

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE
MINAS GERAIS - *CAMPUS* FORMIGA
ENGENHARIA ELÉTRICA

Jean Fonseca da Silva

**UTILIZAÇÃO DE REDES NEURAIIS PARA DETECÇÃO E
LOCALIZAÇÃO DE FALTAS EM LINHAS DE TRANSMISSÃO**

Formiga - MG

2022

JEAN FONSECA DA SILVA

**UTILIZAÇÃO DE REDES NEURAIS PARA DETECÇÃO E
LOCALIZAÇÃO DE FALTAS EM LINHAS DE TRANSMISSÃO**

Trabalho de conclusão de curso apresentado ao Curso de Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais - *Campus Formiga* para a obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Me. José Antônio Moreira de Rezende

Formiga - MG
2022

Silva, Jean Fonseca da
S586u Utilização de Redes Neurais para Detecção e Localização de Faltas em Linhas
de Transmissão
/ Jean Fonseca da Silva -- Formiga : IFMG, 2022.
63p. : il.

Orientador: Prof. M.e José Antônio Moreira de Rezende
Trabalho de Conclusão de Curso – Instituto Federal de Educação,
Ciência e Tecnologia de Minas Gerais – *Campus* Formiga.

1. Redes Neurais. 2. Linha de Transmissão. 3. Faltas em Sistemas Elétricos.
4. Redes Neurais Recorrentes. 5. LSTM. I. Rezende, José Antônio Moreira de.
II. Título.

CDD 621.3

JEAN FONSECA DA SILVA

**UTILIZAÇÃO DE REDES NEURAIS PARA DETECÇÃO E LOCALIZAÇÃO DE
FALTAS EM LINHAS DE TRANSMISSÃO**

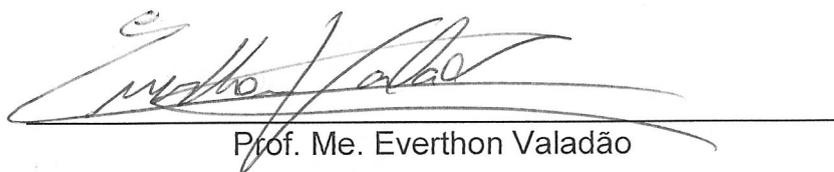
Trabalho de Conclusão de Curso
apresentado ao Curso de Engenharia
Elétrica do Instituto Federal de Minas
Gerais como requisito para obtenção do
Título de Bacharel em Engenharia
Elétrica.

Avaliado em: 06 de julho de 2022.

Nota: 78,0

BANCA EXAMINADORA


Prof. Me. José Antonio Moreira de Rezende (Orientador)


Prof. Me. Everthon Valadão


Prof. Dr. André Roger Rodrigues

Este trabalho é dedicado ao meus pais e a minha família
que sempre me apoiaram na minha caminhada.

AGRADECIMENTOS

Primeiramente gostaria de agradecer a Deus, aos meus pais por todo apoio dado, a minha namorada pelo companheirismo, aos meus professores e funcionários do IFMG que fizeram parte da minha caminhada, e principalmente ao meu orientador que me apoiou além do TCC, em outros projetos durante meus estudos.

“Cem vezes todos os dias lembro a mim mesmo que minha vida interior e exterior, depende dos trabalhos de outros homens, vivos ou mortos, e que devo esforçar-me a fim de devolver na mesma medida que recebi.”

Albert Einstein

RESUMO

Este trabalho propõe o uso de duas redes neurais Long Short-Term Memory (LSTM), uma para classificação de faltas e outra para localização de faltas, em uma linha de transmissão trifásica. Os valores de corrente e de tensão ao longo da linha foram obtidos a partir de simulações computacionais usando o programa de transientes eletromagnéticos ATP. A estrutura de simulação consistiu em uma linha de transmissão trifásica de 100 km com falhas simuladas a cada quilômetro. Conseqüentemente, 693 amostras foram geradas e tratadas por meio da Transformada Discreta de Fourier que, por sua vez, foram aplicadas às entradas das duas redes neurais supracitadas e desenvolvidas em Python no contexto da biblioteca Keras. A principal razão para a utilização da rede neural recorrente LSTM foi a característica temporal dos dados de corrente e tensão obtidos, que requerem uma rede neural que considere os resultados obtidos em instantes de tempo anteriores ao longo das suas tarefas de treinamento e de classificação. Com efeito, na rede neural de classificação do tipo de falha foi alcançada uma precisão de treinamento de 88,88% e uma precisão de teste de 84,57%. Por sua vez, a rede neural de classificação do segmento da falha obteve uma precisão de 61,95% no treinamento e 50,00% no teste (ao procurar classificar o ponto exato da seção da linha em que ocorreu a falta). Essa precisão de classificação foi aumentada para 90,27% quando a seção da linha de transmissão foi relaxada em um quilômetro a montante e a jusante.

Palavras-chave: LSTM. Redes neurais. Linha de transmissão. Faltas em sistemas elétricos.

ABSTRACT

This work proposes the use of two Long Short-Term Memory (LSTM) neural networks, one for fault classification and other for fault location, on a three-phase transmission line. Current and voltage values were obtained from computer simulations using the ATP electromagnetic transient program. The simulation structure consisted of a 100 km three-phase transmission line with simulated faults in every kilometer. Consequently, 693 samples were generated and treated by means of the Discrete Fourier Transform and applied to the inputs of the two aforementioned neural networks built in Python on a Keras library context. The main reason for using the LSTM recurrent neural network was the temporal characteristic of the current and voltage data obtained, that requires a neural network which considers results obtained from previous instant times during your training and classification tasks. As a result, in the fault-type classification neural network a training accuracy of 88.88% and a test accuracy of 84.57% was achieved. In your turn, the fault-segment classification neural network obtained a 61.95% accuracy in the training and 50.00% in the test (in the exact line section point). This classification accuracy was increased to 90.27% when the transmission line section was relaxed by one kilometer upstream and downstream.

Keywords: LSTM. Neural Networks. Transmission Line. Faults in electrical systems.

LISTA DE ILUSTRAÇÕES

Figura 1 – Comportamento de ondas quando $Z_2 > Z_0$	15
Figura 2 – Comportamento de ondas quando $Z_2 = Z_0$	15
Figura 3 – Comportamento de ondas quando $Z_2 < Z_0$	16
Figura 4 – Circuito elétrico completo para linhas de transmissão.	18
Figura 5 – Modelo π para linhas curtas.	19
Figura 6 – Modelo π para linhas médias.	20
Figura 7 – Representação de um neurônio	23
Figura 8 – Rede neural Perceptron de camada única	24
Figura 9 – Representação da função de ativação degrau.	25
Figura 10 – Representação da função de ativação sigmóide.	25
Figura 11 – Representação função de ativação tangente hiperbólica.	26
Figura 12 – Representação gráfica porta lógica XOR ilustrando as condições de classifica- ção de saída da porta lógica XOR	27
Figura 13 – Representação Rede Neural <i>Multilayer Perceptron</i> para o problema XOR.	27
Figura 14 – Camadas básica de uma rede <i>Multilayer Perceptron</i>	29
Figura 15 – Representação da busca de mínimos globais	30
Figura 16 – Célula-exemplo Rede Neural Recorrente	31
Figura 17 – Célula-exemplo Rede Neural Recorrente	33
Figura 18 – Célula exemplo LSTM	34
Figura 19 – Primeira parte da célula LSTM	35
Figura 20 – Segunda parte da célula LSTM	35
Figura 21 – Saída célula LSTM	36
Figura 22 – Topologia para a simulação das faltas.	37
Figura 23 – Dados comuns para configuração dos tipos de linha.	38
Figura 24 – Configuração de modelo e dados de linhas aéreas.	39
Figura 25 – Configuração do tipo de modelo.	40
Figura 26 – Disposição física dos cabos.	41
Figura 27 – Disposição física do modelo.	41
Figura 28 – Exemplo de linha de código do ATP com falta.	42
Figura 29 – Interface TensorBoard	44
Figura 30 – Valores que compõem os setups a partir de cenários de variação dos hiperpa- râmetros para as redes neurais.	45
Figura 31 – Trecho de um cartão do ATP.	46
Figura 32 – Interface PlotXY.	47
Figura 33 – Arquivo de dados da execução do ATP.	47
Figura 34 – Resultados extraídos do arquivo de dados da execução do ATP.	48
Figura 35 – Comportamento da tensão da rede em uma falta da fase A para a terra	49
Figura 36 – Comportamento da tensão da rede em uma falta da fase B para a terra	49

Figura 37 – Comportamento da tensão da rede em uma falta da fase C para a terra	50
Figura 38 – Comportamento da tensão da rede em uma falta das fases AB para a terra . .	50
Figura 39 – Comportamento da tensão da rede em uma falta das fases AC para a terra . .	51
Figura 40 – Comportamento da tensão da rede em uma falta das fases BC para a terra . .	51
Figura 41 – Comportamento da tensão da rede em uma falta das fases ABC para a terra .	52
Figura 42 – Comportamento da tensão fase A durante uma falta monofásica	53
Figura 43 – Comportamento da tensão fase A durante uma falta bifásica	53
Figura 44 – Comportamento da tensão fase A durante uma falta trifásica	54
Figura 45 – Comportamento da tensão fase A durante uma falta BC com a terra fase . .	54
Figura 46 – Gráfico da rede neural de localização da falta.	57
Figura 47 – Gráfico da rede neural de detecção de falta.	57

LISTA DE TABELAS

Tabela 1	–	Combinações da porta lógica XOR	26
Tabela 2	–	Parâmetros da Fonte equilibrada.	37
Tabela 3	–	Parâmetros da Fonte à direita da LT.	38
Tabela 4	–	10 melhores hiperparâmetros encontrados pelo KerasTuner para rede de detecção da distância da falta	55
Tabela 5	–	10 melhores hiperparâmetros encontrados pelo KerasTuner para rede de detecção do tipo de falta	55
Tabela 6	–	Busca dos melhores hiperparâmetros com passos de aprendizagem de 0.1 e 1 para rede de distância da falta.	56
Tabela 7	–	Busca dos melhores hiperparâmetros com passos de aprendizagem de 0.1 e 1 para rede de detecção do tipo de falta.	56
Tabela 8	–	Exemplos de detecção de distância	58
Tabela 9	–	Precisão para cada "range" de distância	59

LISTA DE ABREVIATURAS E SIGLAS

ABNT Associação Brasileira de Normas Técnicas

SEP Sistema Elétrico de Potência

IFMG Instituto Federal de Minas Gerais

ATP Alternative Transient Program

LT Linha de Transmissão

RNA Rede Neural Artificial

LSTM *Long-Short Term Memory*

SUMÁRIO

1	INTRODUÇÃO	13
2	FUNDAMENTAÇÃO TEÓRICA	14
2.1	Linhas de Transmissão	14
<i>2.1.1</i>	<i>Ondas viajantes</i>	<i>14</i>
<i>2.1.2</i>	<i>Representação de Linhas de Transmissão</i>	<i>17</i>
<i>2.1.3</i>	<i>Linhas curtas</i>	<i>18</i>
<i>2.1.4</i>	<i>Linha média</i>	<i>20</i>
2.2	Transformada de Fourier	21
2.3	Redes Neurais Artificiais	22
<i>2.3.1</i>	<i>Redes neurais perceptron de camada única</i>	<i>23</i>
<i>2.3.2</i>	<i>Redes neurais perceptron de múltiplas camadas</i>	<i>26</i>
<i>2.3.3</i>	<i>Redes neurais recorrentes</i>	<i>31</i>
<i>2.3.4</i>	<i>LSTM - Long-Short Term Memory</i>	<i>33</i>
3	MATERIAIS E MÉTODOS	37
3.1	Simulação	37
3.2	Aquisição dos dados	42
3.3	Base de Dados e Rede Neural	43
4	RESULTADOS E DISCUSSÕES	46
5	CONCLUSÃO	60
	REFERÊNCIAS	61

1 INTRODUÇÃO

De acordo com Estado de Minas (2019), o consumo de energia elétrica no Brasil vem crescendo 2,8% ao ano desde 2017 e continuará crescendo a essa proporção até 2040, resultando em um crescimento total no período de 89%.

Devido a um mundo cada vez mais conectado e dependente do uso da energia elétrica, se faz necessário um maior cuidado na proteção de sistemas elétricos de potência contra os efeitos danosos de uma falta, já que se tem cada vez mais complexidade, interligações e crescimento dos mesmos (COURY; OLESKOVICZ; GIOVANINI, 2007).

Estas faltas podem ser causadas pela ruptura dos cabos das linhas elétricas de distribuição e/ou de transmissão de energia elétrica (faltas série) ou por curtos-circuitos envolvendo a terra, as fases ou entre as fases e a terra (faltas em derivação). No caso deste trabalho, o foco segue nas faltas entre os condutores das fases e a terra. As possíveis faltas com a terra em sistemas elétricos trifásicos podem ser:

- Fase-terra;
- Bifásica-terra
- Trifásica-terra.

As faltas podem causar correntes elevadas que caso sejam sustentadas provocam o aquecimento dos condutores e a deterioração irreversível de equipamentos, além também de causar variações de tensão, podendo gerar quedas de tensão muito elevadas em algumas fases e por vezes elevações em outras (MOREIRA, 2010). Para diminuir os problemas causados devido a essas faltas e aumentar a confiabilidade do sistema, é necessário detectá-las e localizá-las para que se consiga rapidamente solucioná-los e restaurar a operação aquele trecho de rede ao sistema.

As redes neurais artificiais possuem uma capacidade de aprender por intermédio de um método de treinamento denominado *backpropagation*, onde as entradas da rede neural são atualizadas baseadas no erro resultante da entrada anterior frente a saída desejada. Desta forma, as mesmas também são capazes de generalizar o conhecimento adquirido, podendo estimar soluções que não eram conhecidas, (SILVA; SPATTI; FLAUZINO, 2010).

Observada a capacidade das redes neurais de aprender e em seguida, de reconhecer padrões (SILVA; SPATTI; FLAUZINO, 2010), essa técnica poderia ser aplicada para a detecção e a localização das faltas, já que cada falta produz um padrão específico de comportamento quando ocorrem.

Com essa motivação, esse trabalho propõe um algoritmo que analisa os dados provenientes de simulação no ATP (*Alternative Transient Program*) e após o devido tratamento, estes serão enviados como entrada de uma rede neural artificial multicamadas.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta a teoria básica sobre linhas de transmissão (LTs) de energia elétrica e sobre métodos computacionais destinados à localização de pontos que manifestam comportamentos anômalos e que ocasionam problemas de qualidade de energia elétrica em uma localidade.

2.1 Linhas de Transmissão

2.1.1 Ondas viajantes

Considere uma onda de tensão (V) e uma onda de corrente (I_0), chamadas de ondas incidentes, que percorrem uma linha de transmissão a uma velocidade (v) em direção ao receptor. Quando uma onda viajante encontra um ponto de descontinuidade, como é o caso de um circuito aberto, uma parte da onda é refletida e direcionada de volta à fonte e outra parte é transmitida. Dependendo da forma da terminação da linha, pode-se dar origem a ondas refletidas que viajam na linha com a mesma velocidade que a onda incidente (SILVA, 2003). Em cada ponto ao longo da linha de transmissão, o valor da tensão e da corrente será sempre igual a soma das ondas incidentes e refletidas, como mostra as equações 2.1 e 2.2 (FUCHS, 1977).

$$U = U_d \pm U_r \quad (2.1)$$

$$I = I_d \pm I_r \quad (2.2)$$

onde:

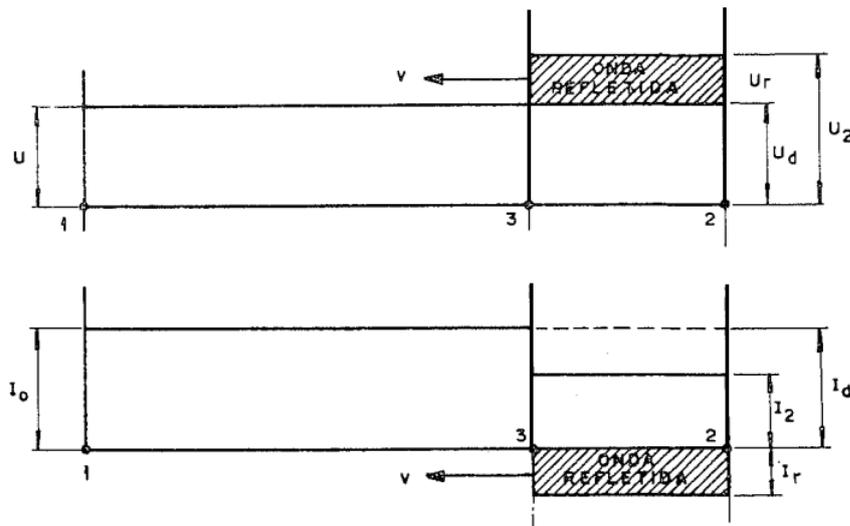
- U_d é a tensão da rede em regime permanente [V];
- U_r é a tensão da onda refletida [V];
- U é o resultado da sobreposição entre as ondas refletida e em regime permanente [V];
- I_d é a corrente da rede em regime permanente [A];
- I_r é a corrente da onda refletido [A];
- I é o resultado da sobreposição entre as ondas refletida e em regime permanente [A].

Existem dois parâmetros importantes em uma linha de transmissão, a saber:

- Z_0 é a impedância natural, ou impedância característica da linha de transmissão que não depende do comprimento da mesma, somente do meio em que se encontra e das dimensões físicas como a distância entre condutores e o raio do mesmo [Ω];

- Z_2 é a impedância no receptor $[\Omega]$.
- $Z_2 > Z_0$: A onda de tensão refletida possui a mesma polaridade que a onda incidente, assim a tensão resultante será maior que a incidente. A onda de corrente refletida no entanto terá sua polaridade invertida tendo assim uma corrente resultante menor que a incidente (Figura 1).

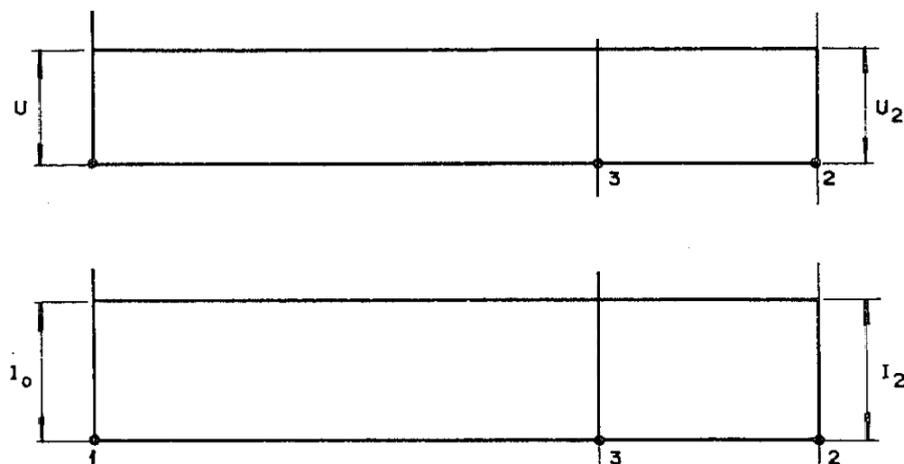
Figura 1 – Comportamento de ondas quando $Z_2 > Z_0$.



Fonte: Fuchs (1977).

- $Z_2 = Z_0$: Tem-se o casamento de impedância e o sistema se encontra apenas com a onda de regime permanente. Neste caso não incidem no sistema ondas refletidas (Figura 2).

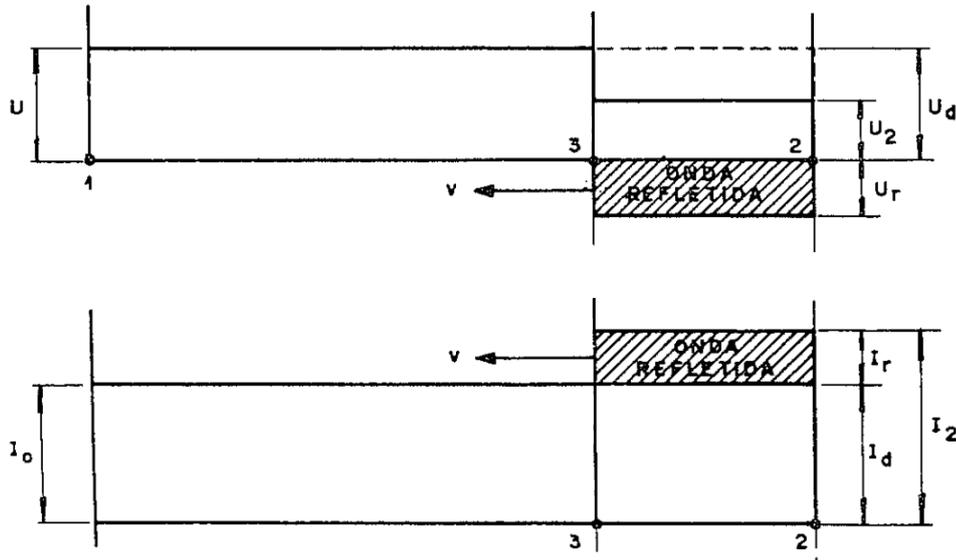
Figura 2 – Comportamento de ondas quando $Z_2 = Z_0$.



Fonte: Fuchs (1977).

- $Z_2 < Z_0$: A onda de tensão reflete com sinal oposto e a onda de corrente reflete com o mesmo sinal da onda incidente (Figura 3).

Figura 3 – Comportamento de ondas quando $Z_2 < Z_0$.



Fonte: Fuchs (1977).

As ondas refletidas têm as mesmas propriedade das ondas incidentes, assim:

$$\frac{U_d}{I_d} = \frac{U_r}{I_r} = \sqrt{\frac{L}{C}} = Z_0 \quad (2.3)$$

Assim em qualquer ponto de uma linha de transmissão tem-se:

$$\frac{U}{I} = \frac{U_d + U_r}{I_d + I_r} \neq Z_0 \quad (2.4)$$

Conhecidos a impedância natural de uma linha e o valor da resistência terminal, é possível determinar o valor da amplitude das ondas refletidas em função das ondas incidentes (FUCHS, 1977):

$$\frac{U_2}{I_2} = \frac{U_d + U_r}{I_d + I_r} = Z_2 \quad (2.5)$$

Assim:

$$I_r = -\frac{U_r}{Z_0} \quad (2.6)$$

$$I_d = \frac{U_d}{Z_0} \quad (2.7)$$

Substituindo as Equações 2.6 e 2.7 na Equação 2.5 tem-se:

$$U_r = \frac{Z_2 - Z_0}{Z_2 + Z_0} \times U_d \quad (2.8)$$

e:

$$I_r = \frac{Z_0 - Z_2}{Z_2 + Z_0} \times I_d \quad (2.9)$$

Os termos que acompanham U_d e I_d são os coeficientes de reflexão da tensão (K_{ru}) e da corrente (K_{ri}) respectivamente. Assim:

$$K_{ru} = \frac{Z_2 - Z_0}{Z_2 + Z_0} \quad (2.10)$$

e:

$$K_{ri} = \frac{Z_0 - Z_2}{Z_2 + Z_0} \quad (2.11)$$

Se $Z_0 = Z_2$, o numerador da fração é zero, indicando assim que não existe onda refletida como dito anteriormente.

Esses coeficientes de reflexão descrevem o comportamento transitório das ondas viajantes quando da ocorrência de um surto na linha, energização, falta, etc. (SILVA, 2003).

2.1.2 Representação de Linhas de Transmissão

Para a obtenção da tensão e corrente elétrica de uma linha de transmissão em qualquer ponto de sua extensão, deve-se utilizar as equações gerais das linhas de transmissão, que são utilizadas para o cálculo da tensão e da corrente elétrica da linha em regime permanente e com excitação alternada senoidal. O circuito elétrico equivalente obtido por emprego das Equações 2.12 e 2.13 está representado na Figura 4, (GLOVER; SARMA; OVERBYE, 2012).

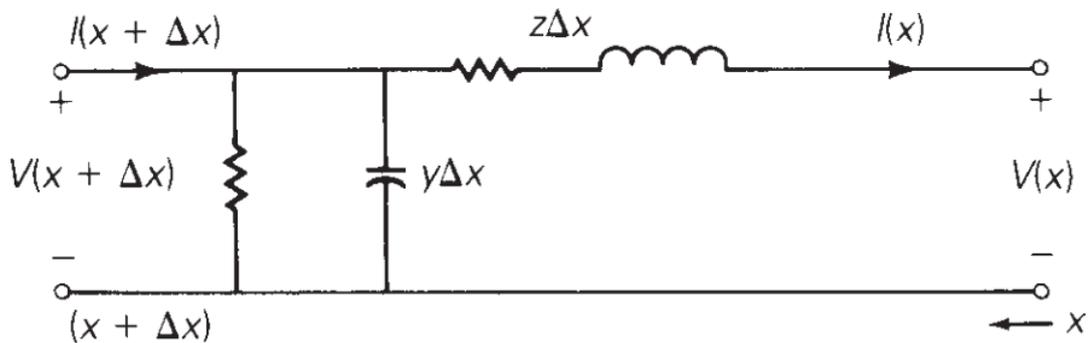
A tensão em qualquer ponto da linha de transmissão pode ser obtida pela Equação 2.12:

$$V(x) = \left(\frac{V_R + Z_0 I_R}{2} \right) e^{\gamma x} + \left(\frac{V_R - Z_0 I_R}{2} \right) e^{-\gamma x} \quad (2.12)$$

e para corrente:

$$I(x) = \left(\frac{V_R + Z_0 I_R}{2Z_0} \right) e^{\gamma x} - \left(\frac{V_R - Z_0 I_R}{2Z_0} \right) e^{-\gamma x} \quad (2.13)$$

Figura 4 – Circuito elétrico completo para linhas de transmissão.



Fonte: Extraída de Glover, Sarma e Overbye (2012).

onde:

- V_R é a tensão no terminal receptor da linha de transmissão no instante de tempo igual a zero [V];
- Z_0 é a impedância natural da linha [Ω];
- I_R é a corrente no terminal receptor da linha de transmissão no instante de tempo igual a zero [A];
- $\gamma = \sqrt{z\bar{y}}$ é a constante de propagação [m^{-1}];
- z é a impedância série [Ω/m];
- y é a admitância shunt [S/m];
- x é o comprimento da linha [m].

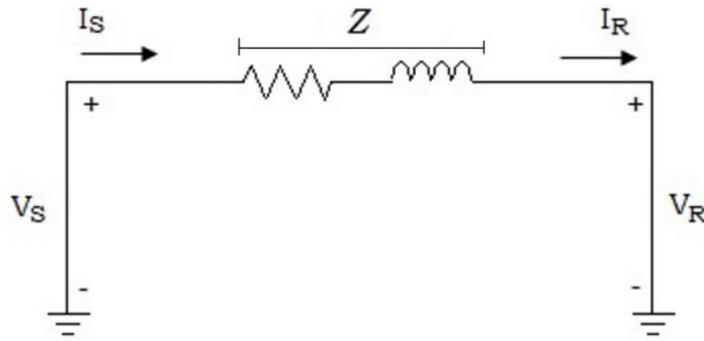
Estas equações podem descrever qualquer linha de transmissão independentemente do seu comprimento elétrico ou físico. Porém, a depender do comprimento da LT, alguns parâmetros não irão influenciar significativamente no cálculo da tensão e corrente da linha. De acordo com Grainger (1999), uma linha é considerada curta em um comprimento menor que 80 km, uma linha média são aquelas com comprimento entre 80 km e 240 km. Para comprimentos maiores que 240 km, no geral, deve-se utilizar as equações gerais com parâmetros distribuídos conforme Equações 2.12 e 2.13 (GRAINGER, 1999).

2.1.3 Linhas curtas

Em linhas curtas, a capacitância *shunt* é tão pequena que pode ser omitida inteiramente com pouca perda de precisão nos valores de tensão e de corrente comparados àqueles obtidos

com emprego das equações totais. É preciso considerar apenas os parâmetros distribuídos de resistência e indutância série para representação do comprimento físico real da LT. Assim a linha pode ser representada pelo circuito elétrico equivalente mostrado na Figura 5.

Figura 5 – Modelo π para linhas curtas.



Fonte: Diefenthaler (2019).

Conforme pode-se observar na Figura 5, tem-se um circuito em série da resistência e indutância da linha representado por uma impedância (Z):

$$Z = (r + j\omega L)l \quad (2.14)$$

onde:

- Z é a impedância da linha de transmissão [Ω];
- l é o comprimento da linha [km];
- r é a resistência por unidade de comprimento da linha de transmissão [Ω/km];
- j é a componente complexa ($\sqrt{-1}$);
- ωL é a reatância indutiva da linha de transmissão [Ω];
- ω é a velocidade angular [rad/s];
- L é a indutância da linha de transmissão [H/km].

As equações que descrevem o modelo equivalente para LTs curtas são descritas nas Equações 2.15 e 2.22 e de forma matricial na equação 2.17:

$$V_S = V_R + ZI_R \quad (2.15)$$

$$I_S = I_R \quad (2.16)$$

$$\begin{bmatrix} V_S \\ I_S \end{bmatrix} = \begin{bmatrix} 1 & Z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_R \\ I_R \end{bmatrix} \quad (2.17)$$

2.1.4 Linha média

Diferente do circuito elétrico equivalente para LTs curtas, no modelo de linha média o circuito elétrico equivalente deve considerar a capacitância concentrada na admitância total da linha (Y), conforme equação 2.18 (DIEFENTHALER, 2019).

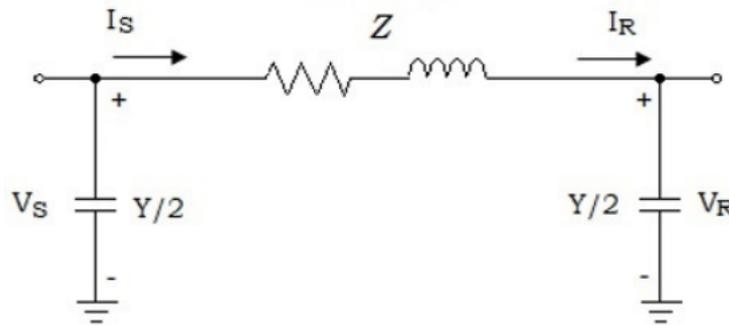
$$Y = j\omega Cl \quad (2.18)$$

onde:

- Y é a admitância total da linha [S];
- C é a capacitância [F/km];
- l é o comprimento da linha [km].

A Figura 6 apresenta o circuito elétrico equivalente e observa-se uma diferença com relação a Figura 5 em relação a representação de uma linha curta com relação a linha média. Conforme dito no parágrafo anterior é acrescentada uma admitância Y no circuito.

Figura 6 – Modelo π para linhas médias.



Fonte: Diefenthaler (2019).

Aplicando a segunda Lei de Kirchhoff (LKT) no circuito da Figura 6:

$$V_S = \left(\frac{ZY}{2} + 1 \right) V_R + ZI_R \quad (2.19)$$

$$I_S = V_S \frac{Y}{2} + V_R \frac{Y}{2} + I_R \quad (2.20)$$

Substituindo a equação 2.20 na Equação 2.19 tem-se:

$$I_S = YV_R \left(\frac{ZY}{4} + 1 \right) + \left(\frac{ZY}{2} + 1 \right) I_R \quad (2.21)$$

Assim as Equações 2.19 e 2.21 podem ser descritas matricialmente conforme o sistema de Equações 2.22 e 2.23:

$$I_S = I_R \quad (2.22)$$

$$\begin{bmatrix} V_S \\ I_S \end{bmatrix} = \begin{bmatrix} \frac{ZY}{2} + 1 & Z \\ Y \left(\frac{ZY}{4} + 1 \right) & \frac{ZY}{2} + 1 \end{bmatrix} \begin{bmatrix} V_R \\ I_R \end{bmatrix} \quad (2.23)$$

2.2 Transformada de Fourier

A Transformada de Fourier descreve dados do domínio do tempo para o domínio da frequência, o que permite a detecção de variações na frequência do sinal, funcionando como uma ferramenta que trata o sinal, podendo ser utilizada como uma ferramenta de pré-processamento de dados (SOTERO *et al.*, 2012). Dada uma função periódica, representando um sinal ou um sistema, a mesma pode ser representada através de uma série de Fourier. Porém, quando tem-se uma função não periódica necessita-se da utilização da transformada de Fourier, (NILSSON; RIEDEL, 2009):

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{i\omega_n t} \quad (2.24)$$

onde:

$$C_n = \frac{1}{T} \int_0^T f(t) e^{-i\omega_n t} dt \quad (2.25)$$

ou:

$$C_n = \frac{1}{T} \int_{-T/2}^{T/2} f(t) e^{-i\omega_n t} dt \quad (2.26)$$

Ao tender o período T ao infinito, tem-se uma função não periódica. À medida que T aumenta, a distância entre as frequência harmônicas adjacentes diminui, ou seja, (NILSSON; RIEDEL, 2009):

$$\Delta\omega = (n+1)\omega_0 - n\omega_0 = \omega_0 = \frac{2\pi}{T} \quad (2.27)$$

Assim, tem-se que a diferença entre duas frequências consecutivas é ω_0 .

Observando a equação 2.26 pode-se observar que a medida que o período T cresce, os coeficientes de Fourier C_n diminuem, ou seja $C_n \rightarrow 0$ quando $C_n \rightarrow \infty$.

Ao mover o período T para a esquerda da equação 2.26, temos que $C_n T$ quando $T \rightarrow \infty$ é:

$$C_n T \rightarrow \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt \quad (2.28)$$

Assim, a integral obtida é a transformada de Fourier de $f(t)$ e representada por, (NILSSON; RIEDEL, 2009):

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt \quad (2.29)$$

E a transformada inversa de Fourier é:

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{j\omega t} d\omega \quad (2.30)$$

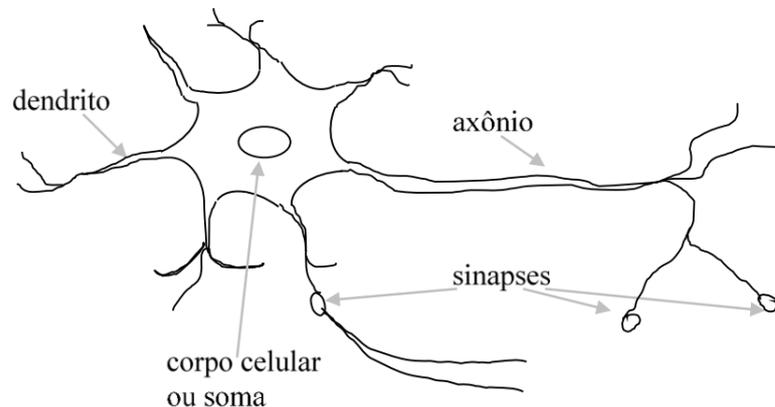
Com a possibilidade de se conduzir uma função do domínio do tempo para o domínio da frequência, é possível utilizá-la em processamento de sinais para obter o espectro de frequência do mesmo, trazendo a possibilidade de, no caso de faltas em linhas de transmissão, detectar as várias frequências geradas em uma falta podendo utilizar essa informação para a classificação dos tipos e do trecho onde as faltas ocorrem.

2.3 Redes Neurais Artificiais

De forma geral, uma rede neural artificial se baseia na capacidade de como o cérebro realizaria atividades não lineares. Assim como uma criança aprende observando o mundo a sua volta e tendo um desenvolvimento cerebral percebendo aos poucos padrões de comportamento, a rede neural também funciona desta forma, através da aprendizagem (HAYKIN, 2007).

A estrutura de uma rede neural artificial segue o funcionamento de uma rede de neurônios naturais que realizam entre si as sinapses. Um neurônio é composto de um axônio, onde é feita toda a transmissão de dados e os dendritos, que são as zonas receptivas de dados. Uma representação de um neurônio é exemplificada na Figura 7.

Figura 7 – Representação de um neurônio



Fonte: Mario, Filho e Abe (2021).

O processo de aprendizagem utilizado em uma rede neural artificial é feito através da atualização de parâmetros internos ao processo (pesos sinápticos) aos quais através de resultados advindos de iterações anteriores a esta, consegue realizar uma comparação com o resultado desejado e, assim verificar se a saída é aquela desejada. Caso contrário, é realizado um novo processo de aprendizagem. Assim, a cada iteração a rede neural vai aprendendo a identificar melhor os padrões e melhorando o seu desempenho.

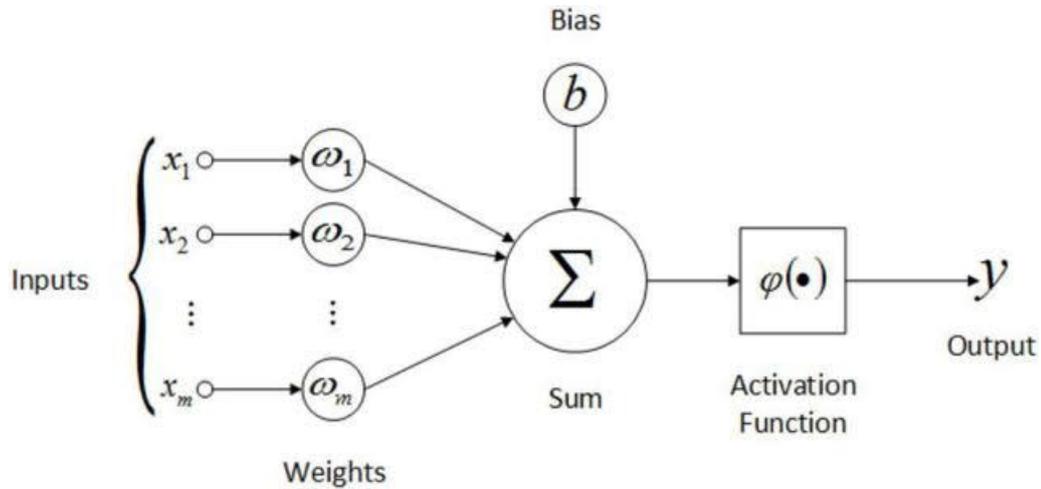
2.3.1 Redes neurais *perceptron* de camada única

É um dos primeiros modelos e a forma mais simples de uma rede que é utilizada para classificação de padrões linearmente separáveis. No entanto, os conceitos nela utilizados são base para arquiteturas de redes com topologias mais complexas.

Ela é composta por uma única camada de neurônios com pesos sinápticos ajustáveis, uma função aditiva, uma função de ativação, o *bias* e uma saída (HAYKIN, 2007). O *bias* é uma constante necessária para o ajuste da fronteira de decisão, de forma que ela não passe sempre pela origem, reduzindo significativamente o desempenho da rede (CECCON, 2020).

Contudo, esta rede é limitada a realizar classificação de padrões com apenas duas classes e, ainda, linearmente separáveis. Um exemplo de uma rede neural *perceptron* de camada única é apresentado na Figura 8.

Figura 8 – Rede neural Perceptron de camada única



Fonte: Raj (2020).

Ao apresentar o vetor $\mathbf{x} = [x_1, x_2, \dots, x_n]^T$ à entrada desta rede neural, é feita uma soma de produtos com o vetor de pesos sinápticos $\mathbf{w} = [w_1, w_2, \dots, w_m]^T$:

$$u = \sum_{j=1}^n x_j w_j \quad (2.31)$$

Que em sua forma matricial podemos expressar u como um produto interno dos vetores \mathbf{x} e \mathbf{w} :

$$\mathbf{u} = \mathbf{x}^T \mathbf{w} \quad (2.32)$$

Em seguida, o seu resultado é apresentado a uma função de ativação $\varphi(\times)$ que delimitará a amplitude da saída y do neurônio:

$$y = \varphi(u + b) \quad (2.33)$$

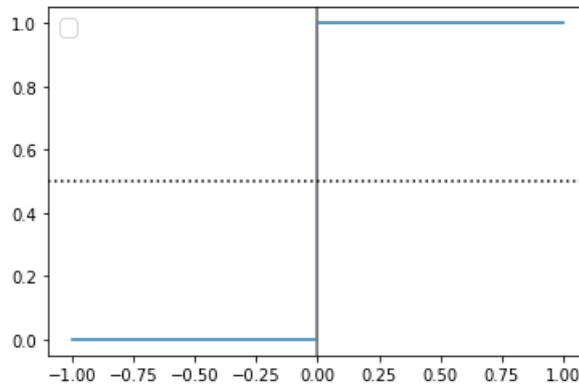
O *bias* b é adicionado a soma para impedir que a função de ativação transmita valores zerados, fazendo com que a rede fique “presa” em mínimos locais e não consiga aprender adequadamente, (FALCÃO *et al.*, 2019).

Existem diversas funções de ativação utilizadas nas redes neurais. Um exemplo de função de ativação é a degrau, a mais simples, utilizada para classificações binárias, tem a classificação conforme equação 2.34 (HAYKIN, 2007).

$$y = \begin{cases} 0, & \text{se } \varphi(u + b) < 0 \\ 1, & \text{se } \varphi(u + b) \geq 0 \end{cases} \quad (2.34)$$

Onde graficamente pode ser representada conforme Figura 9.

Figura 9 – Representação da função de ativação degrau.

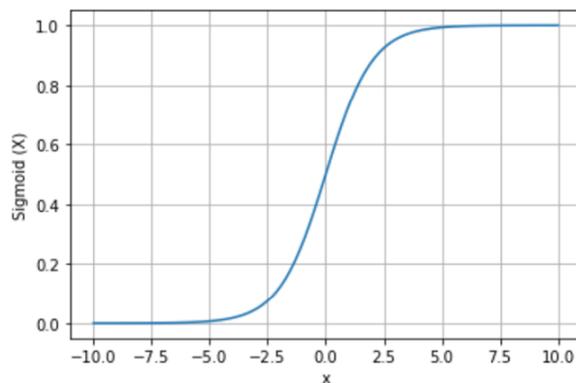


Fonte: Elaborado pelo autor (2022).

Outro exemplo de função de ativação é a função sigmoide. Esta função compreende de uma função que abrange valores de 0 a 1, se encontra na Equação 2.35 e graficamente representada na Figura 10 (HAYKIN, 2007).

$$\phi(z) = \frac{1}{1 + e^{-z}} \quad (2.35)$$

Figura 10 – Representação da função de ativação sigmoide.

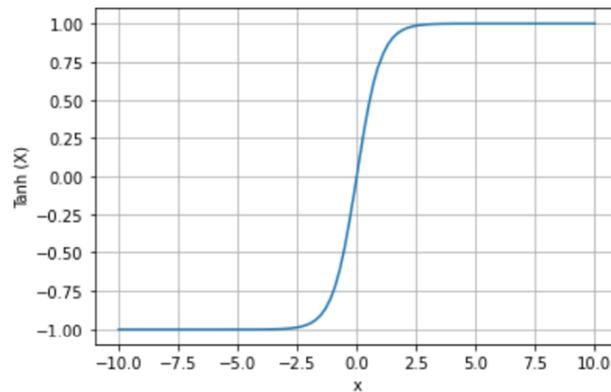


Fonte: Elaborado pelo autor (2022).

Também tem-se a função de ativação tangente hiperbólica, que é uma função semelhante a função sigmoide, mas é simétrica da origem. Os valores resultantes vão de -1 a 1. Em comparação com a função sigmoide, o gradiente da função tanh é mais acentuado. Esta função está explicita na equação 2.36 e representada graficamente na Figura 11 (HAYKIN, 2007).

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 = \tanh(x) \quad (2.36)$$

Figura 11 – Representação função de ativação tangente hiperbólica.



Fonte: Elaborado pelo autor (2022).

2.3.2 Redes neurais *perceptron* de múltiplas camadas

Uma rede neural perceptron de múltiplas camadas (*Multilayer perceptron*) possibilita a resolução de problemas de classificação que não são linearmente separáveis, como é o caso da classificação da saída de portas lógicas XOR (HAYKIN, 2007). A tabela-verdade desta função lógica é apresentada pela Tabela 1.

Tabela 1 – Combinações da porta lógica XOR

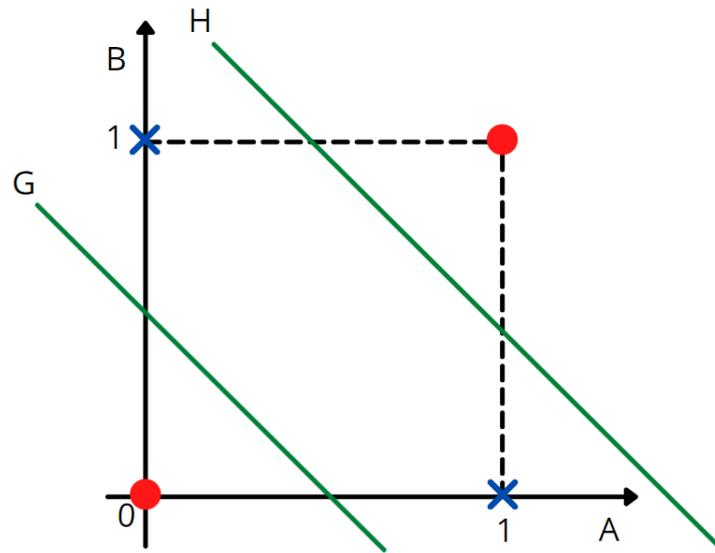
A	B	Saída
0	0	0
0	1	1
1	0	1
1	1	0

Fonte: Extraído de Haykin (2007).

Este exemplo é impossível de ser classificado com a rede perceptron de camada única pois a fronteira de separação é linear. Inserindo uma nova camada de neurônios (a camada oculta), será possível a inserção de uma componente não linear ao ajuste de uma fronteira de decisão criada pela rede neural (HAYKIN, 2007).

A Figura 12 apresenta na forma de plano cartesiano os dados da tabela-verdade da função XOR. Note que, com uma fronteira de separação linear não será possível discriminar os padrões de saída. Ainda na Figura 12, pode-se observar que apenas com uma das retas disponíveis “G” ou “H” não é possível realizar a classificação da saída da porta lógica. Apenas com um reta seja ela em qual posição estiver, sempre terá pelo menos um exemplar de cada entrada.

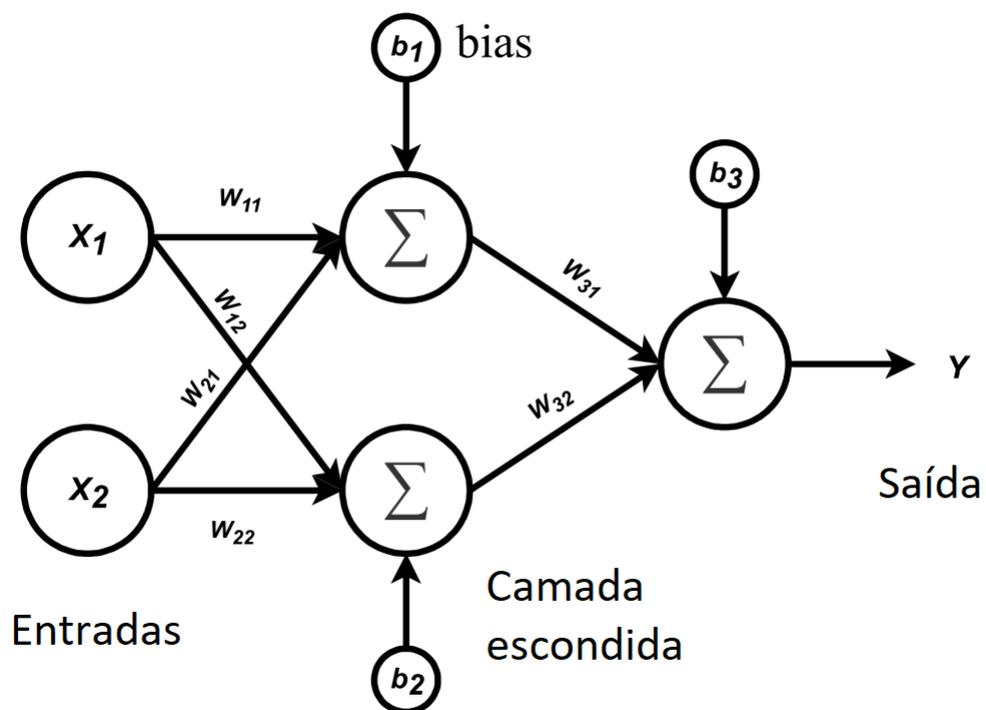
Figura 12 – Representação gráfica porta lógica XOR ilustrando as condições de classificação de saída da porta lógica XOR



Fonte: Elaborado pelo autor (2022).

Ao acrescentar uma única camada oculta com dois neurônios, será promovida uma transformação não linear no espaço de entrada, possibilitando a classificação correta do problema XOR, conforme a Figura 13.

Figura 13 – Representação Rede Neural *Multilayer Perceptron* para o problema XOR.



Fonte: Adaptado de Haykin (2007).

O funcionamento de uma rede neural de múltiplas camadas segue dois passos. O passo "Feedforward", onde as entradas são apresentadas aos neurônios da camada oculta que estão dispostos em uma formação amplamente conectada (*fully connected*). Em cada neurônio oculto é realizada uma soma de produtos entre a entrada da rede e o seu respectivo vetor de pesos sinápticos (w), passando depois por uma função de ativação para gerar a sua saída. Por fim, a saída de cada neurônio oculto é apresentada a um neurônio de saída que, processará as suas entradas da mesma forma que a rede *perceptron* de camada única (HAYKIN, 2007).

Assim o resultado de cada camada é propagado para a camada seguinte até que se chega a saída, (BENTO, 2021). Porém apenas com este passo, a rede não seria capaz de aprender, pois ela apenas converge para um lado e não retorna para verificar os resultados. Posto isto, o passo "Backward" é implementado utilizando um algoritmo denominado "Backpropagation" que permite o ajuste dos pesos sinápticos da rede neural de forma iterativa, minimizando a média do quadrado dos erros (CACERES, 2020).

Desta forma, o objetivo do "Backpropagation" é minimizar o erro de discriminação da rede neural a medida que os pesos são atualizados. O erro medido depende do valor da função de ativação onde no caso da rede neural *multilayer perceptron* deve-se utilizar uma função que não seja linear. Para explicação será utilizada a função sigmóide (ou sigmoideal) (CACERES, 2020).

A função de custo é uma métrica de desempenho da rede neural e ela é definida por emprego da Equação 2.37 (CACERES, 2020):

$$MSE = \frac{1}{2} \sum_k (\hat{y}_k - y_k)^2, \quad (2.37)$$

onde:

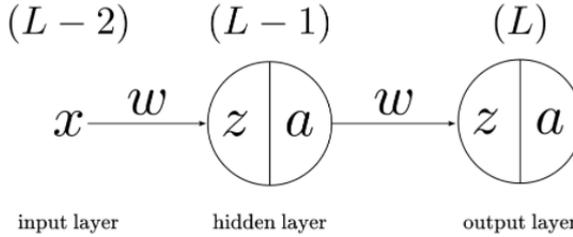
- MSE : é a soma dos erros quadráticos para todo o conjunto de dados (*Mean of Squared Error*);
- k : é o k -ésimo vetor de entrada apresentado a rede neural;
- \hat{y}_k : é o valor previsto pela rede;
- y_k : é o valor esperado (desejado) a ser dado pela rede.

A minimização do MSE é dada pela seguinte expressão, (CACERES, 2020):

$$\frac{\partial E}{\partial w^{(L)}} = \frac{\partial E}{\partial \hat{y}^{(L)}} \times \frac{\partial \hat{y}^{(L)}}{\partial z^{(L)}} \times \frac{\partial z^{(L)}}{\partial w^{(L)}} \quad (2.38)$$

Para fins de melhor entendimento das equações, os termos sobrescritos (L) , $(L - 1)$ e $(L - 2)$, representam as 3 partes de uma rede neural: camada de entrada, camada oculta e camada de saída, respectivamente. Tais termos estão dispostos conforme apresentado na Figura 14.

Figura 14 – Camadas básica de uma rede *Multilayer Perceptron*



Fonte: Caceres (2020).

Diferenciando cada termo da Equação 2.38, vem:

$$\frac{\partial E}{\partial \hat{y}^{(L)}} = \hat{y}^{(L)} - y \quad (2.39)$$

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \hat{y}^{(L)} (1 - \hat{y}^{(L)}) \quad (2.40)$$

$$\frac{\partial z^{(L)}}{\partial w^{(L)}} = \hat{y}^{(L-1)} \quad (2.41)$$

Substituindo 2.39, 2.40 e 2.41 em 2.38:

$$\frac{\partial E}{\partial w^{(L)}} = \hat{y}^{(L)} - y \times \hat{y}^{(L)} (1 - \hat{y}^{(L)}) \times \hat{y}^{(L-1)} \quad (2.42)$$

Assim na equação 2.42 temos o comportamento de como o erro altera conforme o peso que conecta a camada oculta e a camada de saída é alterado. Para saber como é esta mesma relação porém entre a camada de entrada e a camada oculta é bem simples, bastando apenas aplicar a regra da cadeia novamente para os dados relacionados a esta camada conforme equação 2.43 (CACERES, 2020).

$$\frac{\partial E}{\partial w^{(L-1)}} = \frac{\partial E}{\partial \hat{y}^{(L)}} \times \frac{\partial \hat{y}^{(L)}}{\partial z^{(L)}} \times \frac{\partial z^{(L)}}{\partial \hat{y}^{(L-1)}} \times \frac{\partial \hat{y}^{(L-1)}}{\partial z^{(L-1)}} \times \frac{\partial z^{(L-1)}}{\partial w^{(L-1)}} \quad (2.43)$$

Seguindo a mesma metodologia para a passagem da equação 2.38 para a equação 2.42 tem-se um resultado onde:

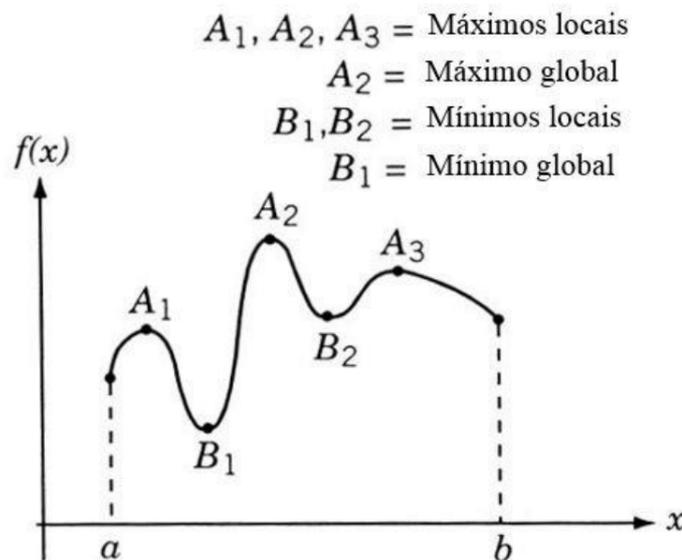
$$\frac{\partial E}{\partial w^{(L-1)}} = \hat{y}^{(L)} - y \times \hat{y}^{(L)} (1 - \hat{y}^{(L)}) \times w^{(L)} \times \hat{y}^{(L-1)} (1 - \hat{y}^{(L-1)}) \times x^{(L-1)} \quad (2.44)$$

Esta equação é utilizada para redes com apenas um neurônio, porém se quisermos trabalhar com mais de um neurônio na camada escondida, iremos implementar índices sub escritos extras.

$$\frac{\partial E}{\partial w_{ki}^{(L-1)}} = \hat{y}_j^{(L)} - y \times \hat{y}_j^{(L)} (1 - \hat{y}_j^{(L)}) \times w_{jk}^{(L)} \times \hat{y}_k^{(L-1)} (1 - \hat{y}_k^{(L-1)}) \times x_i^{(L-1)} \quad (2.45)$$

Os índices j , i e k representam o índice do neurônio da camada de saída, entrada e oculta respectivamente. O algoritmo de retropropagação muitas vezes através deste gradiente, irá encontrar mínimos locais que não seriam o menor erro possível dado a rede completa. Isso resume-se na Figura 15.

Figura 15 – Representação da busca de mínimos globais



Fonte: Marsilio (2015).

Assim é necessário realizar um correto ajuste do passo de aprendizagem aqui denominado η . Um passo de aprendizagem muito grande irá agilizar o aprendizado da rede neural, porém pode deixar de detectar um mínimo global. Um passo de aprendizagem pequeno terá uma maior probabilidade de encontrar o mínimo global, porém com um tempo de processamento maior, (CACERES, 2020).

Conforme equação 2.45 tem-se o gradiente calculado. Agora, utiliza-se da equação 2.46 para realizar a atualização dos pesos sinápticos. É por meio deste procedimento que ocorre a aprendizagem da rede.

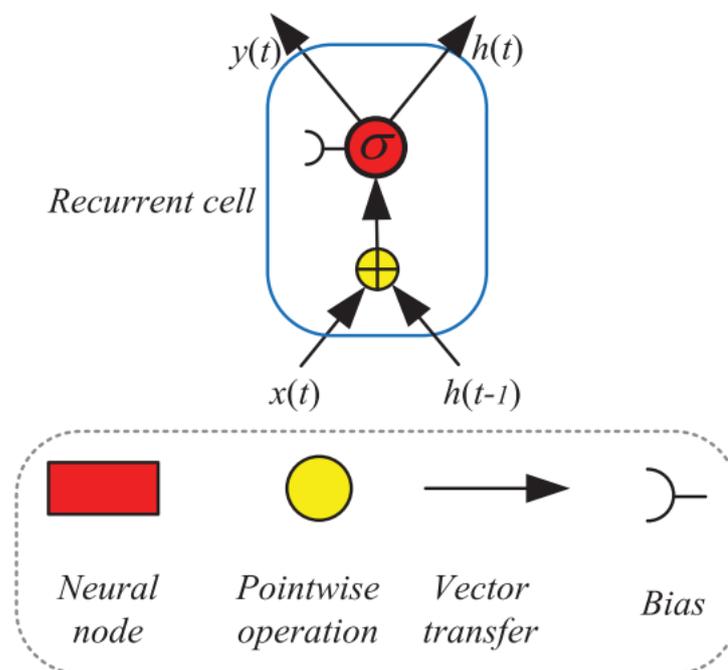
$$w_{ki}^{L-1} = w_{ki}^{L-1} - \eta \times \frac{\partial E}{\partial w_{ki}^{L-1}} \quad (2.46)$$

2.3.3 Redes neurais recorrentes

As redes neurais perceptron de multicamadas são capazes de resolver muitos problemas de classificação. Porém, a resposta destas redes neurais levam em consideração o conjunto de entradas apresentados somente no instante de tempo t , sem levar em consideração, portanto, as saídas obtidas em instantes de tempo anteriores. Após o processamento de cada conjunto de dados de entrada, todo o estado da rede é perdido. Isto em uma rede que visa por exemplo classificar tipos de plantas não é um problema, pois os exemplos são pontuais, como tamanho da pétala, cor, etc. Porém, no caso do presente trabalho, temos uma análise de um dado em uma sequência temporal, assim é preciso fazer um processamento de uma janela de dados que estão relacionados entre si, não apenas de um ponto em específico. Para tal classe de problemas pode-se utilizar as Redes Neurais Recorrentes (RNN) que são modelos com a capacidade de transmitir informações seletivamente pelas camadas da rede, enquanto processa os dados sequenciais, (LIPTON; BERKOWITZ; ELKAN, 2015).

Estas redes funcionam como redes neurais de multicamadas porém acrescido de uma recorrência em seus neurônios. A Figura 16 exemplifica como funciona uma célula sigma recorrente padrão.

Figura 16 – Célula-exemplo Rede Neural Recorrente



Fonte: Extraída de Yu *et al.* (2019).

A expressão matemática que representa a imagem se encontra nas equações 2.47 e 2.48,

(YU *et al.*, 2019).

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b) \quad (2.47)$$

e

$$y_t = h_t, \quad (2.48)$$

onde:

- $\sigma(\cdot)$ é a função de ativação sigmóide;
- x_t é a entrada;
- h_{t-1} é a informação recorrente;
- y_t é a saída da célula no tempo t ;
- W_h e W_x são os pesos;
- b é o *bias*.

Na equação 2.47 o estado interno da célula h_t depende tanto da entrada x_t , quanto do estado da célula anterior onde na Figura 16 podemos visualizar como o h_{t-1} . Assim é feito a multiplicação de cada um desses pelos seus respectivos pesos. Por fim essa expressão é condensada pela função σ , que pode ser uma função sigmoide logística ou tanh, para que os valores se tornem viáveis para os gradientes na retropropagação.

Para que as redes neurais recorrentes possam aprender, é necessário utilizar um algoritmo aprimorado do *backpropagation*, cujo nome é *Backpropagation Through Time*.

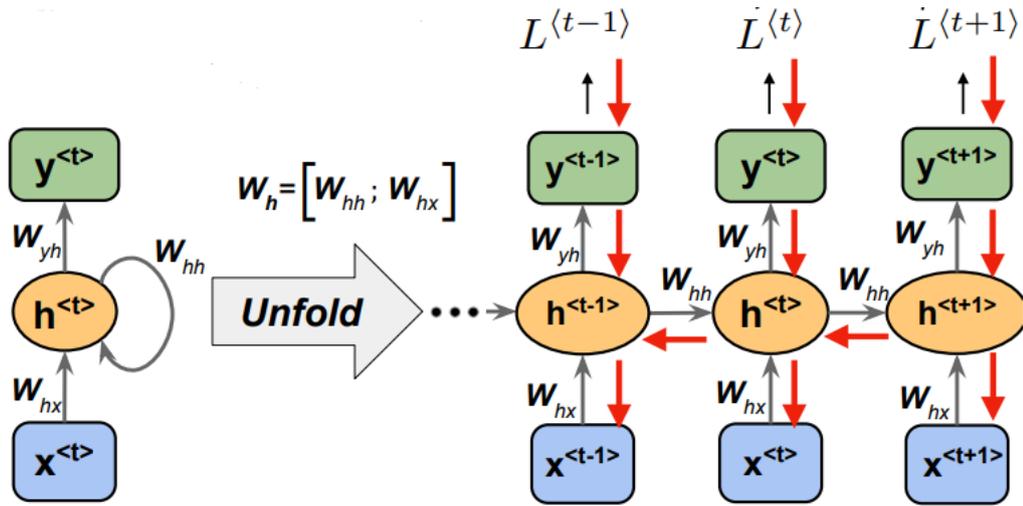
Na Figura 17 podemos observar que temos perdas L em cada etapa do ciclo da célula recorrente, e neste algoritmo necessitamos realizar a soma das mesmas para calcularmos a perda geral.

O gradiente da perda no tempo em relação a matriz de pesos ocultos W_{hh} é dado por:

$$\frac{\partial L^{(t)}}{\partial \mathbf{W}_{hh}} = \frac{\partial L^{(t)}}{\partial y^{(t)}} \times \frac{\partial y^{(t)}}{\partial \mathbf{h}^{(t)}} \times \left(\sum_{k=1}^t \frac{\partial \mathbf{h}^{(t)}}{\partial \mathbf{h}^{(k)}} \times \frac{\partial \mathbf{h}^{(k)}}{\partial \mathbf{W}_{hh}} \right) \quad (2.49)$$

Os dois primeiros termos da equação 2.49 já utilizamos no *backpropagation* convencional, porém agora além disto temos o terceiro termo em um somatório que pode nos causar um

Figura 17 – Célula-exemplo Rede Neural Recorrente



Fonte: Raschka (2021).

problema, pois aqui surgimos com um produtório, pois o termo $\frac{\partial \mathbf{h}^{(t)}}{\partial \mathbf{h}^{(k)}}$ engloba todos os passos de uma célula. Assim temos que:

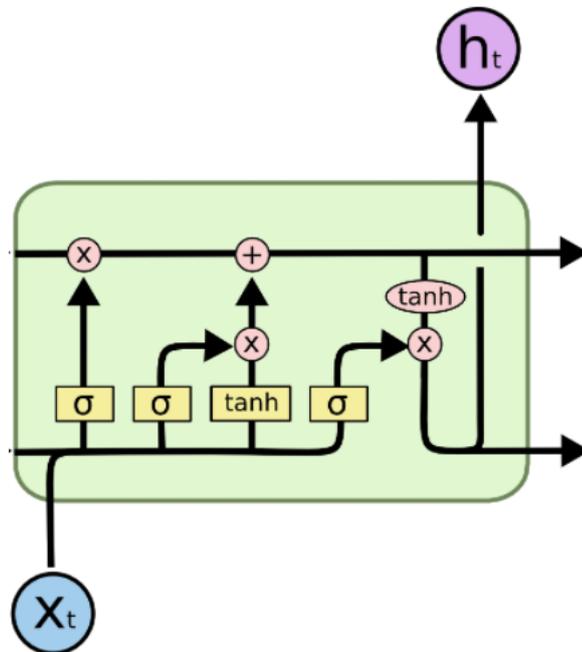
$$\frac{\partial \mathbf{h}^{(t)}}{\partial \mathbf{h}^{(k)}} = \prod_{i=k+1}^t \frac{\partial \mathbf{h}^{(i)}}{\partial \mathbf{h}^{(i-1)}} \quad (2.50)$$

Isto pode nos causar o problema de desaparecimento ou explosão do gradiente. Conforme o algoritmo de retropropagação avança em direção à camada de entrada, os gradientes geralmente ficam cada vez menores e se aproximam de zero, o que eventualmente deixam os pesos das camadas iniciais ou inferiores quase inalterados. Como resultado, a descida do gradiente nunca converge para um valor ótimo. Isto é conhecido como o problema dos gradientes de desaparecimento. Ao contrário, em alguns casos, os gradientes ficam cada vez maiores à medida que o algoritmo de retropropagação progride (PYKES, 2020). Isto, causa atualizações de peso muito grandes e faz com que a descida do gradiente não consiga convergir. Isso é conhecido como problema de gradientes explosivos, com estes problemas, a rede não consegue guardar dados de longo prazo. Para resolver isto, utilizamos o LSTM (*Long-Short Term Memory*) (PYKES, 2020).

2.3.4 LSTM - *Long-Short Term Memory*

A rede neural LSTM (*Long Short-Term Memory*) foi criada como solução para a memória de curto prazo, pois através de seus mecanismos, esta célula consegue filtrar as informações que são realmente relevantes, ou seja, o LSTM escolhe quais informações ela deseja manter ou descartar. Esta célula tem um fluxo de controle semelhante a uma rede neural recorrente pois processa os dados a medida que se propagam (OLAH, 2015). A diferença está nas operações internas à célula. Um exemplo de célula LSTM é apresentado na Figura 18.

Figura 18 – Célula exemplo LSTM



Fonte: Olah (2015).

O LSTM contém vários "portões" conforme pode-se observar na Figura 18 que contém uma função sigmóide, onde observa-se uma faixa de valores entre 0 e 1. Assim nessa simples função já é possível determinar valores importantes, onde valores multiplicados por 0 são eliminados e valores próximos de 1 são considerados importantes para a rede.

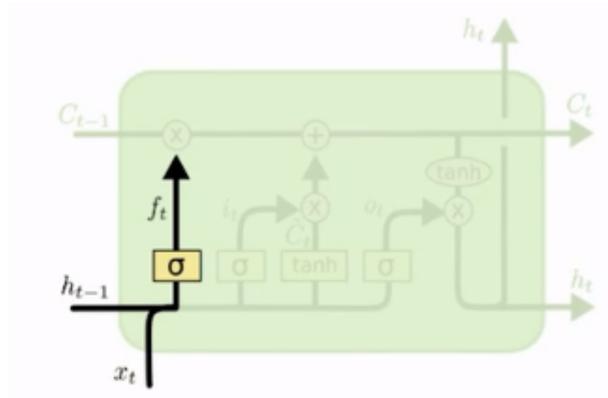
No início de uma célula LSTM, tem-se a primeira função que nomeia-se como célula de descarte exemplificada na Figura 19, onde as informações do estado anterior e as informações da entrada atual são transmitidas pela função sigmóide, seguindo a dinâmica apresentada no parágrafo anterior, onde resultados após a função sigmóide próximos de 0 são descartados e próximo de 1 são mantidos. Isso se resume na Equação 2.51:

$$f_t = \sigma(W_f \times [h_{t-1}, x_t] + b_f), \quad (2.51)$$

onde:

- $\sigma(\cdot)$ é a função de ativação sigmóide;
- W_f é o peso;
- h_{t-1} e x_t são as entradas;
- b_f é o bias.

Figura 19 – Primeira parte da célula LSTM



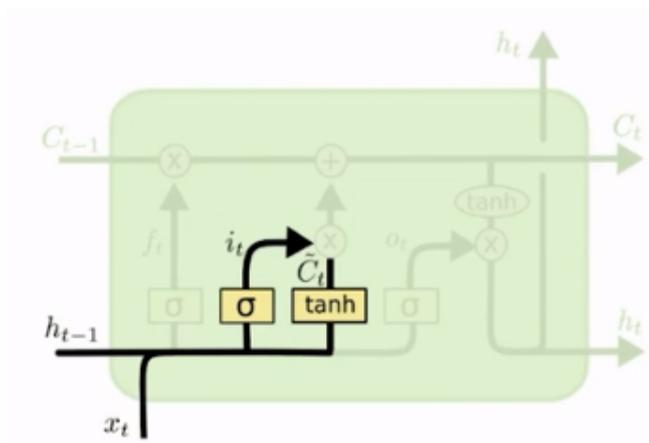
Fonte: Olah (2015).

Para atualizar o estado da célula o próximo passo é feita a multiplicação de uma função sigmóide aplicada aos estados de entrada e de uma função tanh também aplicada aos estados de entrada da célula. Isto está exemplificado na Figura 20. Isto resume-se na equação 2.52 e 2.53 (OLAH, 2015).

$$i_t = \sigma(W_i \times [h_{t-1}, x_t] + b_i) \quad (2.52)$$

$$\tilde{C}_t = \tanh(W_C \times [h_{t-1}, x_t] + b_C) \quad (2.53)$$

Figura 20 – Segunda parte da célula LSTM



Fonte: Olah (2015).

Agora, com informações suficientes é feito a multiplicação das informações obtidas na Figura 19 com o estado anterior da célula. Novamente caso a multiplicação tenha resultado

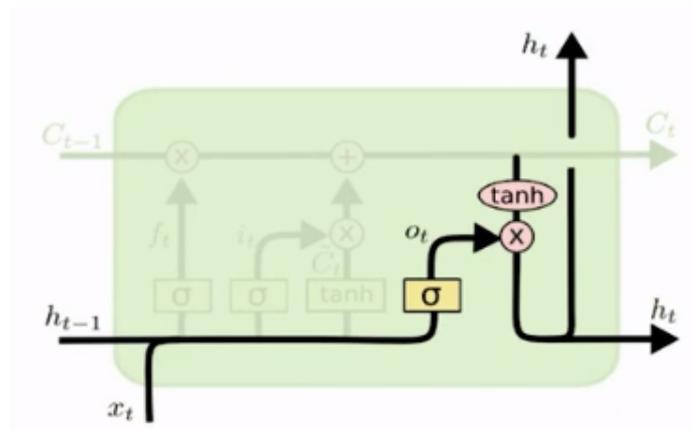
nulo, o dado é descartado. Após, essa informação é somada ao resultado da Figura 20 e, assim, resulta-se em um novo estado da célula.

Por fim tem-se a saída da célula conforme Figura 21, onde é decidido qual deve ser o próximo estado oculto, ou seja, quais informações serão repassadas para a próxima etapa. Como pode-se observar, a saída tanh é multiplicada pela saída sigmóide, decidindo assim quais informações o estado oculto deve transportar. Finalizando assim na equação 2.54 e 2.55 (OLAH, 2015).

$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o) \quad (2.54)$$

$$h_t = o_t \times \tanh(C_t) \quad (2.55)$$

Figura 21 – Saída célula LSTM



Fonte: Olah (2015).

3 MATERIAIS E MÉTODOS

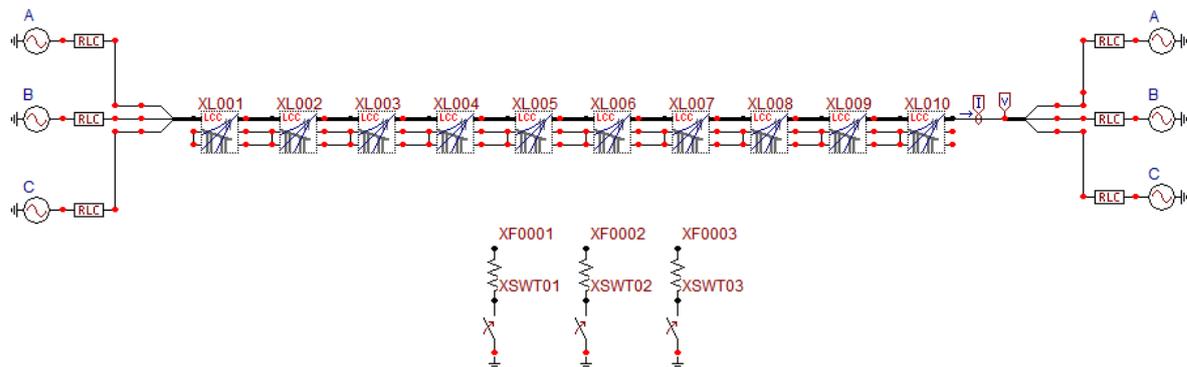
3.1 Simulação

Devido a dificuldade na realização de medições em campo, utilizou-se o *software* ATP Versão 1.11 para simular várias situações de ocorrência de falta. De acordo com Ye *et al.*, (2012) o ATP é um dos *softwares* mais utilizados para simulação digital de transitórios eletromagnéticos e eletromecânicos em sistemas de energia elétrica.

Com o intuito de simular as faltas Fase-terra, Bifásica-terra e Trifásica-terra, foram criados inicialmente 10 vãos de linhas de 10 km cada e também 3 chaves com uma resistência de falta de 0.5Ω , (VIEIRA, 2016). O mesmo pode parecer uma situação anormal devido as distâncias de entre vãos reais não serem tão longas como considerado neste estudo computacional. Porém no algoritmo de execução essas mesmas seções são subdivididas em seções de 1 km.

A topologia da linha de transmissão está ilustrada pela Figura 22, onde XL001 à XL010 são os nomes dos nós onde estarão ocorrendo cada falta simulada.

Figura 22 – Topologia para a simulação das faltas.



Fonte: Elaborado pelo autor (2022).

A linha de transmissão contém uma fonte equilibrada com os parâmetros da Tabela 2, retirados de (VIEIRA, 2016):

Tabela 2 – Parâmetros da Fonte equilibrada.

Fase	Tensão (kV)	Impedância (Ω , para 60Hz)
Fonte A	$241.5 / 0^\circ$	$0.050 + j11.309$
Fonte B	$241.5 / 120^\circ$	$0.050 + j11.309$
Fonte C	$241.5 / 240^\circ$	$0.050 + j11.309$

Fonte: Elaborado pelo autor (2022).

E uma outra fonte com os parâmetros da Tabela 3, retirados de (VIEIRA, 2016):

Tabela 3 – Parâmetros da Fonte à direita da LT.

Fase	Tensão (kV)	Impedância (Ω , para 60Hz)
Fonte A	225.5 / 0°	6.000 + j37.700
Fonte B	225.9 / 122°	5.000 + j45.239
Fonte C	225.2 / 241°	5.000 + j52.779

Fonte: Elaborado pelo autor (2022).

No ATP, foi utilizada a rotina LCC (*Line/Cable Constants*) onde são configurados parâmetros de resistividade do solo, comprimento do vão, número de fases, e também parâmetros do condutor e distância dos mesmo entre si, à torre e à terra (REIS; JR; CAIXETA, 2012). O modelo de linhas de transmissão utilizados é o Jmarti que utiliza parâmetros distribuídos e é dependente da frequência, sendo um dos modelos mais utilizados para simular linhas longas (ORAMUS; FLORKOWSKI, 2014) (MARTI, 1982).

É selecionado os valores de cada campo na interface do LCC, para melhor visualização e explanação, será dividido em Figuras esta interface. Primeiramente, é necessário selecionar os dados comuns utilizados em qualquer tipo de linha e cabo. Conforme Figura 23, temos os dados de:

- Rho [ohm*m]: Refere-se a resistividade do solo em ohms para uma terra homogênea, (REIS; JR; CAIXETA, 2012);
- Freq. init [Hz]: Refere-se a frequência em que os parâmetros da linha serão calculados (Bergeron e PI) ou a menor frequência (JMarti, Noda e Semlyem) para cálculo dos parâmetros, (REIS; JR; CAIXETA, 2012);
- Length [km]: Comprimento das linhas aéreas, em km.

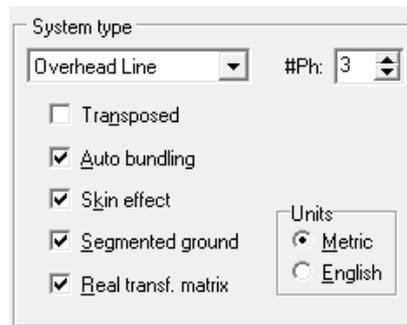
Figura 23 – Dados comuns para configuração dos tipos de linha.

Fonte: Elaborado pelo autor (2022).

Na interface é necessário a configuração do modelo e dados de linhas aéreas. Conforme Figura 24, a configuração necessária é dos seguintes campos:

- Transposed: Informa se a linha é transposta ou não;

Figura 24 – Configuração de modelo e dados de linhas aéreas.



Fonte: Elaborado pelo autor (2022).

- **Auto bundling:** Se ativado habilita o recurso de agrupamento automático, (REIS; JR; CAIXETA, 2012);
- **Skin effect:** Se setado, o efeito pelicular será considerado, (REIS; JR; CAIXETA, 2012);
- **Metric/English:** Alternar entre os sistemas de unidade;
- **Segmented ground:** Se o botão estiver desmarcado, os cabos de aterramento serão assumidos como continuamente aterrados, (REIS; JR; CAIXETA, 2012);
- **Real trans. matrix:** Se marcado a matriz de transformação é assumida como real. Recomendado para simulações em regime transitório. Caso contrário, uma matriz de transformação completa complexa será usada;
- **#Ph:** Define a quantidade de fases do circuito.

Como é utilizado na simulação o modelo JMarti, na Figura 25 temos a sua configuração onde:

- **Decades e Ponts/Dec:** Faixa de frequência que se inicia a partir da frequência inicial em Standard data até um limite de frequência máxima aqui especificada, (REIS; JR; CAIXETA, 2012);
- **Freq. Matrix [Hz]:** Frequência onde a matrix de transformação é calculada, (REIS; JR; CAIXETA, 2012);
- **Freq. SS [Hz]:** Frequência para o cálculo da condição de regime permanente.
- **Use default fitting:** Setar caso o ajuste do modelo for default.

Por último é necessário a configuração da página de dados das linhas, conforme Figura 26, onde:

Figura 25 – Configuração do tipo de modelo.

The image shows a software dialog box titled "Model". It is divided into two main sections: "Type" and "Data".

Type: This section contains five radio button options: "Bergeron", "PI", "JMartí" (which is selected), "Semlyen", and "Noda".

Data: This section contains several input fields and a checkbox:

- "Decades": A text box containing the number "8".
- "Points/Dec": A text box containing the number "5".
- "Freq. matrix [Hz]": A text box containing the number "5000".
- "Freq. SS [Hz]": A text box containing the number "60".
- "Use default fitting": A checked checkbox.

Fonte: Elaborado pelo autor (2022).

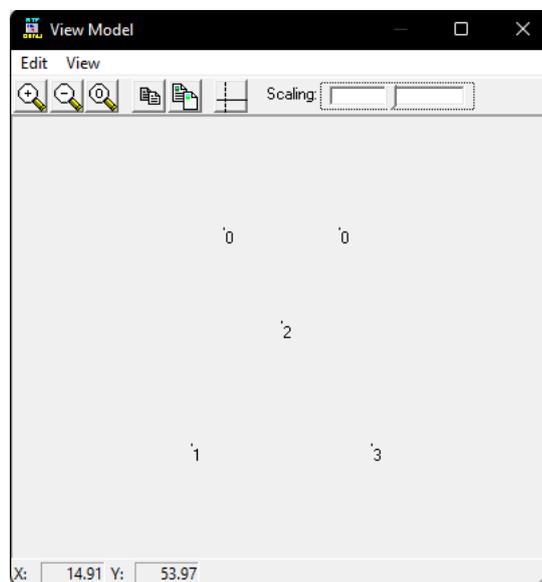
- Ph.no.: Número da fase. 0 para cabos de guarda;
- Rin: Raio interno do condutor. Considerado apenas se o botão de efeito pelicular estiver sendo considerado;
- Rout: Raio externo do condutor;
- Resis: Resistência DC do condutor(ohm/unidade de comprimento);
- Horiz: Distância horizontal a partir do centro do feixe de uma linha de referência;
- Vtower: Altura vertical do feixe na torre;
- Vmid: Altura vertical do meio do vão;
- Separ: Distância entre os condutores no feixe de cada fase;
- Alpha: Posição angular de um dos condutores em um feixe;
- NB: Número de condutores em um feixe.

Figura 26 – Disposição física dos cabos.

#	Ph.no.	Rin [cm]	Rout [cm]	Rsis [ohm/km DC]	Horiz [m]	Vtower [m]	Vmid [m]	Separ [cm]	Alpha [deg]	NB
1	1	0.2178	0.801	0.05215	-20	50	50	18	0	1
2	2	0.2178	0.801	0.05215	0	77.5	77.5	18	0	1
3	3	0.2178	0.801	0.05215	20	50	50	18	0	1
4	0	0	0.193	2.61	-12.9	98.5	98.5	0	0	0
5	0	0	0.193	2.61	12.9	98.5	98.5	0	0	0

Fonte: Elaborado pelo autor (2022).

Figura 27 – Disposição física do modelo.



Fonte: Elaborado pelo autor (2022).

De forma visual, a disponibilização dos cabos se mostra conforme Figura 27.

Como é possível observar na Figura 22, a obtenção da corrente e tensão trifásica acontece em uma extremidade da linha devido ao fato que, a localização da falta depende de um ponto de referência para sua localização (REDDY; MOHANTA, 2008).

3.2 Aquisição dos dados

Uma das características mais relevantes envolvidas na aplicação da rede neural, é a sua adaptação por experiência, ou seja, seus parâmetros internos. Normalmente seus pesos sinápticos, são ajustados a partir de apresentação sucessiva de exemplos que podem ser amostras, padrões e medidas relacionados ao comportamento do processo. Assim possibilitando a aquisição do conhecimento por meio da experimentação (SILVA; SPATTI; FLAUZINO, 2010). Além disto, Silva et al. (2010) evidenciam também que a rede neural possui a capacidade de aprendizado. Através de um treinamento a rede consegue extrair a relação existente entre as diversas variáveis que constitui a aplicação.

Para a aquisição dos dados para a rede, utilizou-se uma biblioteca em Python disponibilizada por Vieira (2016). Essa biblioteca contém um conjunto de funções que conseguem executar o ATP externamente e assim gerar a base de dados. O algoritmo altera os parâmetros no ATP automaticamente e desta forma é possível simular cada falta para cada quilômetro testado, permitindo assim a criação de uma base de dados satisfatória para se treinar a rede neural.

Como o ATP é um arquivo executado através de um código, utilizando Python criou-se um algoritmo que manipula cada parte da simulação do ATP. Assim é possível simular a falta em várias situações distintas. Alterando a linha de código do ATP é possível fechar uma chave com a terra conforme mostra a Figura 22 e assim simular a falta. A Figura 28 exemplifica a parte da linha de código no ATP que refere-se ao tempo de fechamento das chaves, e assim a falta é simulada em cada quilômetro iterativamente.

Figura 28 – Exemplo de linha de código do ATP com falta.

```

; XL006C, XL006A
$INCLUDE, C:\ATP\project\lcc\littcc_L10.lib, XL006B, XL006C, XL006A, XL007B $$
, XL007C, XL007A
$INCLUDE, C:\ATP\project\lcc\littcc_L10.lib, XL007A, XL007C, XL007B, XL008A $$
, XL008C, XL008B
$INCLUDE, C:\ATP\project\lcc\littcc_L10.lib, XL008A, XL008C, XL008B, XL009A $$
, XL009C, XL009B
$INCLUDE, C:\ATP\project\lcc\littcc_L10.lib, XL009A, XL009B, XL009C, XL010A $$
, XL010B, XL010C
$INCLUDE, C:\ATP\project\lcc\littcc_L10.lib, XL010A, XL010B, XL010C, XL011A $$
, XL011B, XL011C
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde >< Ie ><Vf/CLOP >< type >
XL001BX0001B MEASURING 1
XL001CX0001C MEASURING 1
XL001AX0001A MEASURING 1
XSWT01XF0001 .05 -1. 0
XSWT03XF0003 .05 -1. 0
XSWT02XF0002 .05 -1. 0
/SOURCE

```

Fonte: Elaborado pelo autor (2022).

3.3 Base de Dados e Rede Neural

Foi necessário a criação de duas redes neurais separadas devido a necessidade além classificar o tipo de falta, também localizar em qual distância a mesma ocorreu. Para a rede neural localizadora tem-se a localização da falta entre 1 a 100 quilômetros com passos de 1 km. Já para a rede neural classificadora temos as seguintes saídas:

- Falta ABCT
- Falta AT
- Falta BT
- Falta CT
- Falta ABT
- Falta ACT
- Falta BCT

Foram utilizadas nas duas redes criadas a rede neural recorrente com LSTM (*Long-Short Term Memory*) devido à propriedade de armazenar memória do processo a longo prazo. Assim consegue-se fazer uma análise de dados ao longo do tempo, como é o caso de corrente e tensão. As redes neurais criadas utilizaram a biblioteca Keras como base, uma biblioteca de *software* de código aberto que fornece uma interface Python para redes neurais artificiais, utilizando-se em seu *backend* o TensorFlow, outra biblioteca de código aberto, (ABADI *et al.*, 2015), (CHOLLET *et al.*, 2015).

Em cada rede neural foram utilizados células LSTM, Dropout e neurônios densos de saída. As células LSTM foram utilizadas a partir da entrada da rede até uma camada antes da saída. São nestas células onde ocorrem toda a inteligência da rede que através de gradientes e cálculos matemáticos expostos na seção 2.3.4 conseguem realizar a classificação do problema.

O *Dropout* utilizado auxilia para que a rede não fique superajustada aos parâmetros ao qual ela foi treinada, já que este algoritmo elimina no treinamento, temporariamente, algumas células ocultas aleatoriamente durante o treinamento, para que assim a rede tenha uma generalidade maior na classificação, (ACADEMY, 2018).

Os neurônios densos de saída, conforme (CHOLLET *et al.*, 2015), constituem basicamente a camada de saída densamente conectada, igualmente a qualquer rede neural, aplicando uma função de ativação no resultado da rede e assim obtendo a saída.

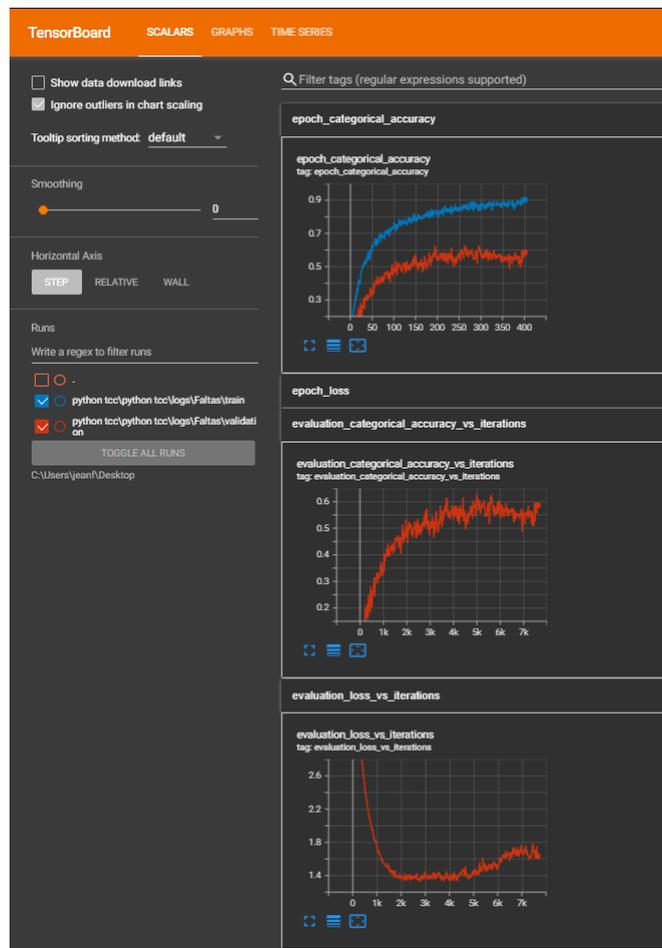
Para se obter o melhor desempenho de cada rede neural, utilizou-se da biblioteca Keras Tuner que contém uma estrutura de otimização de hiperparâmetros (parâmetros ajustáveis dentro

de uma rede que permite seu ajuste para um melhor desempenho da rede) para auxiliar na pesquisa destes com o objetivo de conseguir encontrar a melhor estrutura possível para a rede neural, (CHOLLET *et al.*, 2015).

Para a utilização destas ferramentas, primeiramente é criada uma função contendo o modelo criado da rede neural, e assim adiciona-se o espaço de busca para se encontrar os melhores hiperparâmetros possíveis. Assim inicializa-se outra função que encontrará o melhor modelo de rede, através de iterações combinando cada hiperparâmetro escolhido.

Para a visualização dos resultados, utilizou-se da ferramenta TensorBoard, ela monitora as métricas, perdas e precisão durante o treinamento, podendo obter informações detalhadas de cada época no treinamento da rede neural, (MANÉ *et al.*, 2015). Como exemplo segue a interface do TensorBoard na Figura 29.

Figura 29 – Interface TensorBoard



Fonte: Elaborado pelo autor (2022).

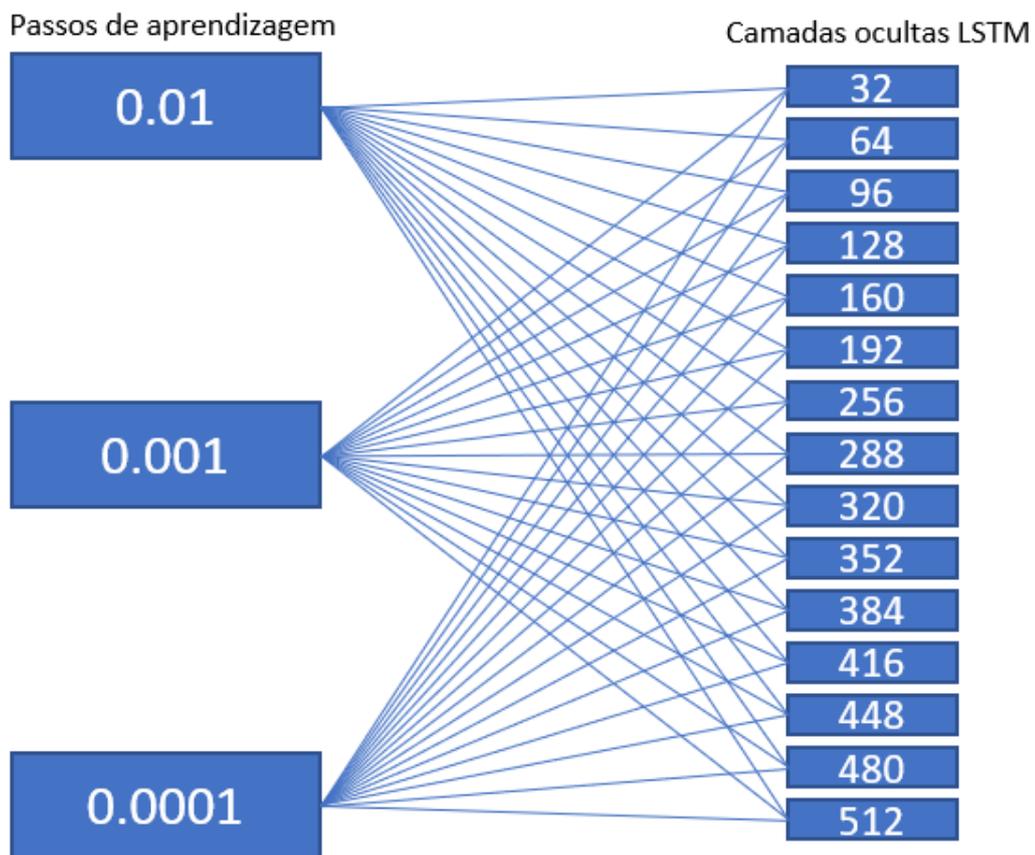
No caso específico das redes neurais utilizadas neste trabalho, foram pesquisados a quantidade de neurônios ocultos da camada LSTM e os passos de aprendizagem conforme segue abaixo:

- Unidades na camada oculta da camada LSTM: 32 a 512 com um passo de 32.
- Passos de aprendizagem: 0,01, 0,001 e 0,0001.

Para a otimização da busca pelo mínimo global foi utilizada a otimização de Adam, consistindo em um método de gradiente descendente estocástico computacionalmente eficiente, com pouca necessidade de memória e adequado para problemas que contêm muitos dados, (CHOLLET *et al.*, 2015).

As redes neurais são executadas 48 vezes variando seus hiperparâmetros conforme configurações representadas na Figura 30:

Figura 30 – Valores que compõem os setups a partir de cenários de variação dos hiperparâmetros para as redes neurais.



Fonte: Elaborado pelo autor (2022).

4 RESULTADOS E DISCUSSÕES

A Figura 31 mostra o conteúdo inicial de um arquivo ATP, que é um *software* que utiliza uma linguagem de programação que consegue aplicar métodos numéricos para resolução matemática consistente e com resultados precisos para transitórios eletromagnéticos em redes elétricas. Devido a dificuldade de se traduzir em textos a configuração de um sistema elétrico, geralmente é utilizado o programa ATPDraw que fornece uma interface gráfica onde é possível criar circuitos e sistemas complexos e traduzi-los em linguagem de programação reconhecida pelo ATP. Na Figura 22 contém um exemplo gráfico de um circuito no ATPDraw.

Figura 31 – Trecho de um cartão do ATP.

```

FFTA1 - Bloco de notas
Arquivo  Editar  Exibir

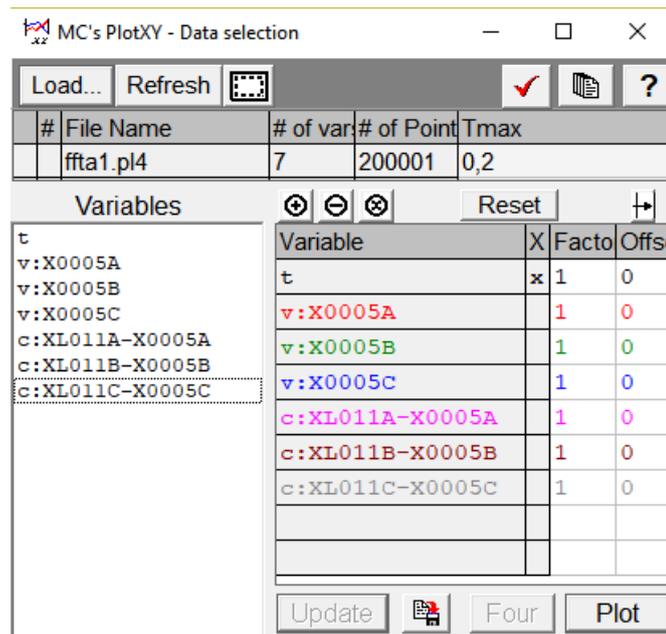
BEGIN NEW DATA CASE
C -----
C Generated by ATPDRAW  outubro, sexta-feira 7, 2016
C A Bonneville Power Administration program
C by H. K. Høidalen at SEFAS/NTNU - NORWAY 1994-2015
C -----
$DUMMY, XYZ000
C dT >< Tmax >< Xopt >< Copt ><Epsiln>
  1.E-6      .1
  1      1      1      1      0      0      1      0
C      1      2      3      4      5      6      7      8
C 3456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n1 >< n2 ><ref1><ref2>< R >< L >< C >
C < n1 >< n2 ><ref1><ref2>< R >< A >< B ><Leng><<>0
VR002 XL011B      .05  30.      0
VR001 XL011A      .05  30.      0
VR003 XL011C      .05  30.      0
VE002 X0001C      5.  120.     0
VE003 X0001A      5.  140.     0
VE001 X0001B      6.  100.     0
XF0001      .5      0
XF0003      .5      0
XF0002      .5      0

```

Fonte: Elaborado pelo autor (2022).

Na Figura 32 é possível observar a interface de criação dos gráficos de resultados onde tem-se o eixo “X” como um eixo de tempo e o eixo “Y” contendo a variável de saída, que no caso são as tensões e correntes trifásicas do sistema.

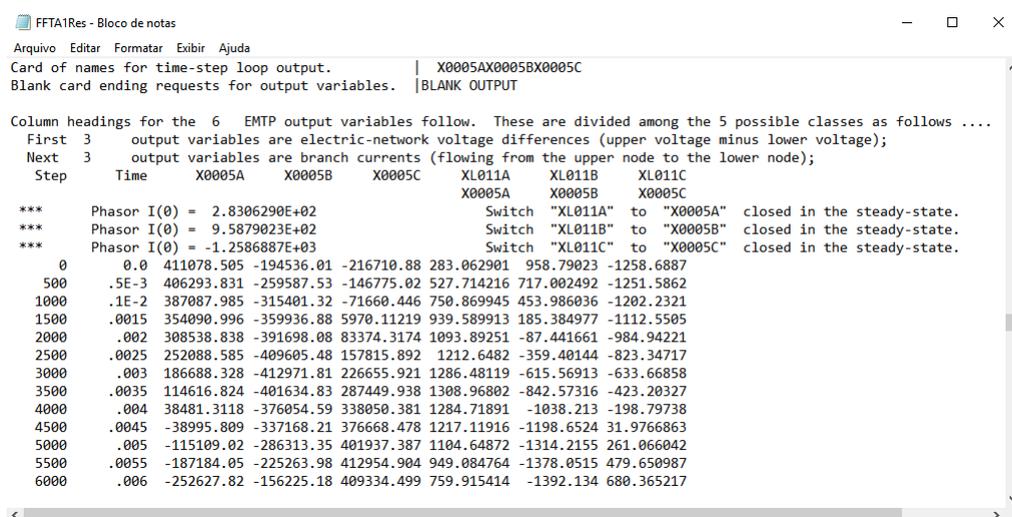
Figura 32 – Interface PlotXY.



Fonte: Elaborado pelo autor (2022).

O Resultado da execução do ATP gera um arquivo como o apresentado na Figura 33. Como esse arquivo contém todos os dados referentes a simulação, não apenas os resultados, é preciso que apenas os resultados sejam extraídos desse arquivo. Isso é feito através da função “extrairResultados” contida na biblioteca disponibilizada por Vieira (2016). Assim o resultado extraído desse arquivo fica como exibido na Figura 34.

Figura 33 – Arquivo de dados da execução do ATP.



Fonte: Elaborado pelo autor (2022).

Figura 34 – Resultados extraídos do arquivo de dados da execução do ATP.

Step	Time	X0009A	X0009B	X0009C	X0079A	X0079B	X0079C
500	.5E-3	406678.564	-259522.43	-146919.49	566.18713	723.513023	-1266.033
1000	.1E-2	387391.022	-315341.3	-71742.191	781.173136	459.988077	-1210.4064
1500	.0015	354338.399	-359881.69	5919.98922	964.329701	190.904078	-1117.5626
2000	.002	308727.225	-391648.15	83360.6426	1112.73072	-82.448774	-986.30943
2500	.0025	252187.923	-409562.9	157861.746	1222.58137	-355.14356	-818.76152
3000	.003	186698.453	-412932.27	226760.974	1287.49309	-611.61511	-623.16302
3500	.0035	114559.163	-401601.44	287590.849	1303.20127	-839.23393	-409.11189
4000	.004	38353.843	-376032.29	338225.661	1271.97144	-1035.9834	-181.26922
4500	.0045	-39204.328	-337152.82	376888.31	1196.2667	-1197.1135	53.9600874
5000	.005	-115387.09	-286299.87	402188.932	1076.8414	-1312.8675	286.220695
5500	.0055	-187506.57	-225257.92	413218.097	916.831819	-1377.4452	505.97046
6000	.006	-252988.04	-156230.1	409605.863	723.89241	-1392.6263	707.501708
6500	.0065	-309504.72	-81671.877	391488.134	505.566234	-1358.8854	884.469171
7000	.007	-355067.76	-4196.8269	359492.626	268.257393	-1274.6493	1029.19215
7500	.0075	-388068.39	73456.859	314746.914	19.8404845	-1142.2621	1135.95569
8000	.008	-407324.12	148511.344	258850.004	-229.53966	-969.10519	1202.36421
8500	.0085	-412149.55	218304.58	193785.425	-470.72513	-761.63331	1226.40062
9000	.009	-402382.89	280382.539	121848.564	-696.08552	-525.35312	1206.2574
9500	.0095	-378372.02	332546.795	45585.8098	-897.8271	-268.5633	1142.44731
10000	.01	-340958.59	372932.009	-32292.126	-1067.9091	-2.1000668	1038.1373
10500	.0105	-291467.02	400107.834	-109025.46	-1200.2081	264.628873	897.11441
11000	.011	-231656.69	413123.502	-181902.19	-1290.6496	523.36935	723.743048
11500	.0115	-163646.45	411516.005	-248340.7	-1336.0343	764.763368	524.157787

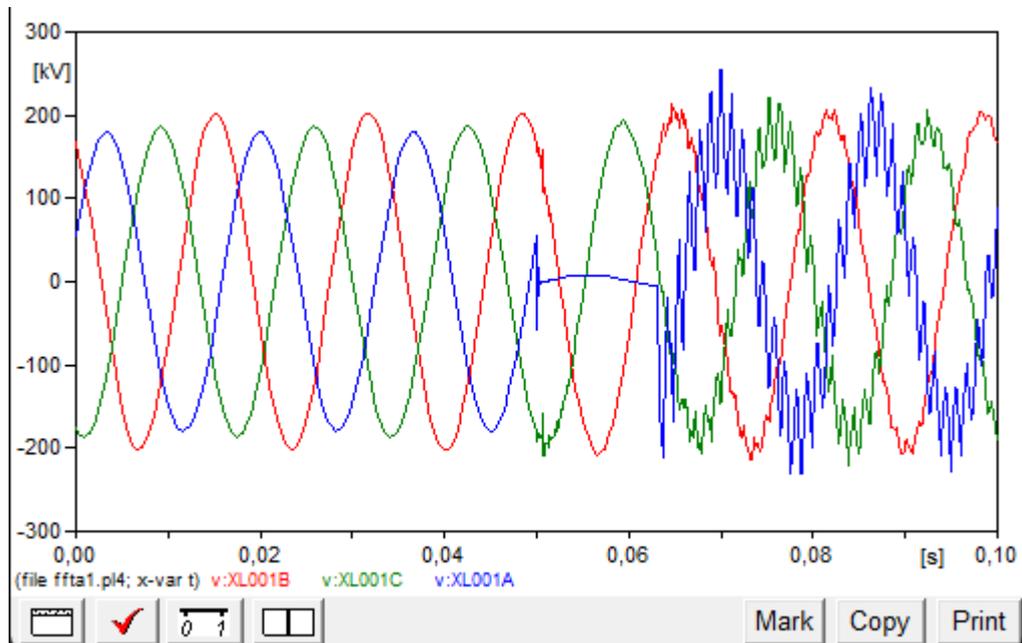
Fonte: Elaborado pelo autor (2022).

Através do algoritmo utilizado, executa-se os arquivos de simulação para se obter os resultados e assim ser possível extrair dados de corrente e tensão de cada fase da linha de transmissão. Foram feitas simulações de sete faltas diferentes dentro de um comprimento de 99 quilômetros para que a rede consiga detectar a distância e o tipo da falta ocorrida na linha de transmissão.

Podemos observar nas faltas monofásicas que no momento da falta a fase onde ocorre a falta tem sua tensão reduzido a praticamente zero causando um distúrbio grande na fase. Quando a mesma é mantida podemos observar a presença de harmônicos de múltiplas frequências no sinal, não só na fase que sofre a falta mas também nas outras duas fases. O comportamento da tensão das faltas monofásicas são exemplificadas nas Figuras 35, 36 e 37

- Falta A-T

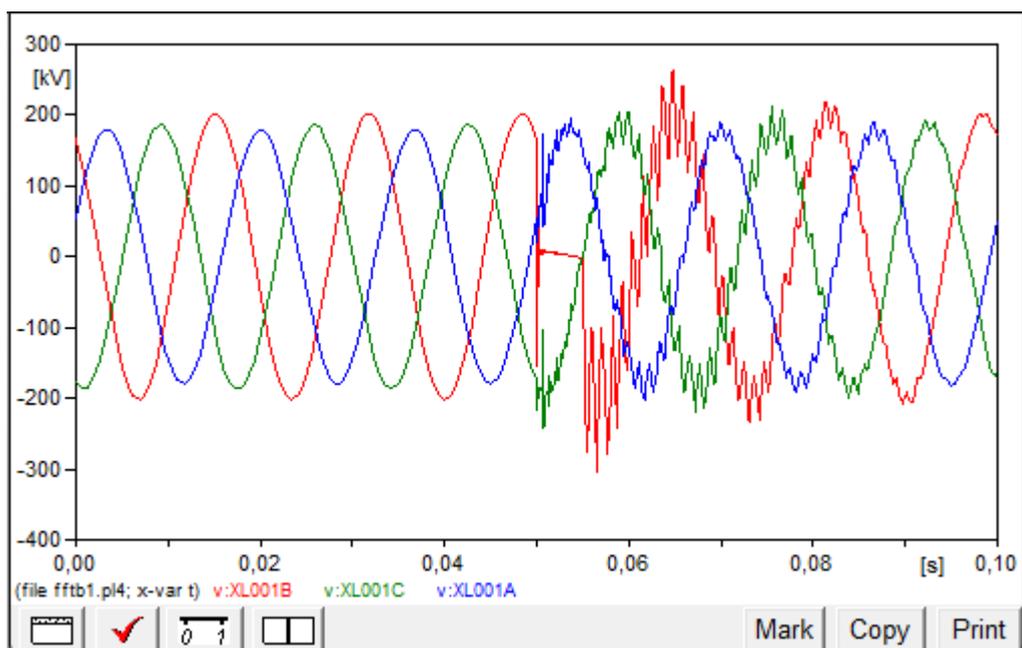
Figura 35 – Comportamento da tensão da rede em uma falta da fase A para a terra



Fonte: Elaborado pelo autor (2022).

- Falta B-T

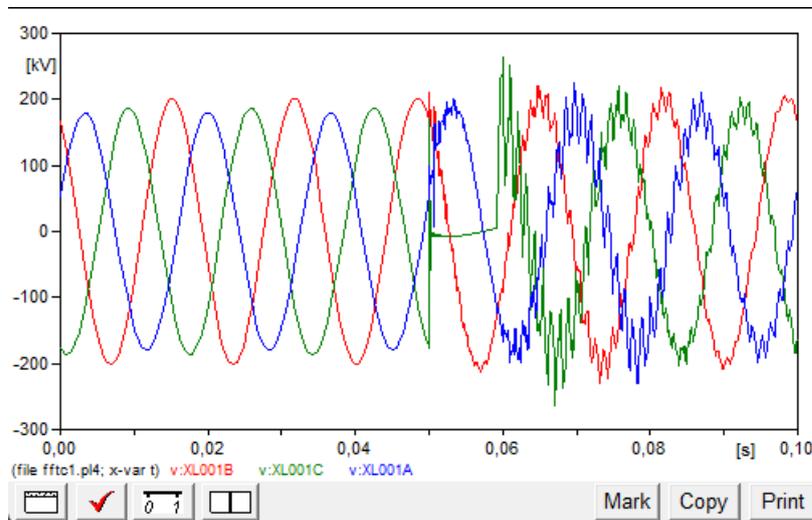
Figura 36 – Comportamento da tensão da rede em uma falta da fase B para a terra



Fonte: Elaborado pelo autor (2022).

- Falta C-T

Figura 37 – Comportamento da tensão da rede em uma falta da fase C para a terra

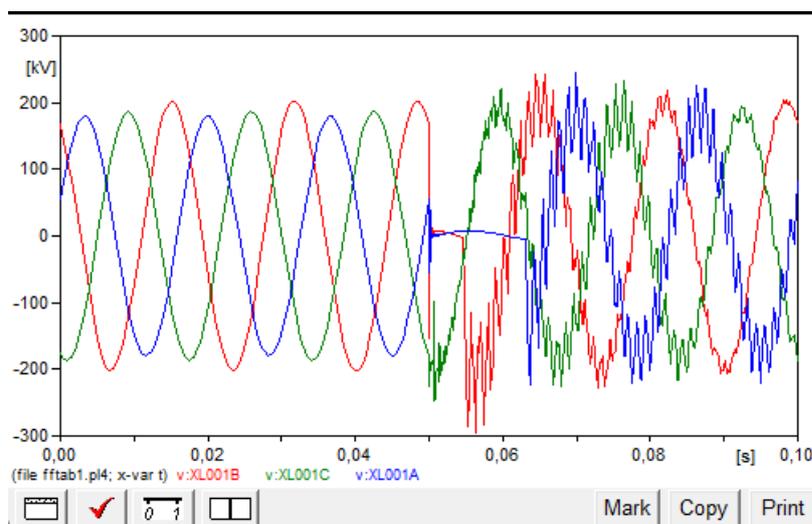


Fonte: Elaborado pelo autor (2022).

Nas faltas bifásicas podemos observar a ocorrência de falta em duas fases, onde é possível observar que a tensão de duas fases chegam próximo de zero, surgindo também aqui as múltiplas frequências harmônicas que também reflete na fase que não sofre a falta. Exemplos de faltas bifásicas ocorridas no quilômetro um se encontra nas Figuras 38, 39 e 40

- Falta AB-T

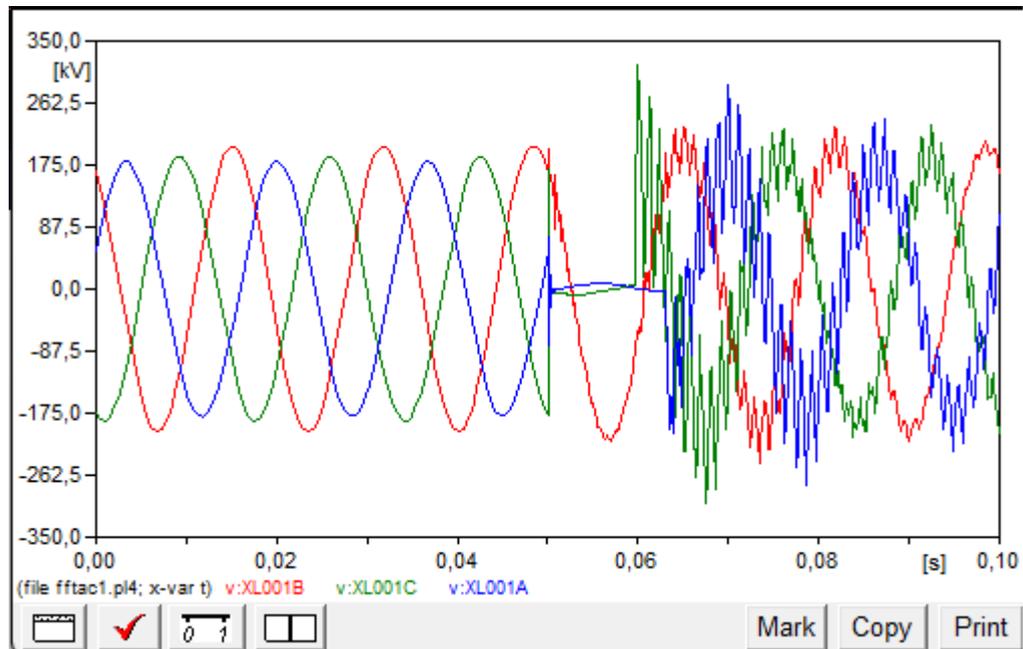
Figura 38 – Comportamento da tensão da rede em uma falta das fases AB para a terra



Fonte: Elaborado pelo autor (2022).

- Falta AC-T

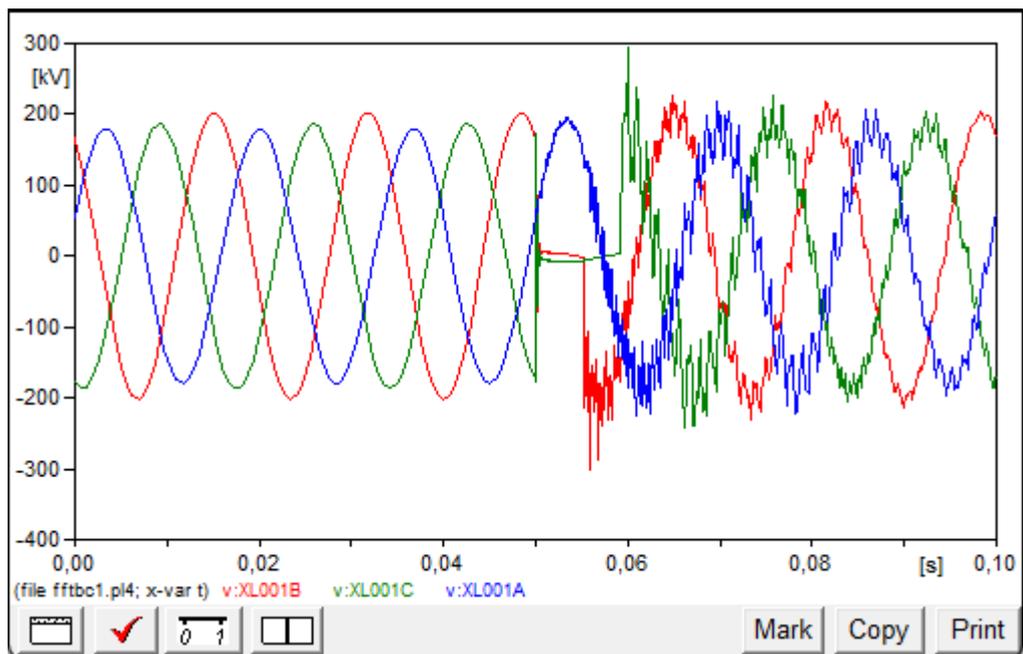
Figura 39 – Comportamento da tensão da rede em uma falta das fases AC para a terra



Fonte: Elaborado pelo autor (2022).

- Falta BC-T

Figura 40 – Comportamento da tensão da rede em uma falta das fases BC para a terra

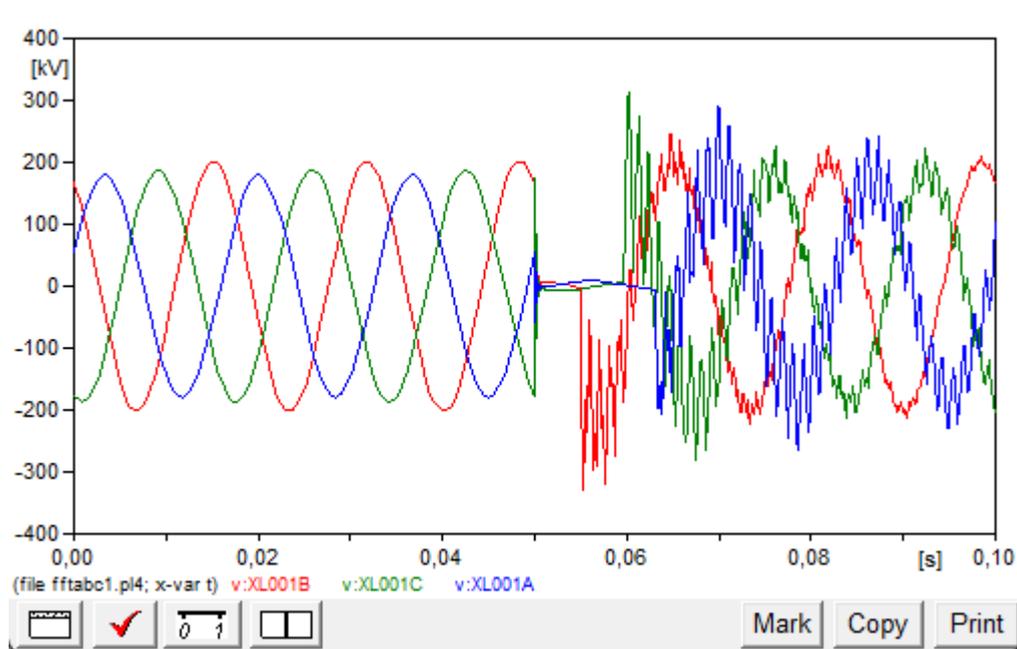


Fonte: Elaborado pelo autor (2022).

Por fim podemos observar a falta trifásica a terra, onde é possível observar a ocorrência da falta em todas as fases, vindo a causar um distúrbio e frequência harmônica nas três fases conforme podemos observar na Figura 41.

- Falta ABC-T

Figura 41 – Comportamento da tensão da rede em uma falta das fases ABC para a terra



Fonte: Elaborado pelo autor (2022).

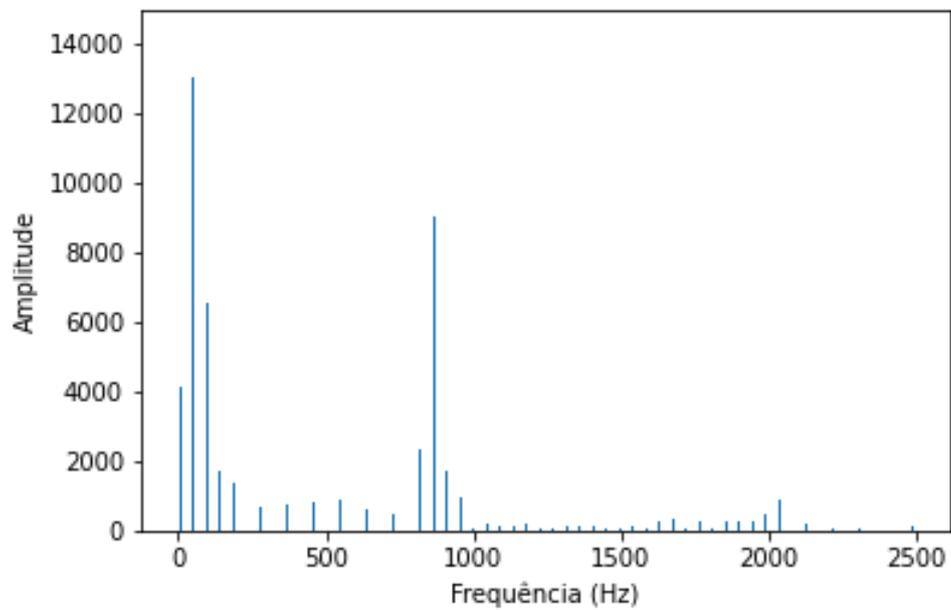
Podemos ver que a amplitude da tensão vai a valores próximos de 0 na fase em que está ocorrendo a Falta, além disso é nítido perceber que o distúrbio causado, afeta também as outras fases, gerando múltiplas frequências no sinal.

Para um melhor tratamento dos dados e detecção do ponto de onde a falta ocorreu, além também para obter a composição espectral da frequência tanto do sinal, foi aplicada a transformada de Fourier aos dados de entrada contendo os valores instantâneos de tensão e de corrente no sistema elétrico para cada caso com respectiva consição de falta. Assim que aplicado a transformada de Fourier nos dados, os mesmos foram normalizados para um melhor desempenho da rede.

Para exemplificar os dados após a aplicação da Transformada de Fourier, nas Figuras 42, 43, 44 e 45, segue o comportamento das frequência de uma fase quando temos uma falta monofásica, bifásica e trifásica.

- Tensão na fase A quando ocorre uma falta monofásica para a terra.

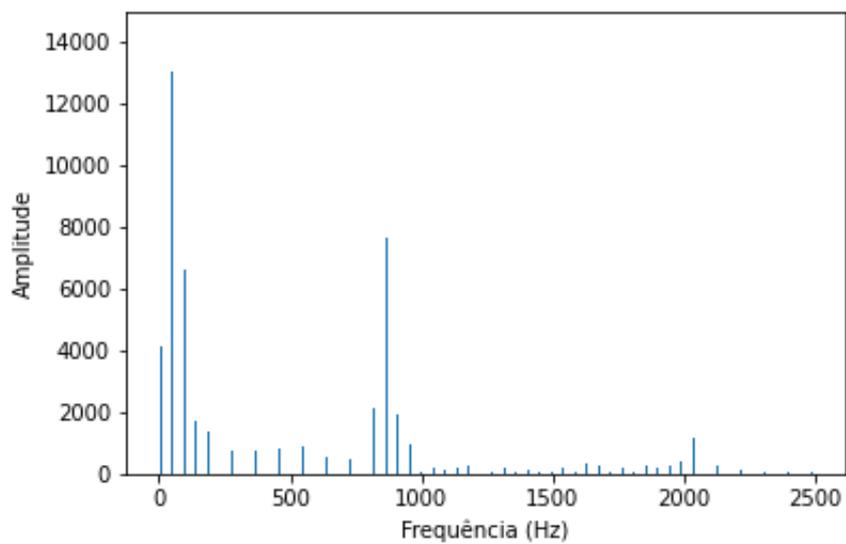
Figura 42 – Comportamento da tensão fase A durante uma falta monofásica



Fonte: Elaborado pelo autor (2022).

- Tensão fase A em uma falta bifásica

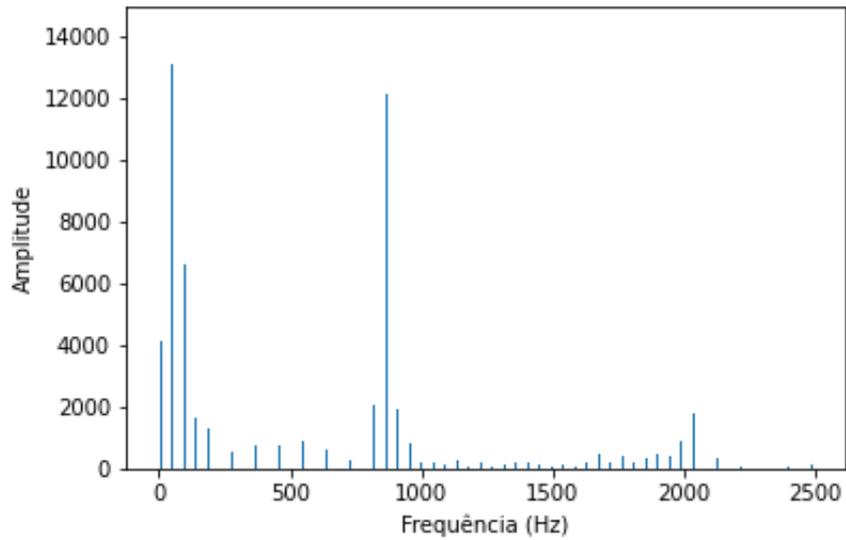
Figura 43 – Comportamento da tensão fase A durante uma falta bifásica



Fonte: Elaborado pelo autor (2022).

- Tensão fase A em uma falta trifásica

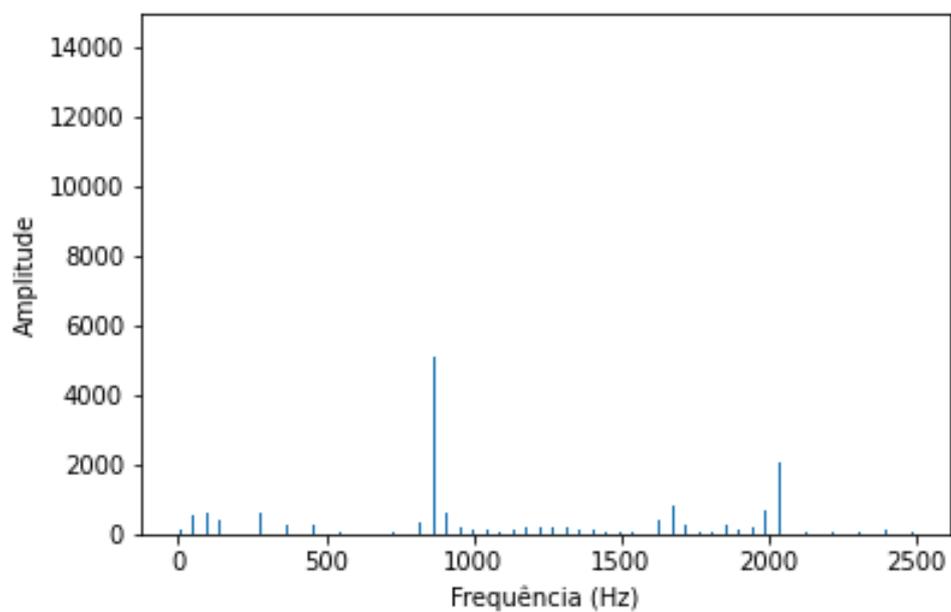
Figura 44 – Comportamento da tensão fase A durante uma falta trifásica



Fonte: Elaborado pelo autor (2022).

- Tensão fase A quando a mesma não participa da falta

Figura 45 – Comportamento da tensão fase A durante uma falta BC com a terra fase



Fonte: Elaborado pelo autor (2022).

Com a ferramenta KerasTuner da biblioteca Keras, é possível buscar pelos melhores hiperparâmetros onde a rede neural irá ter o seu melhor desempenho. Como tem-se a utilização de duas redes neurais, foram feitas duas buscas pelos melhores hiperparâmetros. O resultado das 10 melhores combinações de hiperparâmetros encontrados para a rede neural que busca a distância da falta é apresentados na Tabela 4.

Tabela 4 – 10 melhores hiperparâmetros encontrados pelo KerasTuner para rede de detecção da distância da falta

Neurônios LSTM	Passo de aprendizagem	Precisão
64	0.001	0.6195
128	0.01	0.5252
160	0.01	0.5353
224	0.01	0.5017
256	0.01	0.4983
384	0.001	0.5993
416	0.001	0.5825
416	0.01	0.4781
448	0.001	0.6027
480	0.001	0.5892

Fonte: Elaborado pelo autor (2022).

Pode-se observar através da Tabela 4 que temos a maior pontuação onde temos 64 neurônios na camada LSTM e um passo de aprendizagem de 0.001. Assim com essa informação cria-se o modelo de rede neural com estes hiperparâmetros.

As 10 melhores combinações de hiperparâmetros encontrados para a rede neural que classifica o tipo de falta se encontra na Tabela 5.

Tabela 5 – 10 melhores hiperparâmetros encontrados pelo KerasTuner para rede de detecção do tipo de falta

Neurônios LSTM	Passo de aprendizagem	Precisão
480	0.01	0.8889
448	0.01	0.8855
512	0.01	0.8855
64	0.001	0.8855
288	0.001	0.8821
160	0.001	0.8821
384	0.001	0.8821
416	0.001	0.8788
448	0.0001	0.8585
512	0.0001	0.8585

Fonte: Elaborado pelo autor (2022).

Foi feito também o experimento com passos de aprendizagem maiores para confirmar a eficácia dos valores utilizados. Assim foi possível comprovar que utilizar passos de aprendizagem muito altos pode conduzir a piores resultados conforme Tabelas 6 e 7.

Tabela 6 – Busca dos melhores hiperparâmetros com passos de aprendizagem de 0.1 e 1 para rede de distância da falta.

Neurônio LSTM	Passo de aprendizagem	Precisão
96	0.1	0.0266
480	1.0	0.0213
224	1.0	0.0213
256	1.0	0.0213
352	0.1	0.0186
160	1.0	0.0186
128	1.0	0.0186
384	1.0	0.0186
288	1.0	0.0186
384	0.1	0.0159

Fonte: Elaborado pelo autor (2022).

Tabela 7 – Busca dos melhores hiperparâmetros com passos de aprendizagem de 0.1 e 1 para rede de detecção do tipo de falta.

Neurônio LSTM	Passo de aprendizagem	Precisão
128	0.1	0.7420
96	0.1	0.7314
256	0.1	0.6330
320	0.1	0.5957
192	0.1	0.5824
448	0.1	0.5798
416	0.1	0.5479
352	1.0	0.3245
288	1.0	0.3058
128	1.0	0.2686

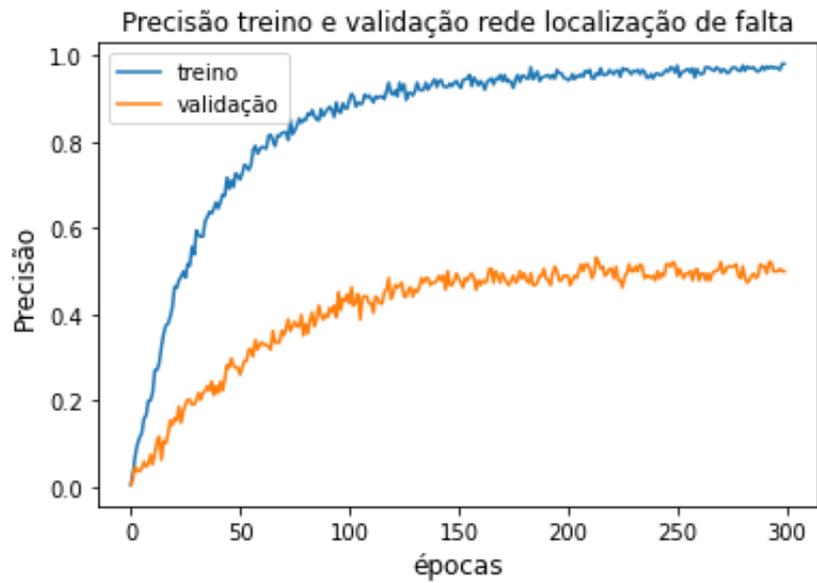
Fonte: Elaborado pelo autor (2022).

Assim, pode-se concluir que as redes neurais tem as seguintes precisões:

- Rede neural para tipo de falta: 88,88% no “*tuning*” e 84,57% no treino;
- Rede Neural para localização da falta: 61,95% no “*tuning*” e 50,00% no treino.

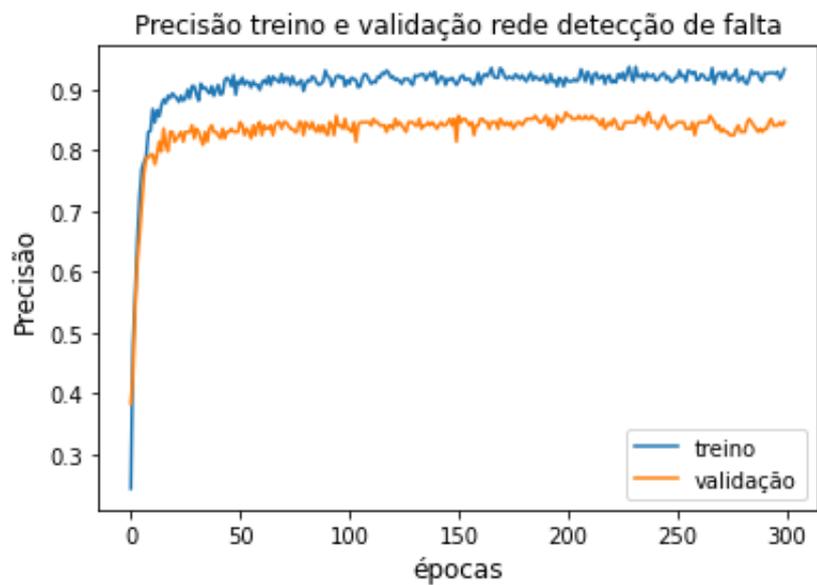
Nas Figuras 46 e 47 tem-se graficamente a precisão em cada época do teste das redes.

Figura 46 – Gráfico da rede neural de localização da falta.



Fonte: Elaborado pelo autor (2022).

Figura 47 – Gráfico da rede neural de detecção de falta.



Fonte: Elaborado pelo autor (2022).

É de suma importância frisar que a precisão para a rede neural localizadora refere-se a detecção exata do ponto da falta, porém a mesma aumenta caso o "range" de detecção for maior, conforme explicitado nas Tabelas 8 e 9.

Tabela 8 – Exemplos de detecção de distância

Referência	Saída da rede neural	Deteccção exata	Deteccção com erro de 1 Km a jusante	Deteccção com erro de 1 Km a montante
0	0	X		
1	0		X	
3	3	X		
5	5	X		
8	8	X		
16	17			X
22	23			X
25	26			X
26	26	X		
28	28	X		
29	29	X		
30	30	X		
31	32			X
33	33	X		
34	35			X
35	35	X		
36	35		X	
37	38			X
38	38	X		
39	39	X		
44	45			X
45	45	X		
46	47			X
47	46		X	
48	48	X		
57	52			
58	57		X	
64	64	X		
68	68	X		
70	69		X	
75	74		X	
76	75		X	
95	96			X
96	96	X		
97	97	X		

Fonte: Elaborado pelo autor (2022).

Tabela 9 – Precisão para cada "range" de distância

Detecção exata	Detecção com erro +/- 1 Km	Detecção com erro +/- 2 Km	Detecção com erro +/- 3 Km	Detecção com erro +/- 4 Km
50%	90,426%	96,277%	97,872%	98,404%

Fonte: Elaborado pelo autor (2022).

Apesar das limitações de ser uma simulação e de se ter uma dificuldade na obtenção de variações para conseguir os dados de tipos de falta e principalmente localização, foi possível concluir que os resultados obtidos, principalmente para a rede localizadora foram satisfatórios, pois em um *range* razoável, obteve-se um precisão na casa dos 90%. A principal limitação da rede neural foi a disponibilidade dos dados, percebendo-se nos gráficos 46 e 47 um *overfitting* devido a isso, principalmente na rede localizadora. Isso foi causado pois a simulação de dados de distância é muito limitada, tendo poucas variáveis para serem alteradas na geração de dados na simulação.

5 CONCLUSÃO

Foi possível analisar com este trabalho a possibilidade de determinar a localização e classificação de faltas com redes neurais recorrentes sem necessitar de uma intervenção invasiva na rede elétrica, já que necessitamos apenas de medições de corrente e tensão.

Foram obtidos 693 amostras de faltas monofásicas com a terra, bifásicas com a terra e trifásicas com a terra através de simulações no ATP que se mostrou um software bastante útil, pois além de permitir a simulação de distúrbios na rede elétrica, permite que essas simulações possam ser configuradas por algoritmos externos, permitindo uma gama de personalização das simulações. Como no exemplo deste trabalho, utilizou-se da linguagem Python para controlar as centenas de iterações necessárias para obter os dados de cada tipo de falta.

Pode-se inferir a partir dos resultados obtidos que a transformada de Fourier como método de pré-processamento de dados foi muito útil para o funcionamento das redes neurais na medida que através dos resultados observou-se que esse método conseguiu decompor os sinais de tensão e de corrente nas diferentes frequências que compõem estas grandezas as diferentes frequências geradas no distúrbio da falta, permitindo uma maior qualidade nos dados de treinamento da rede neural.

Com o “*tuning*” das duas redes neurais, através das várias iterações foi possível a busca pela melhor valor onde como resultado final foi obtido uma precisão de 88,88% no “*tuning*” e 84.57% no teste e para a rede neural de detecção do trecho da falta chegou-se a uma precisão de 61.95% no “*tuning*” e 50.00% no teste para detecção exata do trecho da falta, sendo essa precisão aumentada para 90.27% no teste no caso onde aumentamos o trecho em um quilômetro a montante e a jusante.

REFERÊNCIAS

- ABADI, M. *et al.* **TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems**. 2015. Disponível em: <<https://www.tensorflow.org/>>. Acesso em: 08 jul. 2022.
- ACADEMY, D. S. **Como Funciona o Dropout?** Data Science Academy, 2018. Disponível em: <<https://www.deeplearningbook.com.br/capitulo-23-como-funciona-o-dropout/>>. Acesso em: 08 jul. 2022.
- BENTO, C. **Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis**. Towards Data Science, 2021. Disponível em: <<http://tiny.cc/kyksuz>>. Acesso em: 09 jul. 2022.
- CACERES, P. **The Multilayer Perceptron - Theory and Implementation of the Backpropagation Algorithm**. 2020. Disponível em: <<https://pabloinsente.github.io/the-multilayer-perceptron>>. Acesso em: 08 jul. 2022.
- CECCON, D. **A importância do bias nas redes neurais**. 2020. Disponível em: <<https://iaexpert.academy/2020/09/28/importancia-do-bias-nas-redes-neurais>>. Acesso em: 08 jul. 2022.
- CHOLLET, F. *et al.* Keras documentation. **keras.io**, v. 33, 2015. Acesso em: 08 jul. 2022.
- COURY, D. V.; OLESKOVICZ, M.; GIOVANINI, R. **Proteção digital de sistemas elétricos de potência: dos relés eletromecânicos aos microprocessados inteligentes**. São Carlos: EESC-USP, 2007.
- DIEFENTHALER, A. T. Modelagem matemática de linhas de transmissão baseada em dados reais da rede de distribuição primária de energia elétrica. 2019.
- Estado de Minas. **Consumo total de energia no Brasil deve crescer 2,2% ao ano até 2040, estima BP**. 2019. Disponível em: <https://www.em.com.br/app/noticia/economia/2019/02/14/internas_economia,1030618/consumo-total-de-energia-no-brasil-deve-crescer-2-2-ao-ano-ate-2040.shtml>.
- FALCÃO, J. V. R.; MOREIRA, V. de Á.; RAMOS, C. de Á.; SANTOS, F. A. de O. Redes neurais deep learning com tensorflow. **RE3C-Revista Eletrônica Científica de Ciência da Computação**, v. 14, n. 1, 2019.
- FUCHS, R. D. **Transmissão de energia elétrica: linhas aéreas; teoria das linhas em regime permanente**. Livros Técnicos e Científicos. Itajubá: Escola Federal de Engenharia, 1977. v. 1.
- GLOVER, J. D.; SARMA, M. S.; OVERBYE, T. **Power system analysis & design**. Stanford: Cengage Learning, 2012.
- GRAINGER, J. J. **Power system analysis**. Singapore: McGraw-Hill, 1999.
- HAYKIN, S. **Redes neurais: princípios e prática**. Porto Alegre: Bookman Editora, 2007.
- LIPTON, Z. C.; BERKOWITZ, J.; ELKAN, C. **A Critical Review of Recurrent Neural Networks for Sequence Learning**. 2015. Disponível em: <<https://arxiv.org/abs/1506.00019>>. Acesso em: 08 jul. 2022.
- MANÉ, D. *et al.* Tensorboard: Tensorflow's visualization toolkit. **Retrieved October**, v. 8, p. 2021, 2015.

MARIO, M. C.; FILHO, J. I. da S.; ABE, J. M. Rede neural artificial híbrida–rede para-neural: Implementação da lógica paraconsistente em neurônios artificiais. **ANAIS ASPECTOS DE SISTEMAS INTELIGENTES BASEADOS EM LÓGICAS ANOTADAS**, p. 192, 2021.

MARSILIO, L. T. **Otimização paramétrica de um suporte de suspensão mecânica dos eixos traseiros de caminhões**. 79 p. Monografia (Bacharel) — Universidade de Caxias do Sul, Caxias do Sul, 2015.

MARTI, J. R. Accurate modelling of frequency-dependent transmission lines in electromagnetic transient simulations. **IEEE transactions on power apparatus and systems**, IEEE, n. 1, p. 147–157, 1982.

MOREIRA, C. **Análise de Curto-Circuitos Simétricos**. 2010.

NILSSON, J. W.; RIEDEL, S. A. **Circuitos elétricos, 8a. edição**. São Paulo: Pearson Prentice Hall, 2009.

OLAH, C. **Understanding LSTM Networks**. 2015. Disponível em: <<https://colah.github.io/posts/2015-08-Understanding-LSTMs/>>. Acesso em: 08 jul. 2022.

ORAMUS, P.; FLORKOWSKI, M. Influence of various transmission line models on lightning overvoltages in insulation coordination studies. **Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej**, 2014.

PYKES, K. **The Vanishing/Exploding Gradient Problem in Deep Neural Networks**. 2020. Disponível em: <<http://tiny.cc/jegtuz>>. Acesso em: 10 jul. 2022.

RAJ, D. **Single-layer Neural Networks in Machine Learning (Perceptrons)**. 2020. Disponível em: <<http://tiny.cc/negtuz>>. Acesso em: 10 jul. 2022.

RASCHKA, S. **Introduction to Deep L: Introduction to Deep Learning and Generative Modelsearning and Generative Models**. 2021. Disponível em: <https://sebastianraschka.com/pdf/lecture-notes/stat453ss21/L15_intro-rnn__slides.pdf>. Acesso em: 11 jul. 2022.

REDDY, M. J. B.; MOHANTA, D. K. Performance evaluation of an adaptive-network-based fuzzy inference system approach for location of faults on transmission lines using monte carlo simulation. **IEEE Transactions on Fuzzy Systems**, IEEE, v. 16, n. 4, p. 909–919, 2008.

REIS, A.; JR, A. R.; CAIXETA, D. **Curso de ATPDraw**. Uberlândia, 2012.

SILVA, I. d.; SPATTI, D. H.; FLAUZINO, R. A. Redes neurais artificiais para engenharia e ciências aplicadas. **São Paulo: Artliber**, v. 23, n. 5, p. 33–111, 2010.

SILVA, M. d. **Localização de faltas em linhas de transmissão utilizando a teoria de ondas viajantes e transformada wavelet**. São Carlos: Universidade de São Paulo, 2003.

SOTERO, R. T.; ALBUQUERQUE, M. C. S. de; PAULA, F. G. de; FARIAS, C. T.; FILHO, E. F. de S. Classificação de sinais ultrassônicos pré-processados pela transformada de fourier através das redes neurais artificiais utilizando a técnica pulso eco para identificação de defeitos em juntas soldadas de aços estruturais. In: **VII CONNEPI-Congresso Norte Nordeste de Pesquisa e Inovação**. [S.l.: s.n.], 2012.

VIEIRA, P. H. N. **Localização de faltas fase-terra em linhas de transmissio utilizando rede neural treinada com dados simulados com o EMTP-ATP**. Dissertação (Mestrado) — Centro Universitario de Barra Mansa – UBM Cicuta, 2016.

YE, L.; SUN, H. B.; SONG, X. R.; LI, L. C. Dynamic modeling of a hybrid wind/solar/hydro microgrid in emtp/atp. **Renewable Energy**, Elsevier, v. 39, n. 1, p. 96–106, 2012.

YU, Y.; SI, X.; HU, C.; ZHANG, J. A review of recurrent neural networks: Lstm cells and network architectures. **Neural computation**, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info, v. 31, n. 7, p. 1235–1270, 2019.