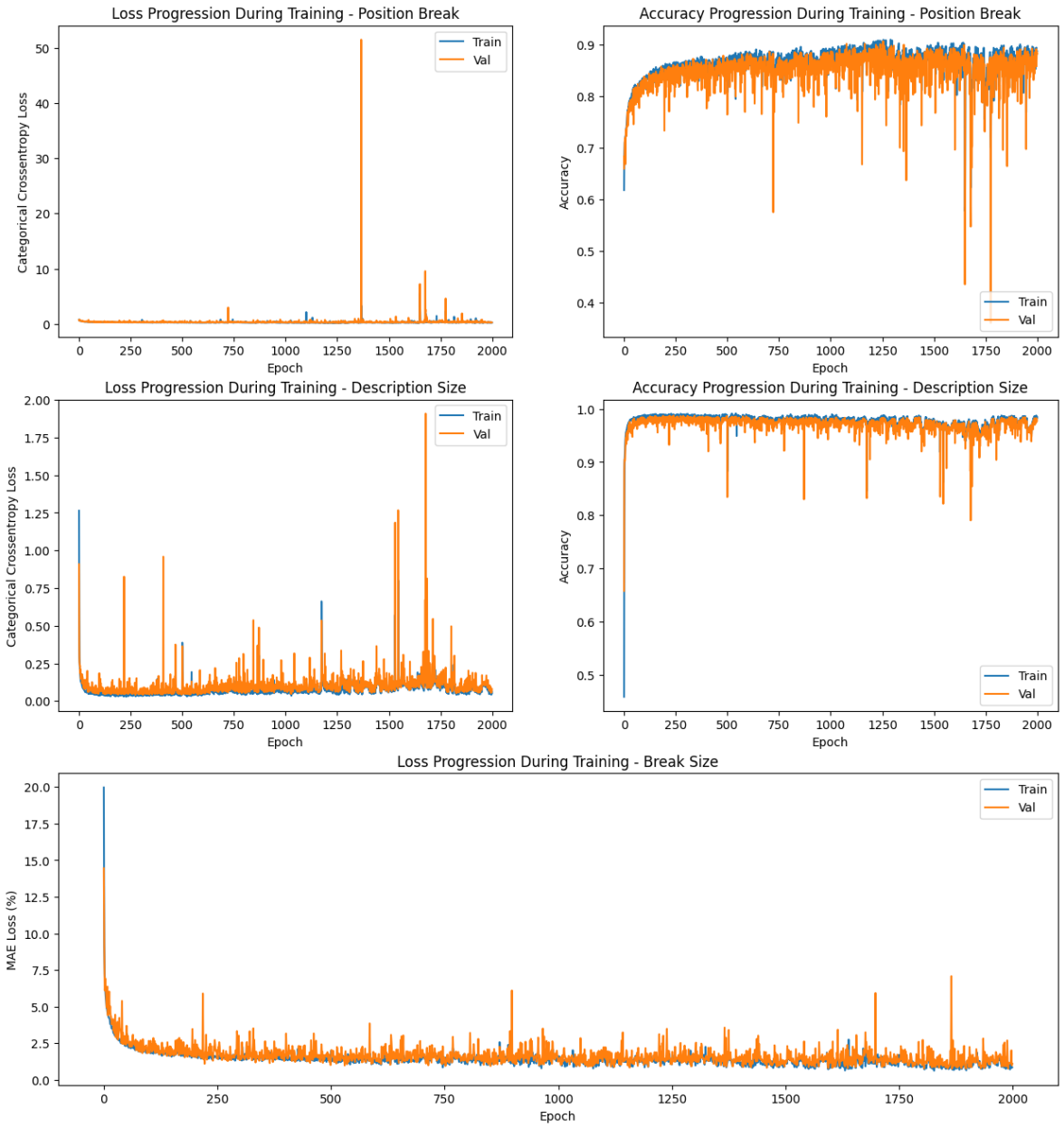


Figura 14: Progresso do erro durante o treinamento para os conjuntos de treinamento e validação

Tabela 9: Resultados do modelo usando todos os dados

Métricas	Treinamento	Validação	Teste
Número de exemplos	54155	13614	8438
Acertos na localização (%)	92,59%	90,13%	82,80%
Acertos na descrição da ruptura (%)	98,46%	98,02%	99,22%
De 0% a 10% de erro na área da ruptura	92,52%	90,38%	88,41%
De 10% a 20% de erro na área da ruptura	4,61%	5,77%	6,59%
De 20% a 30% de erro na área da ruptura	1,87%	2,10%	2,50%
Mais que 30% de erro na área da ruptura	0,99%	1,75%	2,50%
Erro Relativo	3,06	3,76	15,79

O treinamento adicional não evidenciou melhorias significativas, como destacado na Tabela 9. A precisão da localização melhorou para 92.59%, mas não superou os resultados da literatura (TING-HAN LIN et al., 2021). Os resultados quanto a predição do tamanho da ruptura teve o erro acentuado, diminuindo o número de exemplos com erro menor que 10%, demonstrado que para este tipo de arquitetura o treinamento com muitas épocas resulta em *overfiting*.

Para uma análise mais aprofundada dos erros até agora apresentados, será exibida a matriz de confusão. Essa ferramenta permite a identificação das áreas em que o modelo enfrenta maior dificuldade ao tentar determinar a localização da ruptura. Uma matriz específica foi gerada para cada conjunto de dados.

Nas Figuras Figura 15, Figura 16 e Figura 17, que exibem as matrizes de confusão dos conjuntos de dados de treinamento, validação e teste. Na matriz de confusão é demonstrado que quanto mais perto de 1 melhor é a identificação da localização pela MTDDN, deste modo é notável

uma confusão significativa na região do vaso do reator. Essa dificuldade pode ser atribuída à proximidade física da ruptura.

Outros pontos de confusão também foram identificados, evidenciando os desafios enfrentados pela arquitetura durante o processo de aprendizado.

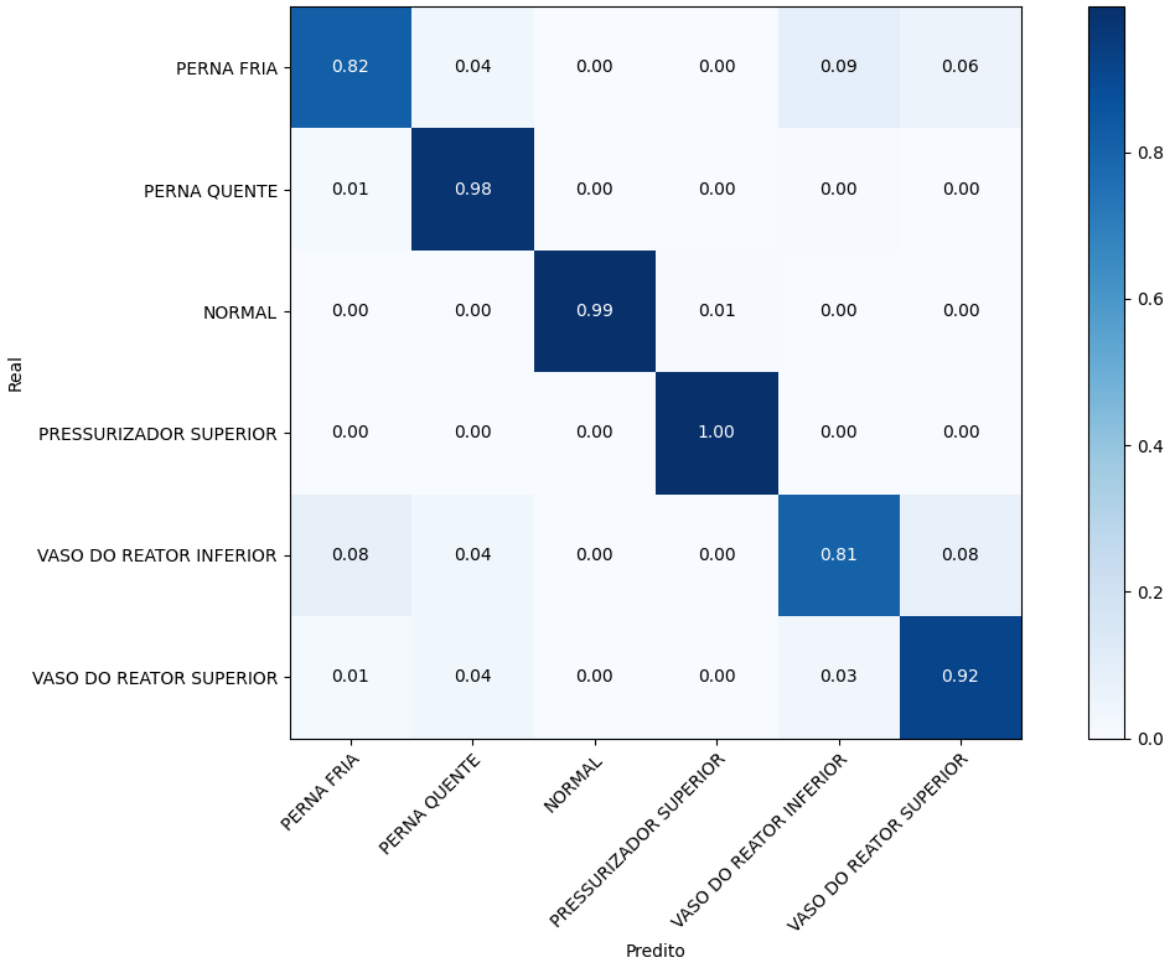


Figura 15: Matriz de confusão para predição da localização no conjunto de treinamento

Na Figura 15, é possível observar que, mesmo no conjunto de treinamento, onde o desempenho é otimizado, ainda ocorrem confusões em localizações bastante distintas, como na perna fria e na parte superior do reator.

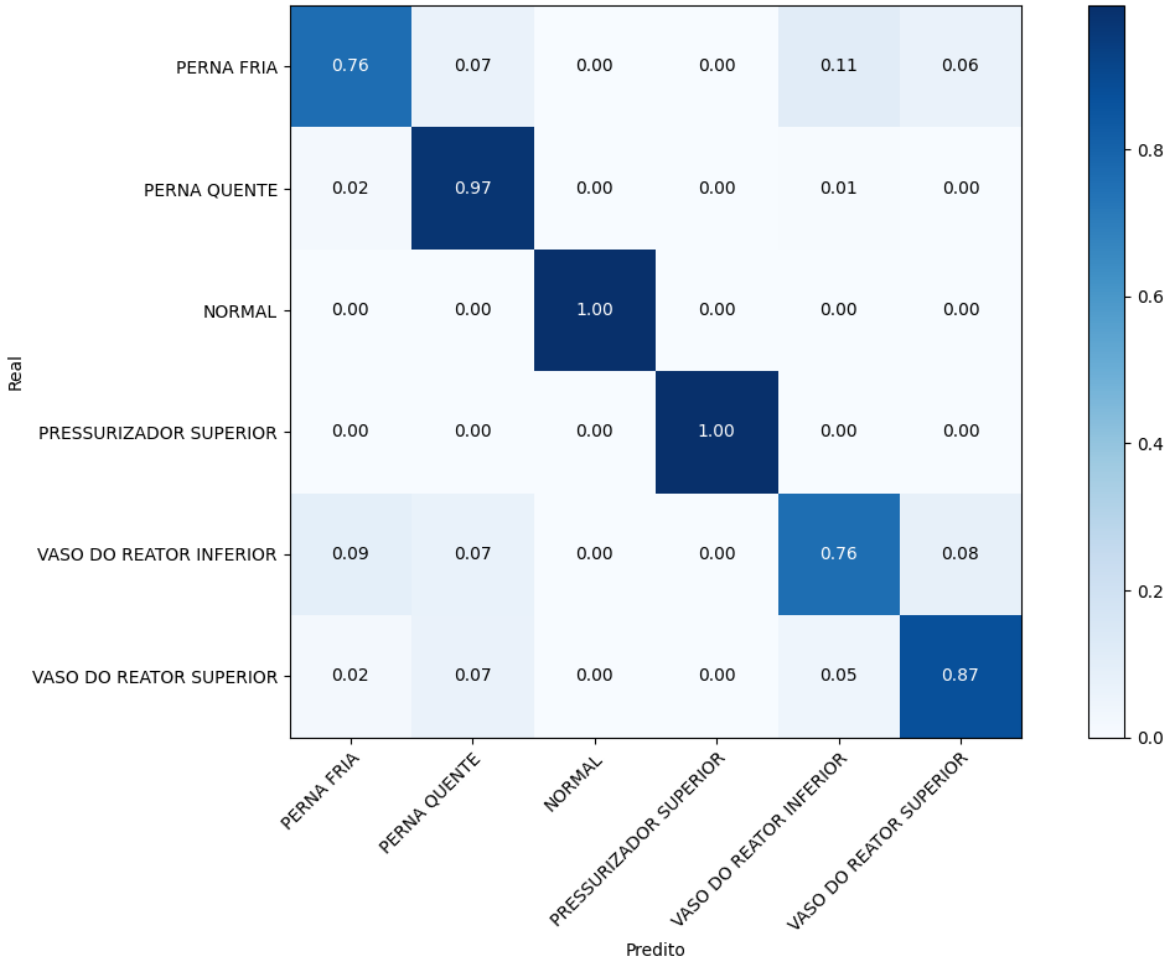


Figura 16: Matriz de confusão para predição da localização no conjunto de validação

Na Figura 16, o conjunto de validação, empregado durante o treinamento para avaliar o próprio processo de aprendizado, igualmente apresenta resultados não satisfatórios devido a confusão entre localizações destintas, como por exemplo a perna fria e a parte superior do reator.

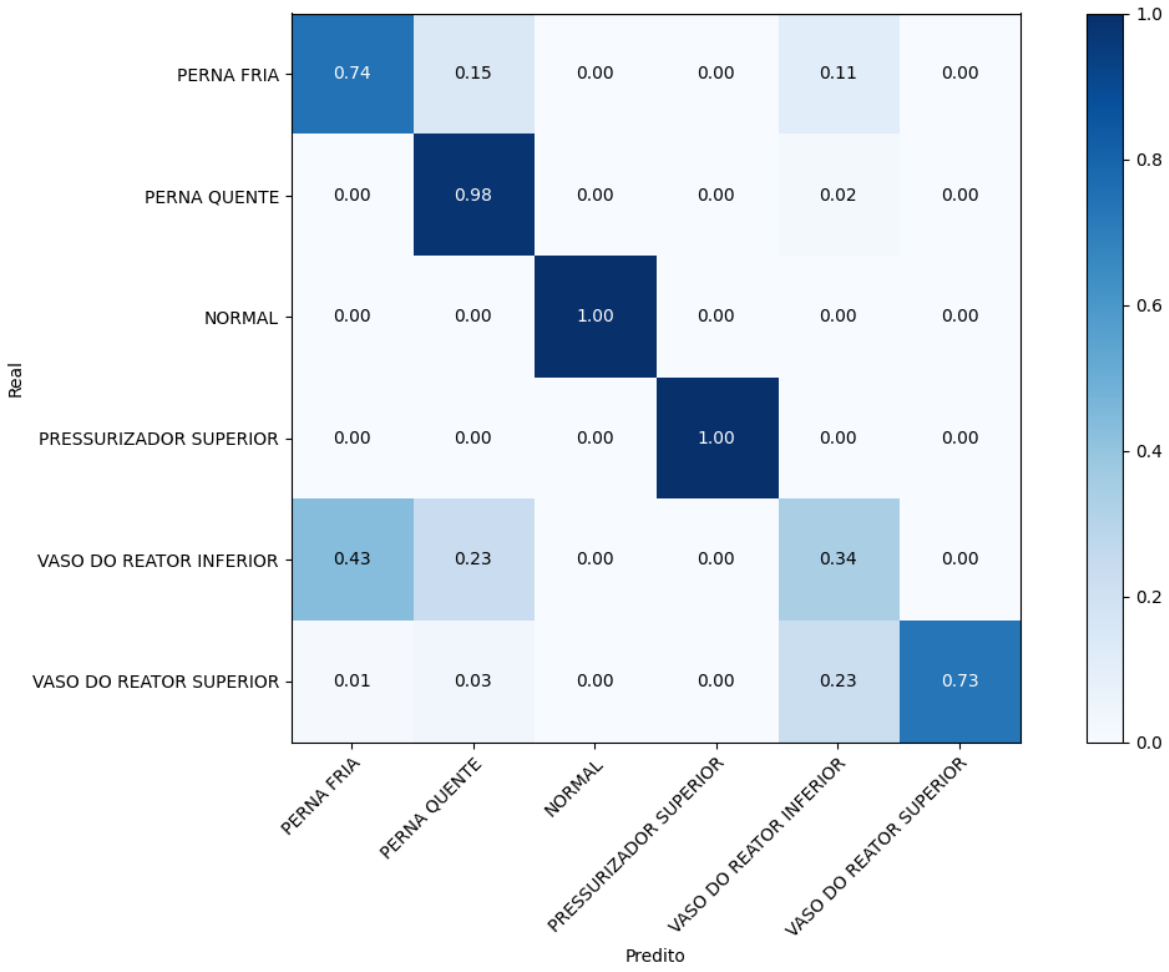


Figura 17: Matriz de confusão para predição da localização no conjunto de teste

Na Figura 17, o conjunto de teste, utilizado para avaliação pós treinamento, também revela resultados não satisfatórios, com predições confusas em locais distintos como perna fria e parte superior do reator. Este padrão de confusão persistente em diferentes áreas indica que o modelo pode enfrentar dificuldades ao generalizar ou reproduzir com precisão padrões específicos presentes nos dados de teste. Essa análise aponta para a necessidade de estratégias adicionais para aprimorar a generalização do modelo, aumentando sua capacidade de lidar eficazmente com variações e complexidades nos dados não vistos durante o treinamento.

4.4 TREINAMENTO DE ARQUITETURAS DE REDES NEURAIS QUE UTILIZAM AS VARIÁVEIS IMPORTANTES PRÉ-SELECIONADAS PELA FA

Dada a constatação de que os modelos de redes neurais, treinados com a totalidade das cem variáveis, não apresentaram resultados substanciais, uma nova série de experimentos foi realizada, focando na utilização exclusiva das variáveis mais importantes. Essas 15 variáveis foram criteriosamente selecionadas por meio da técnica de Floresta Aleatória, apresentado no capítulo 3.2. Os experimentos subsequentes envolveram arquiteturas de redes neurais de diversos tamanhos, mantendo a consistência da pesquisa com a abordagem da primeira fase. As configurações das arquiteturas avaliadas foram ajustadas apenas em relação à quantidade de variáveis de entrada.

Descrição da entrada da arquitetura:

$\{ V_1, V_2, V_3, V_4, V_5 \quad \ldots \quad V_{12}, V_{13}, V_{14}, V_{15} \}$

Descrição da saída (S) da arquitetura:

$\{ S_1, S_2, S_3, \ldots S_9, S_{10}, S_{12} \}$

A Tabela 10 destaca as arquiteturas mais eficazes avaliadas no conjunto de teste.

Tabela 10: Os 10 melhores resultados que os modelos treinados apresentaram no conjunto de teste

Número de camadas internas	Número de neurónio por camada	Função de ativação	Acertos na localização (%)	Acertos na descrição da ruptura (%)	De 0% a 10% de erro na área da ruptura
3	300	elu	97,94%	99,61%	93,62%
3	500	relu	96,69%	99,79%	93,44%
3	300	relu	94,07%	99,72%	92,87%
3	300	elu	98,73%	99,91%	92,18%
3	300	elu	98,28%	99,86%	91,67%
3	400	elu	98,06%	97,38%	90,94%
4	500	relu	93,61%	99,94%	89,18%
3	400	relu	94,66%	99,80%	86,47%
3	500	elu	96,88%	97,64%	85,71%
3	400	relu	96,09%	98%	84,19%

Na Tabela 10, compilada com base na previsão do tamanho da ruptura, destaca-se o primeiro modelo como líder de desempenho. Este modelo evidenciou melhorias significativas em relação à fase anterior, demonstrado uma precisão de 97,94% na identificação da ruptura e 93,62% dos dados têm erro menor que 10% para predição do tamanho da ruptura, enfatizando assim a relevância do uso de variáveis pré-selecionadas.

A arquitetura que teve melhor desempenho, demonstrada na Tabela 10, é caracterizada por uma camada de entrada, uma camada compartilhada para ambos os objetivos, três camadas ocultas dedicadas à predição da localização e mais três camadas ocultas voltadas para a predição do tamanho da ruptura, esse modelo se destaca como líder. É crucial ressaltar que todas essas camadas foram ativadas pela função *elu*, conforme representado de maneira gráfica na Figura 18 e descrita na

Tabela *11*. Esses resultados apontam para a eficácia da abordagem de seleção de variáveis na otimização do desempenho do modelo.

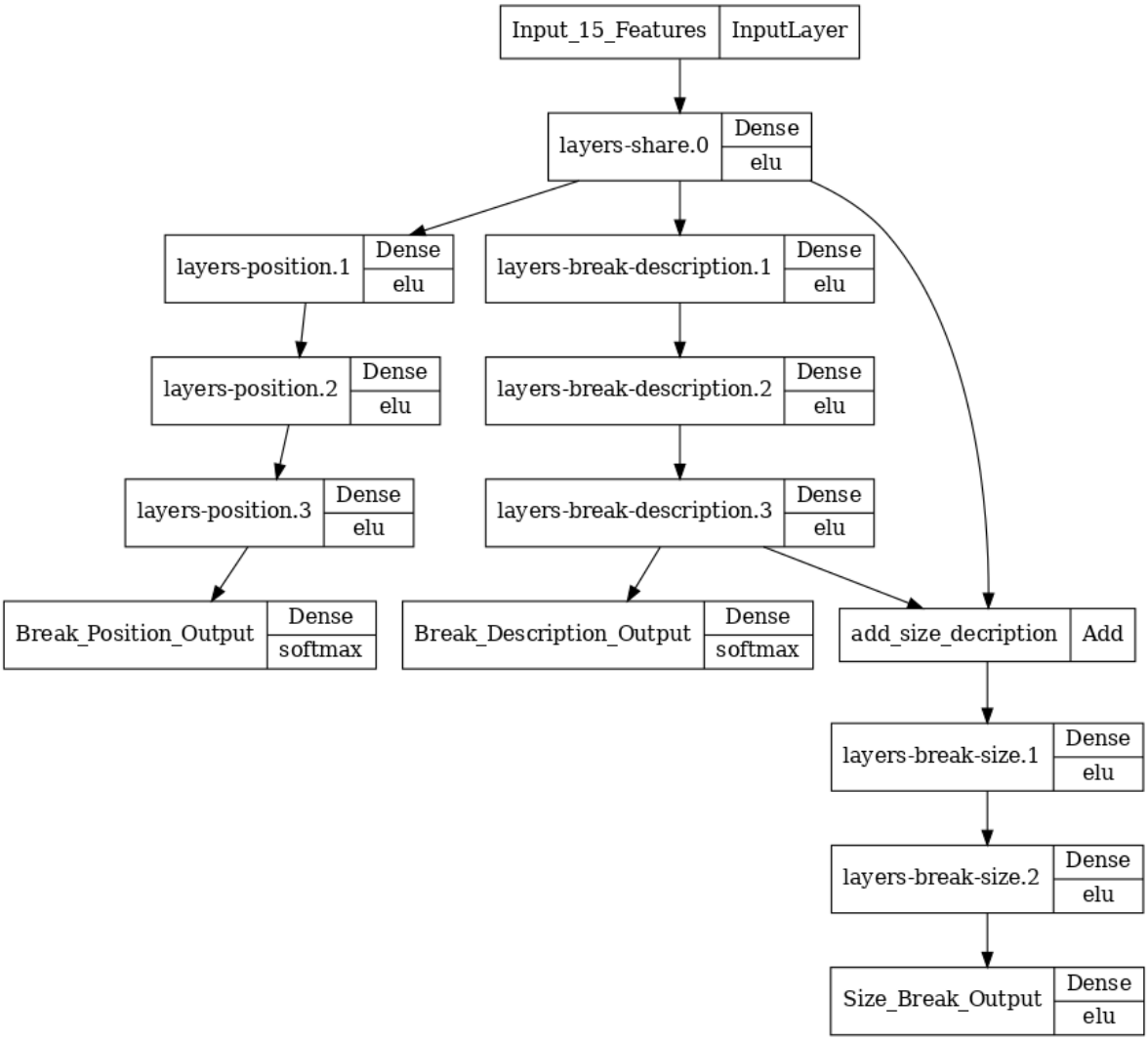


Figura 18: Arquitetura que demostrou melhor aprendizado durante os experimentos, utilizando 15 variáveis mais importantes

Tabela 11: Descrição da arquitetura do melhor modelo treinado na fase 2

Camadas compartilhadas	1
Camadas da Localização	3
Camadas da Descrição	3
Camadas para o tamanho	2
Número de camadas ocultas	9
Função de Ativação	RELU/ELU
Algoritmo de correção dos pesos	ADAM
Números de épocas	Duas mil

Seguindo a metodologia delineada na fase 1, realizou-se um novo ciclo de treinamento empregando a melhor arquitetura, e em seguida, ampliando significativamente o número de épocas. No entanto, constatou-se uma persistência do mesmo problema identificado anteriormente ao superar as duas mil épocas de treinamento. Apesar de melhorias aparentes no conjunto de treinamento, verificou-se, em avaliações subsequentes no conjunto de teste, uma deterioração nos resultados. Esse cenário evidencia a dificuldade dos modelos em generalizar para esse conjunto específico de dados após um treinamento mais extenso.

Dessa forma, optou-se novamente por limitar o número de épocas a duas mil, interrompendo o treinamento caso não fosse observada uma melhoria significativa no desempenho do conjunto de validação, como mostra a Figura 19.

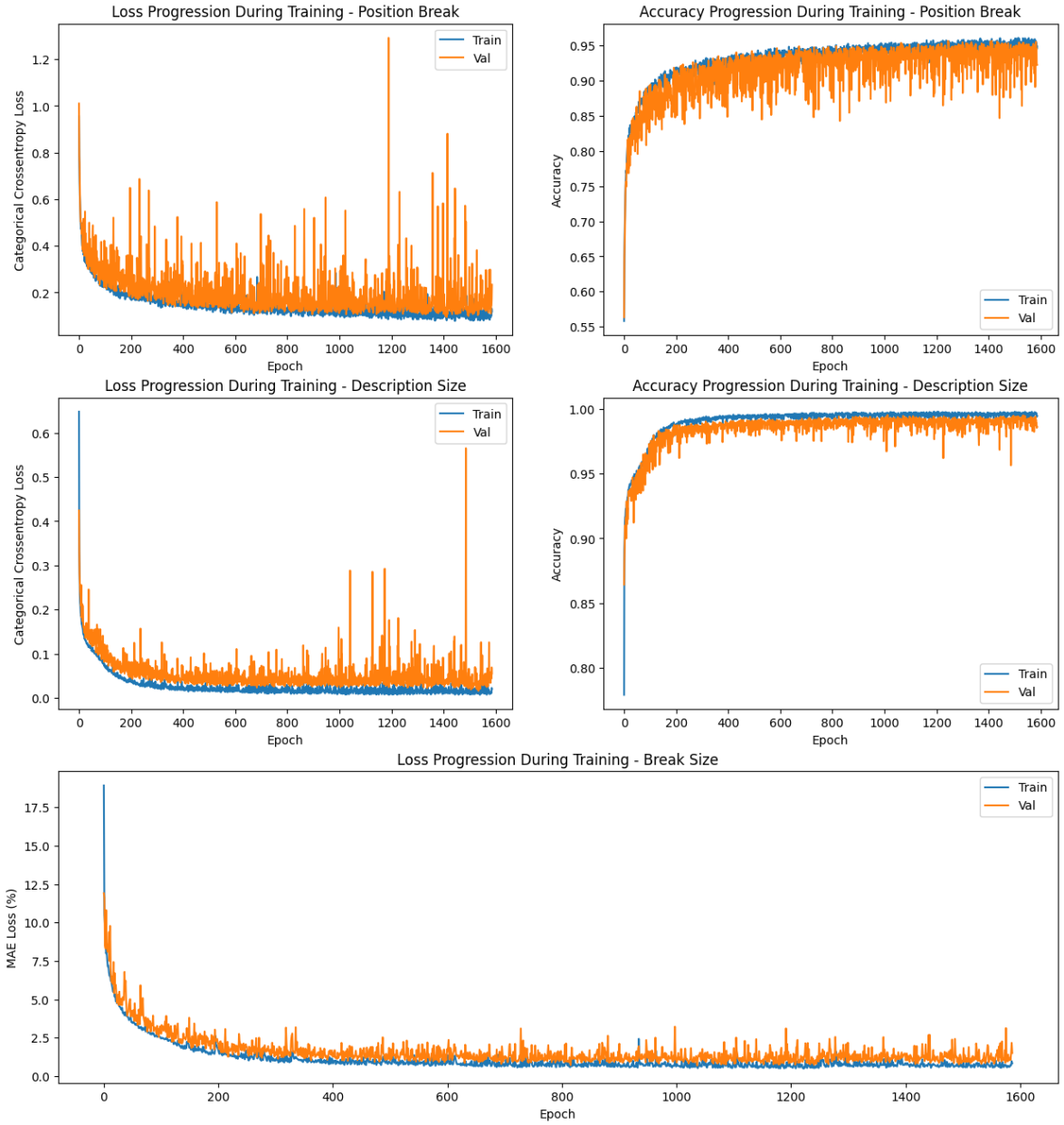


Figura 19: Progresso do erro durante o treinamento para os conjuntos de treinamento e validação

Após a conclusão do treinamento, observa-se uma melhoria significativa nos resultados do conjunto de teste, indicando uma diferença notável em relação à fase anterior. Além disso, o controle do número de épocas teve o efeito desejado, resultando na obtenção dos melhores pesos para o modelo e, consequentemente, alcançando excelentes resultados, conforme evidenciado na Tabela 9. Essa abordagem estratégica não apenas contribui para a eficiência do treinamento, mas

também destaca a capacidade do modelo de generalizar de maneira mais robusta para dados não vistos, resultando em desempenho aprimorado no conjunto de teste.

Tabela 12: Resultados do modelo usando apenas as variáveis mais importantes

Métricas	Treinamento	Validação	Teste
Número de exemplos	54155	13614	8438
Acertos na localização (%)	96,38%	95,31%	**99,12%**
Acertos na descrição da ruptura (%)	99,81%	99,29%	99,02%
De 0% a 10% de erro na área da ruptura	83,55%	82,75%	**94,36%**
De 10% a 20% de erro na área da ruptura	5,55%	5,96%	2,03%
De 20% a 30% de erro na área da ruptura	4,7%	4,75%	2,4%
Mais que 30% de erro na área da ruptura	6.2%	6.54%	1,2%
Erro Relativo	6,87%	7,17%	4,26%

Os resultados obtidos ao término do segundo treinamento constituem um ponto significativo, destacando melhorias substanciais na precisão das previsões em relação à localização, notou-se um notável avanço, para uma taxa de 99.12%. Além disso, ao analisar padrões nos quais os erros na previsão do tamanho da ruptura eram inferiores a 10%, observou-se uma parcela de 94.36% dos exemplos. Este aprimoramento destaca a robustez do modelo, especialmente na categorização de padrões com discrepâncias mínimas entre a previsão e o valor real do tamanho da ruptura.

As análises representadas pelas matrizes de confusão nas FigurasFigura 20,Figura 21, e Figura *22* revelam um desempenho notavelmente assertivo na predição da localização alcançando valores bem próximos de 1, significando uma baixa taxa de confusão em grande parte das áreas.

No entanto, mesmo com a precisão elevada de maneira geral, observa-se uma persistente dificuldade em localizar a ruptura, especialmente na região próxima ao vaso do reator.

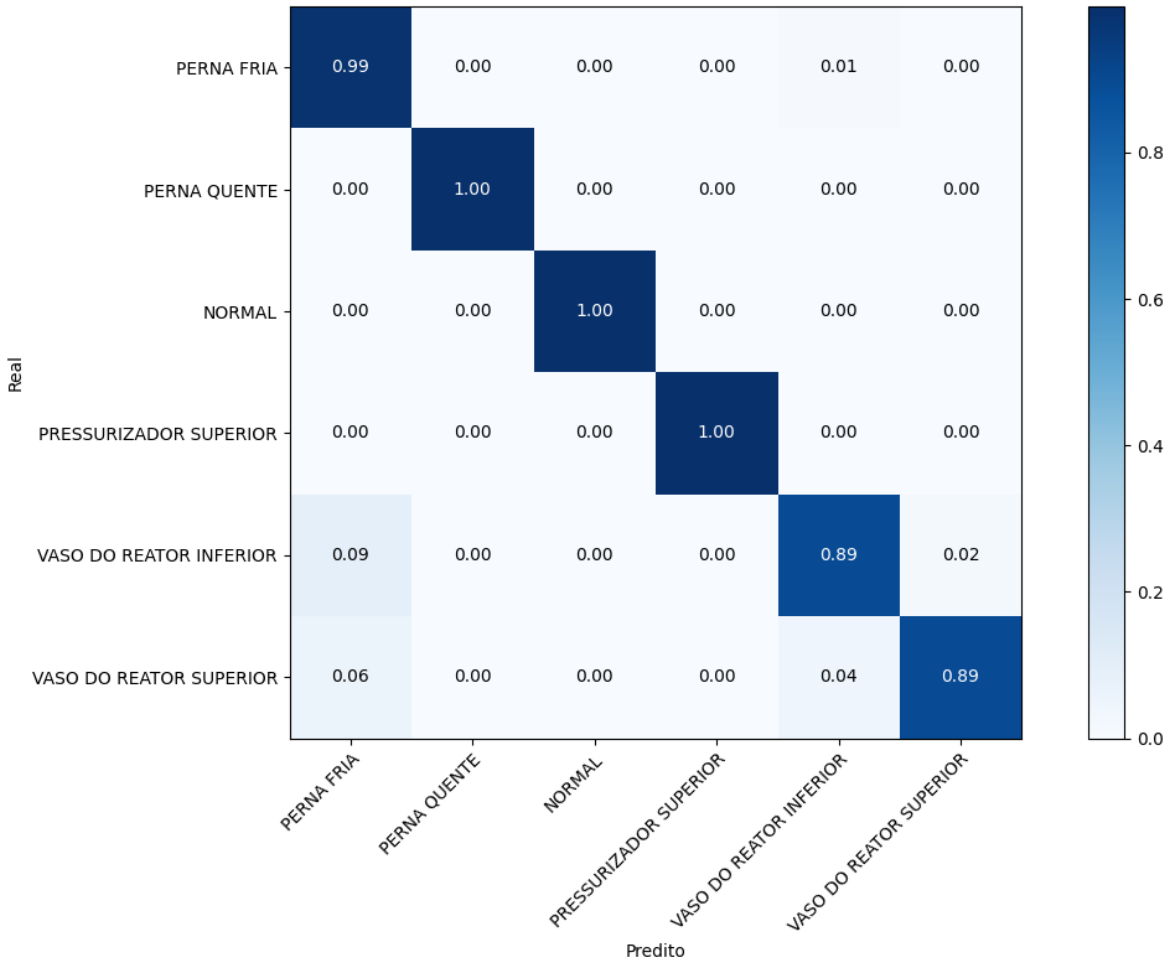


Figura 20: Matriz de confusão para predição da localização no conjunto de treinamento

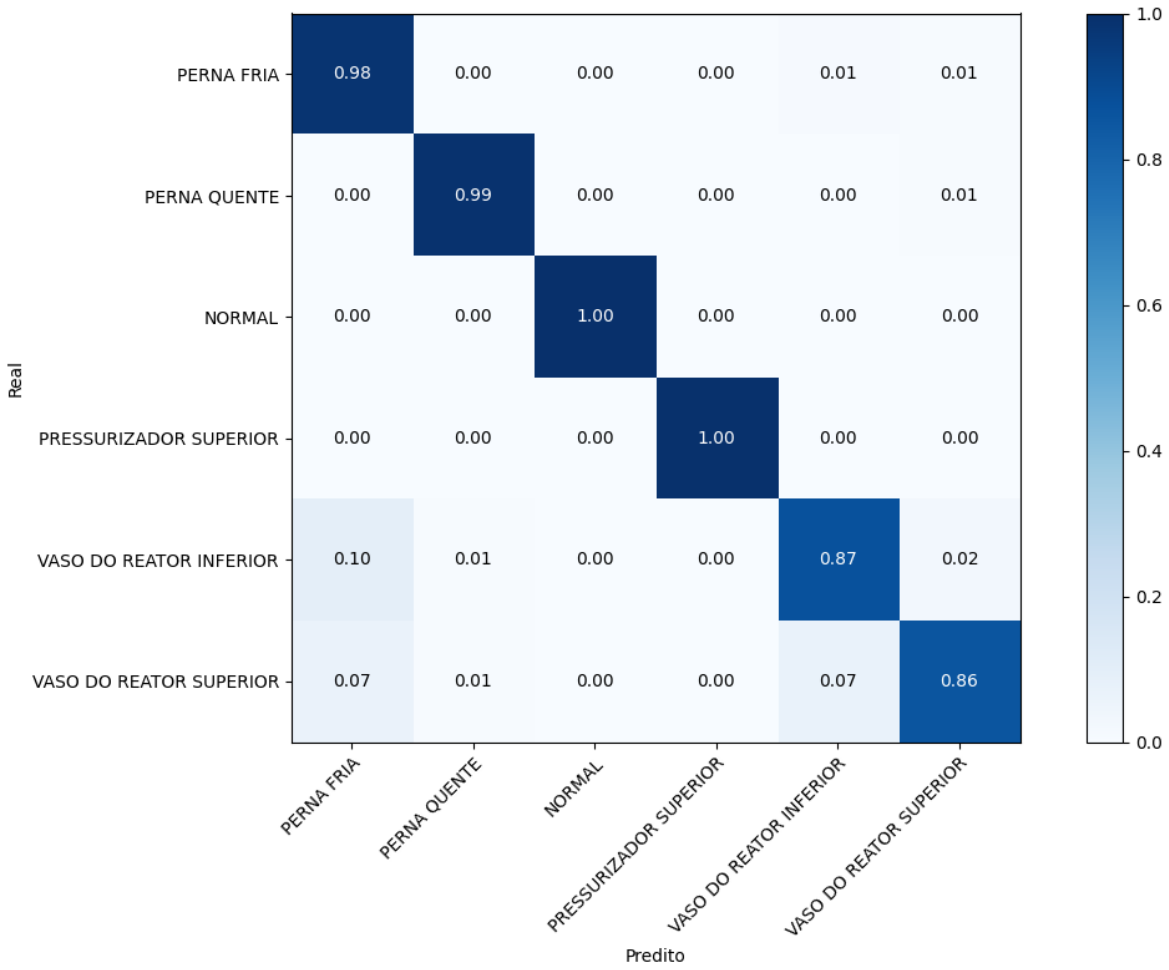


Figura 21: Matriz de confusão para predição da localização no conjunto de validação

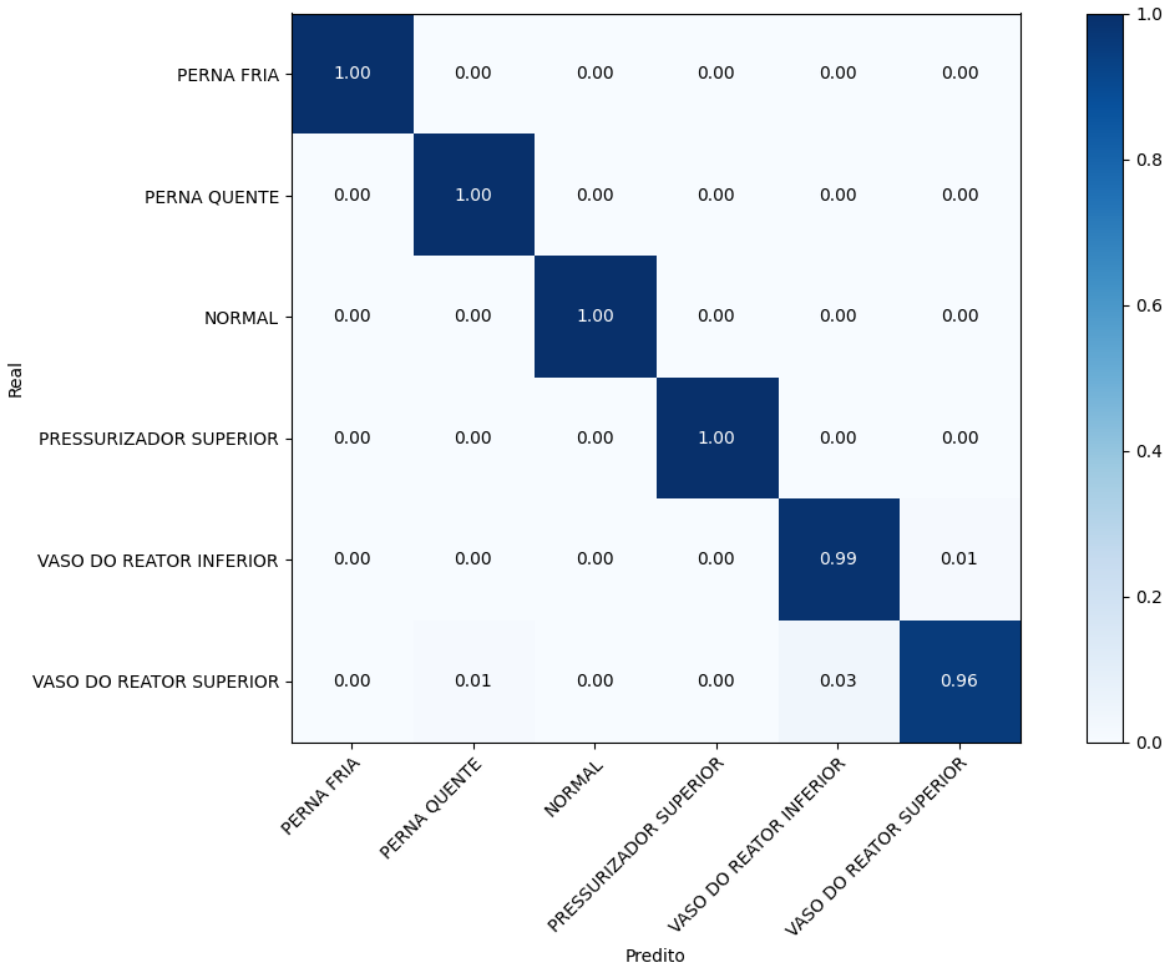


Figura 22: Matriz de confusão para predição da localização no conjunto de teste

Estes resultados confirmam a eficácia da estratégia adotada, conferindo ao modelo uma boa assertividade na identificação da localização da ruptura.

A seleção das quinze variáveis importantes revelou-se crucial para aprimorar a eficiência da MTDNN, confirmando que uma abordagem estratégica na escolha desses elementos resulta em dados mais relevantes. Essa abordagem oferece insights importantes para pesquisadores que buscam melhorar resultados utilizando arquiteturas de redes neurais multitarefas, sublinhando a importância de considerar não apenas a quantidade, mas também a relevância das variáveis durante o treinamento do modelo.

4.5 COMPARAÇÃO E ANÁLISE DE RESULTADOS

O presente estudo teve como parte do objetivo o desenvolvimento de um modelo de rede neural para predição da localização e do tamanho da ruptura de acidentes do tipo LOCA. O desenvolvimento deu-se em duas fases. A metodologia empregada na Fase 1 envolveu o treinamento da rede utilizando todas as cem variáveis disponíveis, enquanto a Fase 2 explorou um refinamento no conjunto de dados, selecionado e utilizando as quinze melhores variáveis identificadas por meio de uma Floresta Aleatória, descrito no capítulo 3.2.

Na Fase 1, o melhor modelo treinado obteve a taxa de acertos na localização de 82,80%, indicando uma possível lacuna na capacidade de modelo de estimar com precisão a posição exata da ruptura. A análise do outro objetivo da arquitetura revelou que apenas 88,41% dos exemplos têm erros entre 0 e 10% para a predição da área da ruptura, sugerindo uma necessidade de aprimoramento na quantificação precisa da extensão da ruptura. O erro relativo de 15,79% para a predição da área da ruptura também reforça a necessidade de ajustes na arquitetura para melhorar a precisão global do modelo.

Na Fase 2, a arquitetura foi otimizada com a seleção das quinze melhores variáveis obtidas por meio de uma Floresta Aleatória. Os resultados desta fase foram superiores, evidenciando uma melhoria substancial em todas as métricas avaliadas.

O modelo na fase 2 conseguiu elevar consideravelmente a precisão na localização da ruptura, atingindo uma taxa de acerto de 99,12%. Outra notável redução na distribuição de erros na área da ruptura, nas faixas de 10-20%, 20-30% e acima de 30%. Vale destacar que o número de casos com erros abaixo de 10% aumentou para 94,36%, indicando a capacidade do modelo em avaliar com precisão a extensão da ruptura. O erro relativo na área da ruptura foi reduzido para apenas 4,26%, o que indica uma melhoria substancial na precisão geral do nosso modelo.

A tabela 13 demonstra como o uso de variáveis importantes tornou os resultados mais precisos na fase 2, principalmente na predição do tamanho de rupturas acima de 500 cm². Outro

benefício que o uso de variáveis importantes trouxe neste estudo é o custo computacional menor para processar os dados mais específicos. Isso fez com que uma arquitetura de rede neural com menos camadas ocultas alcançasse um resultado superior à fase 1 (

Tabela *11*).

Tabela 13: Comparação de Resultados por faixas de tamanho

Faixas de tamanhos em cm^2	Fase 1: Sem seleção de variáveis		Fase 2: Com seleção de variáveis	
	Precisão na Localização	Erro relativo para o Tamanho	Precisão na Localização	Erro relativo para o Tamanho
entre 1 e 10 cm^2	78,11%	3,86%	98,47%	2,85%
entre 10 e 50 cm^2	55,02%	7,98%	99,12%	4,12%
entre 50 e 500 cm^2	99,53%	4,95%	99,63%	1,40%
maior que 500 cm^2	99,42%	60,67%	99,42%	9,80%

Ao observar os cenários apresentados na literatura é notado que não foram abordados todos os tipos de tamanhos de ruptura (muito pequeno, pequeno, médio e grande) unidos no mesmo conjunto de dados, e por isso é importante salientar que as melhorias aqui apresentadas não apenas superam os resultados da primeira fase deste trabalho, mas também demonstram um desempenho superior quando comparadas com os resultados encontrados na literatura, obtendo uma melhora de 2,45% para predição da localização da ruptura e melhora de 3,84% para predição do tamanho da ruptura, quando comparado com os resultados TING-HAN LIN et al., (2021) que utilizou redes neurais com apenas uma tarefa. Os resultados comparativos na

Tabela ***14*** demonstram a superioridade do uso de redes neurais multitarefas.

Tabela 14: Comparação de Resultados de RNA Simples e RNA Multitarefas

	TING-HAN LIN (2021)	Resultado deste Trabalho
Uso de arquitetura multitarefa	NAO	SIM
Uso de técnicas de seleção de variáveis	NÃO	SIM
Tamanhos de Ruptura	5 cm^2 - 2000 cm^2	1 cm^2 – 1200cm^2
Erro relativo na predição do tamanho	8%	4.26%
Precisão na localização	96,67%	99,12%

Esse progresso substancial valida não apenas a eficácia das abordagens adotadas, mas também posiciona o modelo como um avanço notável no campo, ultrapassando não apenas seus próprios feitos anteriores, mas também as expectativas derivadas da revisão da literatura (TING-HAN LIN (2021)). Essa contribuição destaca a relevância e o impacto do presente trabalho no contexto mais amplo da pesquisa em redes neurais aplicadas à predição de falhas em sistemas.

CAPÍTULO 5. CONCLUSÃO

No dinâmico cenário da aprendizagem de máquina, busca-se constantemente abordagens inovadoras para aprimorar o desempenho e capacidade preditiva dos modelos. A abordagem é a aprendizagem multitarefa (MTL), uma técnica que questiona a tradicional visão de treinar modelos para realizar uma única tarefa. Em vez disso, a MTL propõe treinar simultaneamente um modelo em diversas tarefas complexamente correlacionadas.

A Aprendizagem Multitarefa oferece uma vantagem ao possibilitar que um único modelo compartilhe e generalize informações aprendidas de forma aprofundada a partir de diferentes tarefas complexas, resultando em uma melhora importante na capacidade de generalização eficiente e adaptação do modelo. Esta abordagem torna-se particularmente relevante em cenários desafiadores, nos quais distintos e variados aspectos de um problema multifacetado podem ser abordados simultaneamente com sinergia.

A análise minuciosa do conjunto de teste revelou resultados sólidos e robustos, destacando a eficácia do modelo desenvolvido de forma equilibrada. A precisão de 99,12% na localização demonstra a capacidade do modelo em realizar previsões com boa assertividade.

A porcentagem de 94,36% de acertos com um erro de 0% a 10% na área da ruptura reforça a precisão do modelo, especialmente ao lidar com variações de 1cm^2 até 1200 cm^2. Esses resultados posicionam o modelo como uma ferramenta para predições com boa precisão, mesmo considerando as nuances na área da ruptura.

Ao analisar o Erro Relativo para o tamanho da ruptura, que atinge um valor de 4,26, é evidenciado a capacidade do modelo em manter consistência em suas previsões. Esse baixo índice de erro relativo reflete a eficiência do modelo em generalizar para o Conjunto de Teste, proporcionando uma avaliação mais confiável e precisa.

A análise dos resultados obtidos revela a consecução dos objetivos propostos, indo além das conquistas previamente documentadas na literatura, como evidenciado pelos estudos de MAN

GYUN NA (2004), que apresentou técnicas de RNAs para predição do tamanho e identificação da localização; GEON PIL CHOI (2017) e MAHDI (2019), que introduziram RNAs para predição do tamanho da ruptura; e TING-HAN LIN (2021), que também abordou RNA para predição do tamanho da ruptura, sendo este último o que obteve o melhor resultado.

Outro ponto positivo deste trabalho em relação à literatura é o uso de um conjunto que contempla todos os tipos de tamanho de ruptura (muito pequeno, pequeno, médio e grande). Com a estratégia adotada, que integrou a técnica de seleção de variáveis ao modelo MTDNN, demonstrou um notável potencial na resolução dos desafios relacionados aos problemas ITR e ILR abordados neste estudo.

Esta pesquisa destaca-se pela integração bem-sucedida do Aprendizado Multitarefa combinada com a estratégia de seleção de variáveis na resolução de problemas complexos relacionados às falhas na planta nuclear, notadamente nas fases de identificação do tamanho da ruptura (ITR) e a identificação da localização da ruptura (ILR). Ao adotar um modelo de aprendizagem profundo com múltiplas tarefas (MTDNN), este estudo demonstra como a capacidade do modelo de realizar múltiplas tarefas de forma colaborativa resultou em melhorias substanciais na precisão e eficácia das previsões.

A trajetória que culminou no uso de variáveis importantes revelou-se como a mais eficaz e assertiva. Essa conclusão não apenas simplificou o modelo, reduzindo a dimensionalidade, mas também aprimora significativamente a capacidade preditiva e a generalização para novos conjuntos de dados. A preferência por variáveis criteriosamente selecionadas destaca-se como uma escolha estratégica, consolidando-se como a abordagem mais eficiente na resolução dos desafios propostos.

Os resultados alcançados não apenas destacam a eficácia da abordagem adotada, mas também indicam que há espaço para aprimoramentos adicionais, especialmente explorando outras técnicas do aprendizado profundo. Os resultados obtidos na avaliação do Conjunto de Teste evidenciam a eficácia do modelo desenvolvido, particularmente na previsão de localização e área da ruptura.

Esta pesquisa não apenas contribui para o avanço do conhecimento no campo nuclear, superando os resultados anteriores na literatura, mas também aponta para direções futuras. A contínua exploração e aprimoramento de técnicas de aprendizado profundo com múltiplas tarefas oferecem a perspectiva de soluções ainda mais eficazes para os desafios complexos associados aos problemas ITR e ILR, consolidando assim a relevância e o impacto desta pesquisa no cenário mais amplo da engenharia nuclear.

Nesse contexto, surge uma perspectiva para trabalhos futuros. Primeiramente, é importante avaliar outros métodos de seleção de variáveis, como o PCA e o Kbest. Outra área de pesquisa interessante seria explorar técnicas para otimizar a arquitetura MTDNN, levando em consideração múltiplas tarefas, como o PSO Multitarefa. Além disso, um trabalho futuro seria aplicar o modelo de MTDDN no simulador do LABHIS para avaliar seu desempenho em tempo real. Por fim, é necessária uma exploração mais aprofundada do potencial das Redes Neurais Recorrentes (LSTM) em ambientes de aprendizado multitarefa.

REFERÊNCIAS

ALHOWAIDE A., ALSMADI I., TANG J., **PCA, Random-Forest and Pearson Correlation for Dimensionality Reduction in IoT IDS**, *IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pp. 1-6, 2020

ALMEIDA, J.C. S., **Método de Identificação de Transientes com Abordagem Possibilistica Otimizado por Algoritmo Genético**. Dissertação de M.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil. 2000.

ALVARENGA, M.A.B., MARTINEZ, A.S., SCHIRRU, R., **Adaptive Vector Quantization Optimized by Genetic Algorithm For Real-Time Diagnosis Through Fuzzy Sets**. Nuclear Tecnology, v. 120, pp. 188-197, Dec. 1997.

ARIK, S. Ö., CHRZANOWSKI, M., COATES, A., DIAMOS, G., GIBIANSKY, A., KANG, Y., SHOEYBI, M. **Deep Voice: Real-time Neural Text-to-Speech**. In ICML 2017

BARTAL, Y., LIN, J., UHRIG, R.E., "**Nuclear Power Plant Transient Diagnostics Using Artificial Neural Networks that Allow Don't Know Classifications**", Nuclear Technology, vol. 110, June, pp. 346-449. 1995.

BARTLETT, E. B.; UHRIG, R. E. **Power Plant Status Diagnostics Using Artificial Neural Network**. Nuclear Technology, v. 97, pp. 272–281, 1992.

BASU, A., BARTLETT, E.B. **Detecting Faults in a Nuclear Power Plant by Using a Dynamic Node Architecture Artificial Neural Network**, Nuclear Science and Engineering, vol. 116, pp..313-325, 1994.

BAXTER, J. **A Bayesian/information theoretic model of learning to learn via multiple task sampling**. Machine Learning, 28:7–39, 1997.

BREIMAN, L., **Random Forests.** Machine Learning 45, 5–32, 2001.

CARUANA, R. **Multitask learning: A knowledge-based source of inductive bias**, Proceedings of the Tenth International Conference on Machine Learning. 1993

CARUANA, R. **Multitask Learning. Autonomous Agents and Multi-Agent Systems**, p 95–133, 1998.

CLEVERT D., UNTERTHINER T., HOCHREITER S. **Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)**, arXiv, 2016

COSTA, R. G., **Sistema de Auxílio para o Direcionamento da Atenção no Diagnóstico de Acidentes em Usinas Nucleares Baseado em Inteligência Artificial**. Dissertação de M.Sc., CNEN/IEN, Rio de Janeiro, RJ, Brasil, 2009

DESTERRO. FILIPE S.M., MARCELO C. SANTOS, KELCIO J. GOMES, A. HEIMLICH, ROBERTO SCHIRRU, CLAUDIO MNA. PEREIRA, **Development of a Deep Rectifier Neural Network for dose prediction in nuclear emergencies with radioactive material releases**, Progress in Nuclear Energy, Volume 118, 103110, ISSN 0149-1970, 2020.

DESTERRO F., PINHEIRO V., PEREIRA C., SCHIRRU R., **Prediction of LOCA's break size and location based on random forest and MultiTasking Deep Neural Network**, Nuclear Engineering and Design, Volume 415, 2023.

DIEDERIK P. KINGMA AND JIMMY BA. **Adam: A Method for Stochastic Optimization**. In Proceedings of the International Conference on Learning Representations (ICLR), 2017

ELETRONUCLEAR, **Critérios De Segurança Adotados Para As Usinas Nucleares Angra 1, Angra 2 E Angra 3**, Rio De Janeiro, 2011.

GANIN, Y., & LEMPITSKY, V. **Unsupervised Domain Adaptation by Backpropagation**. In Proceedings of the 32nd International Conference on Machine Learning. (Vol. 37), 2015.

GEOFFREY E. HINTON, SIMON OSINDERO, YEE-WHYE THE, **A Fast Learning Algorithm for Deep Belief Nets**, Neural Computation (2006) 18 (7): 1527–1554.

GEON PIL CHOI, KWAE HWAN YOO, JU HYUN BACK, AND MAN GYUN NA, **Estimation of LOCA Break Size Using Cascaded Fuzzy Neural Networks**, Nuclear Engineering and Technology, Volume 49, Issue 3, Pages 495-503. 2017.

GLOROT, X., BORDES, A. & BENGIO, Y. **Deep Sparse Rectifier Neural Networks**. Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, PMLR 15:315-323, 2011.

GIRSHICK, R. **Fast R-CNN**. In Proceedings of the IEEE International Conference on Computer Vision (pp. 1440–1448). 2015.

J. LIN, W. FANG, Y. WANG AND J. CHEN, "**FSF MUSIC for Joint DOA and Frequency Estimation and Its Performance Analysis**," in IEEE Transactions on Signal Processing, vol. 54, no. 12, pp. 4529-4542. 2006.

JEZEQUEL, FABIENNE & LAMOTTE, JEAN-LUC & SAID, ISSAM. **Estimation of numerical reproducibility on CPU and GPU**. 2015.

JÜRGEN SCHMIDHUBER, ***Deep Learning in Neural Networks: An Overview***, Neural Networks, Volume 61, Pages 85-117, ISSN 0893-6080, 2015.

K. H. YOO, Y. D. KOO, J. H. BACK AND M. G. NA, **Identification of LOCA and Estimation of Its Break Size by Multiconnected Support Vector Machines**, *IEEE Transactions on Nuclear Science*, vol. 64, no. 10, pp. 2610-2617. 2017.

LECUN, Y.A., BOTTOU, L., ORR, G.B., MÜLLER, KR. **Efficient BackProp**. In: Montavon, G., Orr, G.B., Müller, KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 7700. Springer, Berlin, Heidelberg, 2012.

MAHDI SAGHAFI, MOHAMMAD B. GHOFRANI, **Real-time estimation of break sizes during LOCA in nuclear power plants using NARX neural network**, Nuclear Engineering and Technology, Volume 51, Issue 3, Pages 702-708. 2019.

MAN GYUN NA, SUN HO SHIN, DONG WON JUNG, SOONG PYUNG KIM, JI HWAN JEONG, BYUNG CHUL LEEC, **Estimation of break location and size for loss of coolant accidents using neural networks**, Nuclear Engineering and Design Volume 232, Issue 3, Pages 289-300. 2004

MCCULLOCH, W.S., PITTS, W. **A logical calculus of the ideas immanent in nervous activity**. Bulletin of Mathematical Biophysics 5, 115–133. 1943.

MEDEIROS, J. A. C. C., **Enxames de Partículas como Ferramenta de Otimização em Problemas Complexos da Engenharia Nuclear**. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2005.

MINSKY, M., AND PAPERT, S. **Review of 'Perceptrons: An Introduction to Computational Geometry**. IEEE Transactions on Information Theory, vol. 15, no. 6, pp. 738-739, 1969.

MÓL CARLOS, **Um Sistema de Identificação de Transientes com Inclusão de Ruídos e Indicação de Eventos Desconhecidos**, Tese Submetida Ao Corpo Docente Da Coordenação Dos Programas De Pós-graduação De Engenharia Da Universidade Federal Do Rio De Janeiro, 2003.

NICOLAU, A. S., **Computação Quântica e Inteligência de Enxames Aplicados na Identificação de Acidentes de uma Usina Nuclear PWR**, Dissertação de M.Sc., COPPE/UFRRJ, Rio de Janeiro, RJ, Brasil, 2010.

NICOLAU, A. S., **Algoritmo Evolucionário de Inspiração Quântica Aplicado na Otimização de Problemas da Engenharia Nuclear**. Tese de D. Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2014.

NICOLAU, A. S., SCHIRRU, R., MENESES, A. A. M. "**Quantum evolutionary algorithm applied to transient identification of a nuclear power plant**", Progress in Nuclear Energy, 53, 86-91. 2011.

QUINLAN, J. R., RIVEST, R. **Information and Computation**, (80), 227-248, 1989.

QUINLAN, J. R. **C4.5: Programs for Machine Learning**, San Mateio, CA: Morgan Kaufmann, 1993.

RAMSUNDAR, B., KEARNES, S., RILEY, P., WEBSTER, D., KONERDING, D., & PANDE, V. Massively Multitask Networks for Drug Discovery, 2015.

ROSENBLATT, F., **The Perceptions: A Probabilistic Model for Information Storage and Organization in the Brain**. Psychol. Rev., v. 65, pp. 386-408, 1958.

ROSENBLATT, F., **Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms**. New York, Spartan Bokks., 1962.

Ruder, S. **An overview of multi-task learning in deep neural networks**. arXiv preprint arXiv:1706.05098, 2017.

PEREIRA, C. M. N. A, SCHIRRU R, MARTINEZ A.S., "**Learning an Optimized Classification System from a Data Base of Time Series Patterns Using Genetic Algorithms**". In: Ebecken, N.S.S., Data Mining, South Hampton, United Kingdom, WIT Computational Mechanics Publication, 1998.

PINHEIRO, V. H. C.; SCHIRRU, R. **Genetic Programming Applied to the Identification of Accidents of a PWR Nuclear Power Plant**. Annals of Nuclear Energy, v. 124, pp. 335-341, 2019.

PINHEIRO, V. H. C.; SANTOS, M. C.; DESTERRO, F. S. M.; SCHIRRU, R.; PEREIRA, C. M. D. N. A. **Nuclear Power Plant accident identification system with "don't know" response capability: Novel deep learning-based approaches**. Annals of Nuclear Energy, v. 137, p. 107, 2019.

SAEYS Y., ABEEL T., VAN DE PEER Y., **Robust Feature Selection Using Ensemble Feature Selection Techniques**. In: Daelemans W., Goethals B., Morik K. (eds) Machine Learning and Knowledge Discovery in Databases. Lecture Notes in Computer Science, vol 5212. Springer, Berlin, Heidelberg, 2008.

SANTOS, M. C.; PINHEIRO, V. H. C.; DESTERRO, F. S. M.; AVELLAR, R. K.; SCHIRRU, R.; NICOLAU, A. S.; LIMA, A. M. M. **Deep rectifier neural network applied to the accident identification problem in a PWR nuclear power plant**. Annals of Nuclear Energy, v. 133, pp. 400-408, 2019.

TING-HAN LIN, CHING CHEN, SHUN-CHI WU, TE-CHUAN WANG, YUH-MING FERNG, **Localization and size estimation for breaks in nuclear power plants**, Nuclear Engineering and Technology, 2021.

TOM MITCHELL, MCGRAW HILL, **Machine Learning**, CHAPTER 3 - Decision Tree Learning, 1997.

VICTOR HENRIQUE CABRAL PINHEIRO, ROBERTO SCHIRRU, **Genetic programming applied to the identification of accidents of a PWR nuclear power plant**, Annals of Nuclear Energy, Volume 124, 2019, Pages 335-341,

X. Glorot, Y. Bengio, **Understanding the Difficulty of Training Deep Feedforward Neural Networks**, Proceedings of the thirteenth international conference on artificial intelligence and statistics, p.p 249-256, 2010.

Zhang, Z., Luo, P., Loy, C. C., & Tang, X. (2014). **Facial Landmark Detection by Deep Multi-task Learning**. In European Conference on Computer Vision (pp. 94–108).

ZHOU YANGPING, ZHAO BINGQUAN, WU DONGXIN, **Application of genetic algorithms to fault diagnosis in nuclear power plants**, Reliability Engineering & System Safety, Volume 67, Issue 2, 2000, Pages 153-160.

WIDROW, B., Hoff M. E, **Adaptive Switching Circuits**, IRE WESCON Convention Record , pp. 96-104, 1960.

XUAN, YI AND SI, WEIGUO AND ZHU, JIONG AND SUN, ZHIQING AND ZHAO, JIAN AND XU, MINGJIE AND XU, SHOULIANG, **Multi-Model Fusion Short-Term Load**

Forecasting Based on Random Forest Feature Selection and Hybrid Neural Network, in IEEE Access, vol. 9, pp. 69002-69009, 2021.

XUE, Y., LIAO, X., CARIN, L., & KRISHNAPURAM, B. **Multi-Task Learning for Classification with Dirichlet Process Priors**. Journal of Machine Learning Research, 8, 35–63, 2007.

ANEXO I: TABELA COM VARIÁVEIS COLETADAS DURANTES AS SIMULAÇÕES NO LABINS

Description	Norm	Description	Norm	Description	Norm	Description	Norm
Time	H	RHR return temp	C	PRZ level setpoint	%	Steam line 1 flow	T/hr
Power range percent power	%	H2 concentration	%	PRZ pressure (narrow range)	kg/cm*2	Aux feedwater flow 1	m*3/hr
Intermediate range neutron level	A	RHR return flow	m*3/hr	Loop 3 average temp	C	Atmospheric steam dump v/v posit.	%
Source range neutron level	CPS	Volume control tank level	%	Loop 2 average temp	C	Aux feedwater flow 2	m*3/hr
Axial offset	%	Volume control tank press	kg/cm*2	Loop 1 average temp	C	FW 3 bypass valve posit.	%
Start-up rate	DPM	RCP seal injection flow	m*3/hr	PRZ temp	C	FW 3 control valve posit.	%
Reference temp	Cº	RCP seal No.1 return flow	m*3/hr	PRZ level	%	FW 2 bypass valve posit.	%
Average temp (auctioneered high)	Cº	Boric acid flow rate	L/sec	PRZ pressure (wide range)	kg/cm*2	FW 2 control valve posit.	%
Fuel temp	Cº	Make-up water flow rate	L/sec	PRZ spray flow	m*3/hr	FW 1 bypass valve posit.	%
Net reactivity	dk/k (%)	Charging flow control v/v posit	%	S/G 3 level(wide)	%	FW 1 control valve posit.	%
lodine concentration	%	NRHX outlet temp	C	S/G 2 level(wide)	%	Aux feedwater flow 3	m*3/hr
Xenon concentration	Pcm	Accumulator pressure	kg/cm*2	S/G 1 level(wide)	%	Main steam header press setp.	kg/cm*2
Overpower delta-temp	%	Charging line output temp	‘C	S/G 3 pressure	kg/cm*2	Main steam header pressure	kg/cm*2
Overtemp delta-temp	%	RHX outlet temp	C	S/G 2 pressure	kg/cm*2	Feedwater temp	C
Source range alarm signal	-	Reactor vessel water level	%	S/G 1 pressure	kg/cm*2	Main steam flow	%
Temp mismatch	deg C	Core outlet temp	%	S/G 3 level(narrow)	%	Secondary radiation	microC/c
Containment sump level	m	Pressurizer relief tank temp	C	S/G 2 level(narrow)	%	Feedwater pump outlet press	kg/cm*2
Refueling water storage tank level	%	PRT pressure	kg/com*2	S/G 1 level(narrow)	%	HP turbine control valve	%
Component cooling water temp	Cº	Loop 3 cold-leg temp	C	Loop 3 flow	%	LP heater outlet temp	C
RHR HX bypass valve position	%	Loop 2 cold-leg temp	C	Loop 2 flow	%	Condenser hot well temp	C
Instrument air comp pressure	kg/cm*2	Loop 1 cold-leg temp	C	Loop 1 flow	%	Aux FW storage tank level	%
Containment radiation	mrem/hr	Loop 3 hot-leg temp	‘C	Feedwater line 3 flow	T/hr	Running voltage	V
Containment relative humidity	%	Loop 2 hot-leg temp	‘C	Feedwater line 2 flow	T/hr	Voltage	KV
RHR HX discharge valve	%	Loop 1 hot-leg temp	‘C	Feedwater line 1 flow	T/hr	Generator output	Mwe
Containment temp	Cº	Proportional heaters	%	Steam line 3 flow	T/hr		
Containment pressure	kg/cm*2	PRZ delta level	%	Steam fine 2 flow	T/hr		