DOI: 10.48011/asba.v2i1.1536

Classificação de faltas em linhas de transmissão utilizando florestas aleatórias e filtro *notch*

Danton Diego Ferreira [*] **Gabriel Aparecido Fonseca** [**]
Flávio Bezerra Costa [***] **Aryfrance Rocha Almeida** [****]

[*] *Departamento de Engenharia – Universidade Federal Lavras, MG, (e-mail: danton@ufla.br).*
[**] *Departamento de Engenharia – Universidade Federal Lavras, MG, (e-mail: gabriel.fonseca@estudante.ufla.br)*
[***] *Escola de Ciências e Tecnologia – Universidade Federal do Rio Grande do Norte, RN, (e-mail: flaviocosta@ect.ufrn.br)*
[****] *Departamento de Engenharia Elétrica – Universidade Federal do Piauí, PI, (e-mail: aryfrance@ufpi.edu.br)*

Abstract: Overhead energy transmission lines are highly susceptible to fail. Some researchers already studied the use of computational intelligence techniques like artificial neural networks and fuzzy logic with pre-processing phases that comprise the use of wavelet, Fourier transform or higher-order statistics. This work aims to show the use of the random forest method with a pre-processing step with a notch filter to classify faults in transmission lines. The performance of the model was compared with that obtained by a neural network to show its efficiency. Using k-fold cross-validation to train, test and compare the models, it was obtained the mean accuracy of 89.59% for the neural network and 91.96% for the random forest. In the validation process, it was obtained accuracy of 96.49% for the first model and 91.49% for the second.

Resumo: Sistemas de transmissão de energia elétrica externos estão altamente susceptíveis a falhas. Alguns pesquisadores já investigaram o uso de técnicas de inteligência computacional como redes neurais artificiais e lógica *fuzzy* com etapas de pré-processamento que compreendem o uso de transformada *wavelet*, Fourier ou estatística de ordem superior. Esse trabalho visa mostrar o uso do método de florestas aleatórias com uma etapa de pré-processamento com filtro *notch* para classificação de faltas em linhas de transmissão. O desempenho do modelo foi comparado com o obtido por uma rede neural para mostrar sua eficiência. Utilizando-se a validação cruzada *k-fold* para treinar, testar e comparar os modelos, obteve-se nesse processo a acurácia média de 89,59% para a rede neural e 91,96% para o modelo de florestas aleatórias. No processo de validação foi obtida acurácia de 96,49% para o primeiro modelo e 91,49% para o segundo.

Keywords: Random forest; artificial neural networks; notch filter; transmission lines; fault classification; cross validation.

Palavras-chaves: Florestas aleatórias; redes neurais artificiais; filtro notch; linhas de transmissão; classificação de faltas; validação cruzada.

1. INTRODUÇÃO

Com a crescente demanda de energia elétrica, os sistemas de transmissão têm sofrido com faltas e falhas diversas, por serem componentes altamente suscetíveis a intempéries e eventos externos. Descargas elétricas, animais e erros humanos podem levar à ocorrência de faltas nas linhas de transmissão. Dessa maneira, surgiram projetos que visam a criação de sistemas e métodos protetivos eficazes que visam evitar ou minimizar os prejuízos à concessionária de energia e aos consumidores Tais sistemas, em geral, são compostos por três etapas que compreendem a detecção, a classificação e a localização da falta. No trabalho aqui proposto somente a etapa de classificação será abordada.

Em Avagaddi et al. (2017) são apresentados os tipos de faltas que podem ocorrer em uma linha de transmissão. As faltas mais comuns e menos graves são aquelas que envolvem apenas uma fase (AT, BT, CT). Faltas bifásicas (AB, BC e CA) são um pouco mais raras e severas. Aquelas que envolvem duas fases e o terra (ABT, BCT e CAT) são ainda menos comuns e mais nocivas. Por fim, faltas trifásicas (ABC) e trifásicas com terra (ABCT) ocorrem raramente, mas quando acontecem podem levar ao colapso do sistema elétrico como um todo.

Uma técnica amplamente utilizada no processo de classificação de faltas em linhas de transmissão é a rede neural. Dentre os autores que fizeram uso dela, pode-se citar R. de Carvalho et al. (2014) que utilizaram os sinais de tensão

com uma etapa de pré-processamento com estatística de ordem superior (*higher-order statistics* - HOS). Após isso os sinais foram apresentados à uma rede neural com 3 entradas, 40 neurônios na camada oculta e 10 saídas, obtendo uma acurácia de 99% na classificação com dados simulados. Kumar et al. (2014) também fizeram uso de redes neurais, no entanto, não foi incluída etapa de pré-processamento e como sinais de entrada foram utilizados os valores RMS da tensão e corrente em uma arquitetura com 6 entradas, 5 neurônios na camada escondida e 4 saídas. Saravanan and Rathinam (2012) realizaram uma pesquisa comparativa entre três arquiteturas de redes neurais *feedforward*, a rede *multilayer perceptron* (MLP), a rede de função de base radial (RBF) e a rede em cascata para classificação e localização de faltas em linhas de transmissão de circuito duplo. O resultado da pesquisa desses autores indicou que a rede RBF tem o menor tempo de treinamento enquanto a rede em cascata obteve o menor erro.

Alguns autores exploraram o uso de outros métodos para a resolução desse problema. Almeida et al. (2017) utilizaram a análise de componentes independentes (*independent component analysis* - ICA) como etapa de pré-processamento e filtragem do sinal para uso em um classificador baseado em máquinas de vetor de suporte (*support vector machine* - SVM), realizando assim um comparativo entre a classificação do sinal filtrado e do sinal com ruído, obtendo acurácia de 100%. Das et al. (2005) realizaram um estudo comparando o uso da transformada de Fourier e *wavelet* na classificação de faltas em linhas de transmissão. Como conclusão nesse projeto, foi observado que para os dados e técnicas utilizadas a transformada *wavelet* fornece melhores resultados.

Observando o que já foi estudado na área, a motivação desse projeto é aplicar técnicas mais simples, que possuam menor custo computacional. Por esse motivo, na etapa de pré-processamento foi utilizado unicamente um filtro *notch* de segunda ordem, ele possui um custo computacional inferior às transformada de Fourier e *wavelet* que são geralmente utilizados nessa etapa em trabalhos dessa natureza. A busca pela simplicidade no pré-processamento reside no fato de que o uso de técnicas sofisticadas, apesar de levar a resultados melhores, também indica a necessidade de recursos computacionais superiores que podem não estar disponíveis. Na classificação, foi utilizado o método de florestas aleatórias, que em geral leva a um tempo de treinamento menor do que algumas técnicas de inteligência computacional. Foi possível confirmar a simplicidade do método proposto ao compará-lo com o modelo de redes neurais artificiais. Essa comparação permitiu observar, principalmente em relação ao tempo de treinamento, a eficiência do que foi proposto, enquanto a rede neural artificial demorou em média 1h e 30m para ser treinada o modelo usando florestas aleatórias executou esse processo em apenas 5 min.

2. MATERIAIS E MÉTODOS

2.1 Metodologia adotada nesse projeto

O conjunto de dados utilizado nesse trabalho foi obtido com a utilização do software MATLAB®, ele é composto

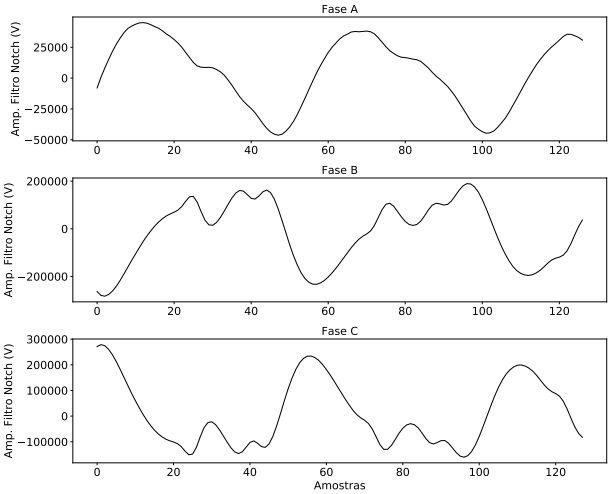

Figura 1. Exemplo de meio ciclo do sinal filtrado.

por 939 arquivos, no qual cada um deles representa uma ocorrência da falta. Dessa maneira, estão presentes 10 tipos de faltas (AT, BT, CT, AB, BC, CA, ABT, BCT, CAT, ABC), 3 distâncias da localização da falta (20km, 150km e 280km), variação do ângulo da falta entre 0°e 180° (com passo de 10°) e 3 diferentes valores para a resistência da falta (1 Ω, 50 Ω e 100 Ω). Além disso, a frequência fundamental do sinal é 60 Hz, e possui 256 amostras por ciclo, portanto, a frequência de amostragem é 15360 Hz. O diagrama de blocos do sistema elétrico analisado nesse projeto pode ser encontrado no final desse artigo na Figura 6.

A frequência fundamental representa uma redundância, já que é constante e presente em todas as fases, isso pode prejudicar modelos de inteligência computacional que utilizam sinais elétricos como entrada. Por esse motivo a frequência de 60 Hz foi eliminada utilizando-se um filtro *notch*. Esse é um filtro rejeita-faixa que deixa passar a maioria das frequências inalteradas, mas atenua aquelas em um intervalo específico. Sua banda de rejeição é muito estreita, ou seja, ele possui um alto fator de qualidade. Por possuir baixo custo computacional e sua maior seletividade das frequências de interesse ele é usado em trabalhos de processamento de sinais. Neste projeto considerou-se um filtro de segunda ordem com valor constante de 60Hz para frequência de corte. Sua transformada z é dada pela equação 1 Ferreira (2010).

$$H(s) = \frac{1 + a_0 z^{-1} + z^{-2}}{1 + \rho_0 a0 z^{-1} + \rho_0^2 z^{-2}} \tag{1}$$

onde, $a_0 = -2\cos\omega_0$ e ρ_0 é o fator *notch* com $0 \ll \rho_o < 1$. Um valor que apresentou melhores resultados para detecção e classificação de distúrbios foi $\rho_0 = 0,97$ Ferreira (2010), e portanto nesse projeto utilizou-se o mesmo valor.

Ao utilizar o filtro *notch*, surge um sinal transitório que pode ser prejudicial ao desempenho do classificador, por esse motivo, as primeiras amostras do sinal filtrado foram removidas. Também, não é interessante fazer uso de todo o sinal para evitar a complexidade dos modelos criados, portanto, nesse trabalho fez-se uso de meio ciclo, o que compreende 128 amostras. Um exemplo desse sinal pode ser observado na Figura 1.

Além do pré-processamento do sinal citado, é importante também realizar a normalização dos dados para evitar que os modelos de aprendizado computacional sejam enviesados. Assim, cada uma das ocorrências de faltas a serem apresentadas aos classificadores é composta pelos sinais filtrados de cada uma das 3 fases após a normalização entre -1 e 1.

Na Figura 2 pode-se verificar o fluxograma apresentando as etapas desenvolvidas durante esse projeto. Os sinais de tensão de cada fase são utilizados como entradas para o filtro *notch*, os sinais de saída não normalizados e utilizados na validação cruzada. Foram implementados dois modelos de classificadores: florestas aleatórias (*random forest* - RF) e redes neurais artificiais (*artificial neural networks* - ANN). O fluxograma da Figura 2 é o mesmo para ambos, foi feito uso também da validação *k-fold* com 10 *folds* sendo que cada *fold* foi repetido dez vezes. Ao final obteve-se um vetor de acurácias, então foi calculada a média e desvio padrão para se obter o desempenho geral do modelo.

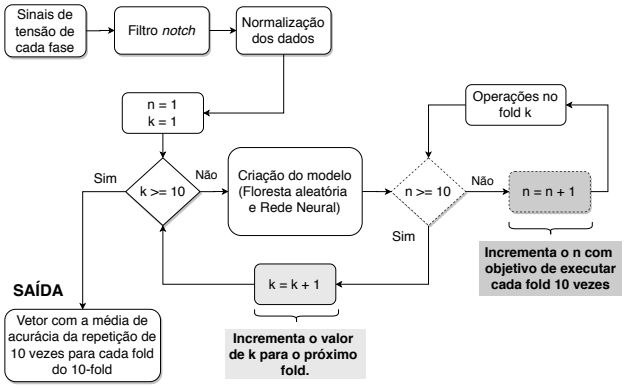

Figura 2. Fluxograma apresentando as etapas do projeto.

2.2 Florestas Aleatórias

Florestas aleatórias (*random forest* - RF) representam uma combinação de árvores de decisão de tal maneira que cada árvore depende dos valores de um vetor aleatório amostrado independentemente e com a mesma distribuição de todas árvores na floresta Breiman (2001). Esse é um método de classificação interessante pois pode ser usado em problemas de classificação e regressão e por ser menos suscetível ao sobre ajuste (*overfitting*) dos dados que pode ocorrer nas árvores de decisão quando elas crescem profundamente. Um problema desse método, no entanto, é a demora no treinamento dependendo da quantidade de árvores utilizadas na construção da floresta.

O modelo de floresta aleatória, validação cruzada e matriz de confusão foram implementados utilizando *Python*, para isso fez-se uso da biblioteca *scikit-learn* e algumas funções criadas para esse projeto. Na Tabela 1 pode-se observar alguns parâmetros utilizados no modelo, o significado de cada um deles e os valores utilizados. Parâmetros não apresentados na Tabela 1 foram utilizados com seus valores padrões, conforme documentação oficial do *scikit-learn* disponível em scikit learn (2019).

Tabela 1. Parâmetros utilizados no modelo de florestas aleatórias

Parâmetro	Significado	Valor usado
n_estimators	Quantidade de árvores na floresta.	50
max_depth	Profundidade máxima das árvores.	20
min_samples_split	Número mínimo de amostras necessárias para dividir um nó interno.	7
min_samples_leaf	Número mínimo de amostras necessárias para estar em um nó folha.	1
criterion	Função que mede a qualidade da divisão.	*'entropy'*
bootstrap	Todo conjunto de dados é usado para construir cada árvore.	False
max_features	Número de características consideradas quando procurar pela melhor divisão.	11

Em trabalhos de inteligência computacional é recomendável separar o conjunto de dados em duas partes. Uma maior porção é reservada para o treinamento dos modelos enquanto a segunda é usada para a validação e representa um conjunto desconhecido. Isso permite avaliar qual pode ser o comportamento dos modelos quando dados desconhecidos forem utilizados. Dessa maneira, nesse projeto, 90% (845 amostras) dos dados foram usados no processo de treinamento e 10% (94 amostras) para validação dos modelos. O primeiro conjunto foi utilizado durante a validação cruzada passando por divisões sucessivas nas quais partes do conjunto foram usadas para treinamento e partes para teste.

Nesse trabalho, utilizou-se o *k-fold* estratificado (com $k = 10$) o qual cria os *folds* de tal maneira que o percentual de amostras é preservado para cada classe. Cada *fold* foi repetido 10 vezes para se obter um resultado mais preciso dada a natureza aleatória dos modelos utilizados. Foi escolhido esse número de repetições uma vez que valores maiores do que esse não apresentaram uma diferença significativa no resultado final. Por fim, o tempo total de treinamento do método de florestas aleatórias incluindo as repetições foi de 5 minutos e 19 segundos.

2.3 Rede Neural Multicamadas

A rede neural multicamada (*multilayer perceptron* – MLP) consiste de um sistema de neurônios simples interconectados, ela representa um mapeamento não linear entre um vetor de entrada e um vetor de saída. Os neurônios são conectados por pesos e sinais de saída que são uma função da soma das entradas do neurônio modificada por uma função de ativação não-linear Gardner and Dorling (1998).

Segundo Murtagh (1991), para se definir uma rede neural artificial multicamadas é necessário determinar: a sua configuração ou arquitetura (número de camadas, número de neurônios por camada, etc); a função de ativação a ser usada nos neurônios; o método de treinamento a ser utilizado; o modo de atualização dos pesos (tempo real ou *off-line*).

Tendo em vista isso que foi apresentado, nesse projeto, utilizou-se o *Keras* que é uma API (*application programming interface*) de alto nível para implementação de redes neurais em *Python*. Após alguns testes, verificou-se que os parâmetros indicados na Tabela 2 foram aqueles que apresentaram o melhor resultado para os dados em questão.

Tabela 2. Parâmetros utilizados na rede neural implementada.

Parâmetro	Valor usado
Quantidade de entradas	384
Quantidade de camadas ocultas	1
Quantidade de saídas	10
Quantidade de neurônios na camada oculta	384
Inicialização dos pesos e bias	Inicialização Glorot
Função de ativação da camada oculta	Relu
Função de ativação da camada de saída	Softmax
Função de perda usada durante o treinamento	Entropia cruzada categórica
Algoritmo de treinamento utilizado	Adagrad
Taxa de treinamento utilizado	0,01
Atualização dos pesos	*Off-line* em lotes de 60
Quantidade de épocas	5000

Durante o treinamento da rede foi utilizada a técnica *droupout*, ela previne a ocorrência de *overfitting* e fornece uma maneira de combinar eficientemente diferentes arquiteturas de redes neurais. Para isso, neurônios são removidos temporariamente da rede junto com todas as suas conexões de entrada e saída Srivastava et al. (2014). A escolha das unidades a serem removidas é realizada de maneira aleatória e para esse projeto escolheu-se uma taxa de 20% para remoção dos neurônios.

Outra técnica utilizada foi a parada antecipada (*early stopping*), ela também evita o *overfitting* e é baseada no monitoramento de uma determinada métrica durante o treinamento. É realizada a constante verificação entre os conjuntos de treino e teste para averiguar se está ocorrendo melhoria ou não, interrompendo o processo no momento correto e evitando que a rede decore os dados de entrada.

Por fim, o processo de validação cruzada foi conduzido de maneira similar ao que foi exposto no final da subseção 2.2. Deve-se destacar que o tempo total de treinamento da rede incluindo as repetições realizadas foi de 1 hora, 28 minutos e 36 segundos.

3. RESULTADOS E DISCUSSÕES

Durante esse trabalho foi possível estudar e trabalhar alguns modelos de inteligência computacional. Especificamente, esse projeto tratou da aplicação de florestas aleatórias e redes neurais artificiais na classificação de faltas em linhas de transmissão. Para o primeiro modelo, obteve-se o seguinte vetor de acurácia com valores percentuais: [93.00, 89.28, 91.46, 92.46, 92.64, 93.22, 92.38, 92.05, 91.53, 91.62]. Assim a acurácia média da classificação foi de 91.96% com desvio padrão de 1.06%. Após o *k-fold*, o conjunto de validação foi utilizado no modelo para a predição em um grupo de dados não visto e foi obtida a matriz de confusão que pode ser analisada na Figura 3. Nela podemos observar que com exceção das faltas do tipo CT, o modelo tem pelo menos um erro em faltas que envolvem o terra. Isso mostra que existe uma dificuldade para o modelo distinguir faltas que possuem essa característica. Ao fim, é possível contar o número de acertos e erros da RF no conjunto de validação e verificar que ela teve 91.49% de acurácia.

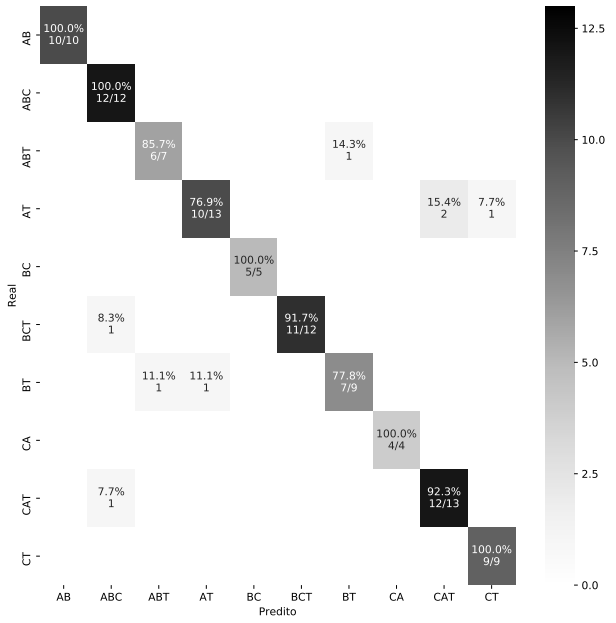

Figura 3. Matriz de confusão Florestas Aleatórias.

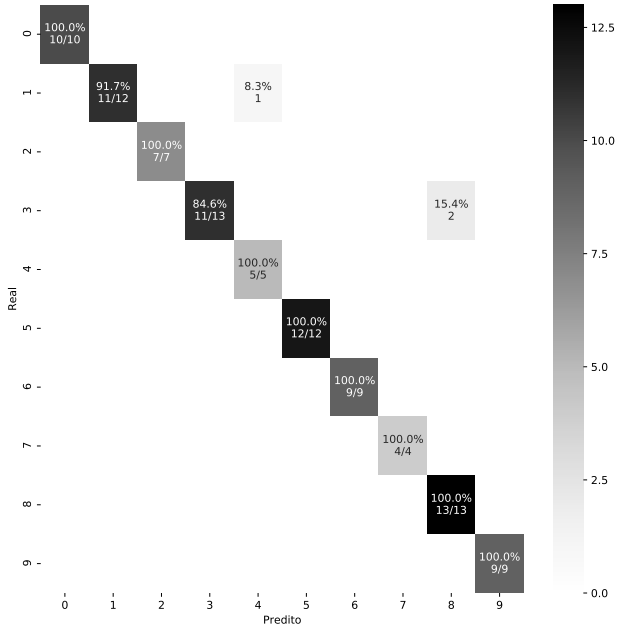

Figura 4. Matriz de confusão Rede Neural.

Já para o modelo de redes neurais artificiais obteve-se os seguintes valores para a acurácia: [91.78, 86.97, 88.52, 89.82, 90.26, 90.68, 89.93, 89.88, 89.02, 89.05]. Assim a acurácia média obtida pelo modelo foi de 89.59% com desvio padrão de 1.24%. O melhor modelo foi salvo durante o treinamento e utilizado posteriormente com o conjunto de testes na classificação de dados não vistos, e dessa maneira, foi obtida a matriz de confusão apresentada na Figura 4. Nela é possível observar que a acurácia obtida por esse modelo no conjunto de validação foi de 96.81%. Na Figura 4, os valores numéricos das linhas e colunas indicam as seguintes classes: AB = 0, ABC = 1, ABT = 2, AT = 3, BC = 4, BCT = 5, BT = 6, CA = 7, CAT = 8 e CT = 9. Portanto, nota-se uma dificuldade do modelo para predizer as faltas ABC e AT, em um maior nível do que

as demais. Cabe destacar que os resultados obtidos pelas matrizes de confusão não são constantes nessa etapa pois não foi realizado nenhum *loop* com conseguinte cálculo da média e desvio padrão. Dessa maneira, em cada execução do algoritmo uma nova divisão para o banco de dados é realizada o que leva à geração de matrizes de confusão distintas.

Além das análise apresentadas, também verificou-se o tempo necessário em operação para cada um dos modelos fornecer a classificação de uma única amostra. Assim, notou-se que o método usando *random forest* tem um tempo de processamento menor do que a rede neural. Nessa análise incluiu-se tanto o tempo necessário para tratamento dos sinais quanto predição dos modelos, sendo realizados 100 loops com 10 repetições. Observou-se que em geral o método usando *random forest* foi 8 vezes mais rápido do que usando rede neural. O melhor tempo obtido por cada modelo pode ser observado na Figura 5.

```
[13]  1 %%timeit -n100 -r10
      2 pipeline(V, fs, f0, fator_notch, rf_model)

      100 loops, best of 10: 4.11 ms per loop

[14]  1 %%timeit -n100 -r10
      2 pipeline(V, fs, f0, fator_notch, mlp_model)

      100 loops, best of 10: 33.6 ms per loop
```

Figura 5. Tempo de processamento dos modelos.

4. CONCLUSÕES

Nesse trabalho, utilizou-se os modelos de florestas aleatórias e redes neurais artificiais para classificação de faltas em linhas de transmissão. Tendo em vista que esse é um problema amplamente estudado na literatura, diversos autores já propuseram métodos para a realização dessa tarefa. Cabe ressaltar que os métodos utilizados no projeto proposto possuem componentes aleatórios tanto no ajuste manual dos hiperparâmetros, que foi realizado por tentativa e erro, quanto na adaptação dos parâmetros internos do modelo. Para a obtenção de resultados fidedignos busca-se eliminar essa aleatoriedade usando a repetição do laço de validação cruzada, nesse trabalho ele foi repetido 10 vezes, que pode ser considerado um número relativamente baixo. No entanto, um número maior de repetições não foi possível ser executado devido a configuração da rede neural utilizada e os recursos computacionais disponíveis, uma vez que mesmo com essa pequena quantidade de repetições o tempo de treinamento da rede foi elevado (superior a uma hora).

Após o treinamento dos modelos o conjunto de validação foi utilizado para gerar as matrizes de confusão e avaliar o desempenho final. A acurácia média obtida para o modelo de florestas aleatórias foi de 91.96% com desvio padrão de 1.06%, enquanto para a rede neural obteve-se uma média de 89.59% com 1.24% de desvio padrão. Em relação ao desempenho do modelo no conjunto de validação, obteve-se acurácia de 91.49% para o modelo RF e 96.81% para a ANN. Os resultado obtidos foram comparados com outros projetos observados na literatura como pode ser analisado na Tabela 3.

Tabela 3. Comparação entre os resultados obtidos e alguns projetos de outros autores.

Trabalho	**Pré-processamento**	**Método**	**Acurácia**
R. de Carvalho et al. (2014)	HOS	Redes Neurais	99%
Almeida et al. (2017)	ICA	SVM	100%
Proposto	Filtro *notch*	Redes Neurais	96,81%
Proposto	Filtro *notch*	Florestas aleatórias	91,49%

Dessa forma, comparando-se os resultados com trabalhos realizados anteriormente como em R. de Carvalho et al. (2014) e Almeida et al. (2017), o projeto aqui proposto possui uma etapa de pré-processamento computacionalmente mais eficiente tendo em vista que utilizou-se o filtro *notch* que possui menor custo computacional do que estatística de ordem superior e análise de componentes independentes que foram utilizados nos trabalhos citados. No entanto, visando a melhoria da acurácia na classificação desses dados, futuramente pretende-se realizar um comparativo entre o uso do filtro *notch* com outras técnicas no pré-processamento dos sinais. Além disso, é interessante treinar e testar outros modelos de inteligência computacional e comparar os resultados com o desempenho obtido pelas técnicas de redes neurais artificiais e florestas aleatórias propostas nesse trabalho.

Por fim, é importante destacar que os dados utilizados nesse trabalho foram gerados pelo modelo proposto por Costa et al. (2010), e que em geral os autores não disponibilizam suas fontes de dados, ou como reproduzi-los o que dificulta a comparação direta com trabalhos disponíveis na literatura. Ainda deve-se ressaltar que os métodos de inteligência computacional, podem se beneficiar da paralelização, uso de múltiplos processadores ou GPUs (*Graphics Processing Unit*), dessa maneira, o tempo de treinamento encontrado por outros trabalhos de pesquisa na literatura podem diferir dos que foram obtidos nesse trabalho. Além disso, essa não é uma informação comumente apresentada de maneira direta ou simplificada por artigos da área. No entanto, no contexto do trabalho proposto os resultados foram satisfatórios principalmente em relação ao tempo de treinamento do modelo de florestas aleatórias.

REFERÊNCIAS

Almeida, A.R., Almeida, O.M., Junior, B.F., Barreto, L.H., and Barros, A.K. (2017). ICA feature extraction for the location and classification of faults in high-voltage transmission lines. doi:10.1016/j.epsr.2017.03.030.

Avagaddi, P., Edward, B., and Ravi, K. (2017). A review on fault classification methodologies in power transmission systems: Part i. *Journal of Electrical Systems and Information Technology*, 5. doi:10.1016/j.jesit.2017.01.004.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. doi:10.1023/A:1010933404324. URL `https://doi.org/10.1023/A:1010933404324`.

Costa, F., Souza, B., and Brito, N. (2010). Real-time detection of fault-induced transients in transmission lines. *Electronics Letters*, 46, 753 – 755. doi:10.1049/el.2010.0812.

Das, D., Singh, N.K., and Sinha, A.K. (2005). A comparison of fourier transform and wavelet transform methods for detection and classification of faults on transmission

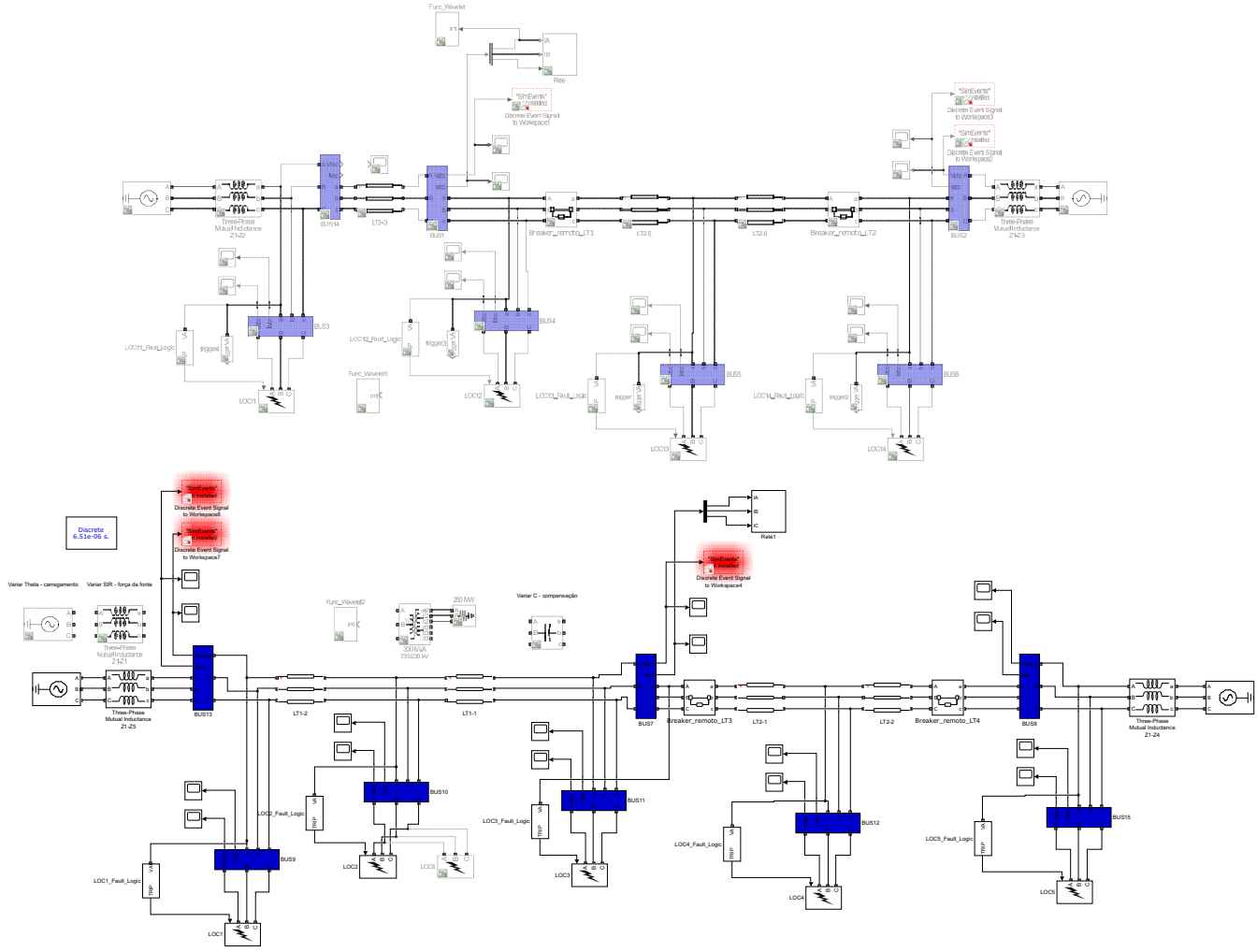

Figura 6. Diagrama do sistema elétrico analisado nesse projeto.