XXI CONGRESSO DE INICIAÇÃO CIENTÍFICA DA UNIVERSIDADE FEDERAL DE CAMPINA GRANDE

# *APLICAÇÃO DE TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL NA DETECÇÃO DE DEFEITOS EM PARA-RAIOS ATRAVÉS DA ANÁLISE DE TERMOGRAFIAS.*

**Hebert Santos Crispim[1], Pablo Bezerra Vilar[2]**

## RESUMO

Os para-raios de óxido de zinco (ZnO) são essenciais para a proteção de sistemas elétricos contra sobretensões, garantindo a continuidade e confiabilidade das operações. No entanto, é crucial que esses equipamentos estejam sempre em boas condições de funcionamento. Diante disso, há uma crescente demanda por técnicas eficientes de detecção de defeitos, especialmente métodos não invasivos, como a análise de temperatura externa. A termografia se destaca como uma ferramenta poderosa, capaz de identificar variações de temperatura que podem indicar falhas nos para-raios. Este trabalho foca na implementação de redes neurais convolucionais para a classificação de defeitos em para-raios a partir de imagens térmicas. Foram analisadas diferentes arquiteturas de redes neurais, incluindo CNN clássica, Inception, Xception, VGG16 e VGG19, com o objetivo de extrair características das imagens termográficas e identificar para-raios bons e defeituosos. Observou-se que o desempenho das redes é influenciado pela complexidade da arquitetura e pelo tamanho restrito da base de dados, sendo que algumas arquiteturas mais avançadas apresentaram sinais de overfitting. O projeto busca aprimorar essas redes para um melhor desempenho na classificação de defeitos.

**Palavras-chave**: Inteligência Artificial; Redes Neurais Convolucionais; Para-raios; Termografia; Detecção de Defeitos; Sobretensões.

[1]Aluno do curso de Engenharia Elétrica, Departamento de Engenharia Elétrica, UFCG, Campina Grande, PB, e-mail: hebert.crispim@ee.ufcg.edu.br
[2]<Titulação>, <Função>, <Departamento>, UFCG, Campina Grande, PB, e-mail: emaildoorientador@seuprovedor.com

***APPLICATION OF ARTIFICIAL INTELLIGENCE TECHNIQUES IN LIGHTNING ARRESTER DEFECT DETECTION THROUGH THERMOGRAPHY ANALYSIS.***


## ABSTRACT

Zinc oxide (ZnO) surge arresters are essential for protecting electrical systems from overvoltages, ensuring operational continuity and reliability. However, it is crucial that these devices are always in good working condition. In this context, there is a growing need for efficient fault detection techniques, especially non-invasive methods such as external temperature analysis. Thermography stands out as a powerful tool, capable of identifying temperature variations that may indicate faults in surge arresters. This work focuses on implementing convolutional neural networks (CNNs) to classify defects in surge arresters based on thermal images. Various neural network architectures were analyzed, including classic CNN, Inception, Xception, VGG16, and VGG19, with the objective of extracting features from thermal images to identify both healthy and faulty surge arresters. It was observed that the performance of the networks is influenced by the complexity of the architecture and the limited size of the dataset, with some more advanced architectures showing signs of overfitting. The project aims to improve these networks for better defect classification performance.



**Keywords**: Artificial Intelligence; Convolutional Neural Networks; Surge Arresters; Thermography; Defect Detection; Overvoltages.

# 1. INTRODUÇÃO

Os para-raios de óxido de zinco (ZnO) são dispositivos utilizados para suprimir surtos e são empregados na proteção de equipamentos presentes no sistema elétrico de potência contra sobretensões originadas tanto externamente (por exemplo, descargas atmosféricas) quanto internamente (por exemplo, sobretensões de manobra). Sua função é controlar o nível de tensão que alcançaria equipamentos críticos, como os transformadores de potência, prevenindo que sejam expostos a níveis de tensão inadequados para seu funcionamento, e assim evitem a retirada de operação deste equipamento. (LIRA, 2012).

A evolução contínua da tecnologia de para-raios tem desempenhado um papel fundamental na salvaguarda de vidas humanas e de infraestruturas elétricas contra os efeitos adversos das descargas atmosféricas. A necessidade de proteger sistemas elétricos contra sobretensões tem impulsionado o aprimoramento dos dispositivos de proteção. Entre esses dispositivos, os para-raios fabricados com óxido de zinco emergem como uma escolha eficaz devido às suas propriedades elétricas superiores e habilidade de dissipação de energia.

Contudo, a eficácia dos para-raios pode ser comprometida por uma variedade de fatores, incluindo falhas intrínsecas ao próprio dispositivo como a perda de estanqueidade; penetração de umidade; descargas parciais internas; dentre outras. Diante dessa realidade, a implementação de técnicas avançadas de diagnóstico torna-se essencial para identificar potenciais falhas e garantir o desempenho adequado dos para-raios de óxido de zinco. Uma dessas técnicas, a termografia, surge como uma ferramenta para avaliar o funcionamento dos para-raios, possibilitando a detecção precoce de anomalias térmicas que indiquem falhas ou irregularidades.

Adicionalmente, o avanço da tecnologia de aprendizado de máquina oferece oportunidades significativas para aprimorar a eficácia e a automatização do processo de diagnóstico de para-raios. A utilização de redes neurais convolucionais (CNNs) tem se mostrado particularmente promissora nesse contexto, permitindo a classificação precisa das imagens térmicas obtidas por meio da termografia. Por meio da análise de padrões complexos e sutis nas imagens, as CNNs são capazes de distinguir entre para-raios em condições normais de operação e dispositivos com

falhas, contribuindo assim para a manutenção preventiva e a segurança do sistema elétrico.

Para enriquecer ainda mais o conjunto de dados e aprimorar a eficácia das CNNs, são conduzidas simulações computacionais utilizando o software COMSOL *Multiphysics*. Essas simulações permitem obter informações adicionais sobre o comportamento térmico dos para-raios de óxido de zinco em diferentes condições operacionais e cenários de falha, proporcionando uma otimização dos modelos de diagnóstico.

Portanto, este estudo adota uma abordagem integrada, combinando técnicas de termografia, aprendizado de máquina e simulação computacional, para fornecer uma avaliação abrangente e precisa do funcionamento dos para-raios de óxido de zinco. Ao contribuir para o aprimoramento da segurança e confiabilidade das redes elétricas, espera-se que esta pesquisa beneficie significativamente a infraestrutura elétrica e a sociedade como um todo.

## 2. MATERIAIS E MÉTODOS (OU METODOLOGIA)

O procedimento de análise adotado no projeto consiste em uma revisão detalhada da literatura que permitirá a compreensão das metodologias adotadas em pesquisas anteriores, identificar lacunas no conhecimento existente e destacar contribuições significativas para o desenvolvimento de Inteligências Artificiais e em simulações de Para-raios.

Primeiramente, a escolha do *software* COMSOL advém de sua capacidade de lidar associar diversos equacionamentos que permitem simular a interação de diferentes fenômenos físicos, como eletrostática, corrente elétrica, calor e radiação térmica, no caso do para-raios, o COMSOL deverá permitir a simulação de comportamentos térmicos a partir de excitação elétrica, espera-se assim que seja possível expandir as bases de dados de imagens térmicas disponíveis num ritmo mais acelerado do que seria possível com medições diretas. O COMSOL permite a modelagem integrada desses fenômenos, proporcionando uma representação mais precisa do sistema. Adicionalmente, o COMSOL oferece uma ampla gama de opções de modelagem e de soluções, permitindo personalizar suas simulações de acordo com as necessidades específicas.

A abordagem de Inteligência Artificial empregada foi a das Redes Neurais Convolucionais. Este método é amplamente reconhecido na literatura como um dos mais eficazes e de maior desempenho na tarefa de classificação de imagens. As Redes Neurais Convolucionais (CNN), parte do campo de *Deep Learning* (Aprendizado Profundo), operam ao receber uma imagem como entrada e, por meio de convoluções, atribuem pesos e identificam características que permitem distingui-la de outras imagens.

Foram utilizadas diferentes arquiteturas de CNN para a tarefa de classificação, incluindo Xception, Inception, VGG16 e VGG19, além de um modelo geral de CNN. Posteriormente, os resultados obtidos por cada uma dessas arquiteturas foram comparados, com o intuito de avaliar o desempenho em termos de precisão e outras métricas relevantes. Essa comparação permite não apenas identificar a arquitetura mais adequada para a tarefa específica, mas também fornecer os pontos fortes e limitações de cada abordagem.

Essas métricas são essenciais para entender e melhorar o desempenho de uma CNN em tarefas de classificação de imagens. Elas fornecem informações sobre a capacidade do modelo de fazer previsões precisas e ajudam a identificar áreas onde o modelo pode precisar de ajustes ou melhorias para alcançar melhores resultados.

## 2.1. MÉTRICAS UTILIZADAS

De forma a analisar o desempenho das redes neurais, a acurácia, a curva ROC, as perdas e a matriz de confusão são conceitos fundamentais ao avaliar o desempenho de CNNs, especialmente em tarefas de classificação de imagens, que é o objeto de estudo.

## 2.2. ACURÁCIA E PERDA

A acurácia é uma métrica amplamente utilizada para medir a precisão global do modelo, representando a proporção de previsões corretas em relação ao total de previsões feitas. Essa é calculada dividindo o número de previsões corretas pelo número total de previsões e multiplicando por 100%.

Por outro lado, as perdas, ou *loss*, refletem a discrepância entre as previsões do modelo e os valores reais dos dados. Durante o treinamento da CNN, o objetivo é minimizar essas perdas ajustando os parâmetros do modelo. Por fim, a matriz de

confusão é uma ferramenta valiosa para avaliar o desempenho do modelo em um conjunto de dados de teste. Ela organiza as previsões do modelo em uma tabela, onde cada linha representa as instâncias em uma classe prevista e cada coluna representa as instâncias em uma classe real (ou vice-versa).

O gráfico de acurácia mostra a evolução da precisão do modelo ao longo do treinamento, exibindo a taxa de acertos do modelo em relação ao total de exemplos de treinamento. À medida que o treinamento avança, a acurácia pode aumentar, indicando que o modelo está aprendendo com sucesso padrões nos dados de treinamento. No entanto, é importante observar se há sinais de *overfitting*, onde a acurácia nos dados de treinamento continua a aumentar, mas a acurácia nos dados de validação começa a diminuir, o que sugere que o modelo está se tornando muito específico para os dados de treinamento e não está generalizando bem para novos dados (indicação de escassez de imagens utilizadas para o treinamento do modelo).

Por outro lado, o gráfico de perda mostra como a perda do modelo diminui ao longo do treinamento. A perda é uma medida do quão bem o modelo está se ajustando aos dados de treinamento. Idealmente, à medida que o treinamento avança, a perda deve diminuir, indicando que o modelo está melhorando sua capacidade de fazer previsões precisas. Assim como com a acurácia, é importante monitorar a perda nos dados de validação para evitar o *overfitting*. Se a perda nos dados de validação começar a aumentar enquanto a perda nos dados de treinamento continua a diminuir, pode ser um sinal de que o modelo está se ajustando demais aos dados de treinamento e não está generalizando bem.

Embora seja uma métrica simples e frequentemente utilizada, ela pode ser insuficiente, especialmente em problemas de classes desbalanceadas. Em situações onde uma classe é muito mais frequente do que as outras, o modelo pode obter uma alta acurácia ao prever sempre a classe majoritária, mas sem capturar corretamente as características das classes minoritárias. Portanto, a acurácia sozinha não reflete o verdadeiro desempenho do modelo nesses casos, e métricas adicionais são necessárias para uma análise mais robusta

### 2.3. MATRIZ DE CONFUSÃO

A matriz de confusão é gerada com os dados de métricas obtidas do modelo criado. A matriz é organizada em linhas e colunas, onde as linhas representam as

classes reais dos dados e as colunas representam as classes previstas pelo modelo. Cada célula da matriz contém o número de exemplos que foram classificados de acordo com a interseção entre a classe real e a classe prevista.

A matriz de confusão oferece uma visão mais detalhada do desempenho do modelo, pois indica a quantidade de verdadeiros positivos (TP), falsos positivos (FP), verdadeiros negativos (TN) e falsos negativos (FN) para cada classe, como ilustrado na tabela 01. Isso permite uma análise mais precisa do comportamento do modelo em cada categoria, especialmente útil para avaliar erros de falsos positivos e falsos negativos. Dessa forma, a matriz de confusão é crucial em problemas onde o impacto de um erro pode variar, como em diagnósticos.

**Tabela 1. Quatro Resultados da Matriz de Confusão.**

| | **FALSO** | **VERDADEIRO** |
|---|---|---|
| **FALSO** | Negativo Verdadeiro (TN) | Positivo Falso (FP) |
| **VERDADEIRO** | Negativo Falso (FN) | Positivo Verdadeiro (TP) |

Fonte: Autoria Própria

## 2.4. CURVA ROC

Por fim, a curva ROC (*Receiver Operating Characteristic*) é uma representação gráfica da performance de um modelo de classificação binária, que mostra a relação entre a taxa de verdadeiros positivos (TPR) e a taxa de falsos positivos (FPR) em diferentes limiares de classificação. Essa curva é usada para avaliar a capacidade do modelo em distinguir, e a área sob a curva (AUC) serve como uma métrica para quantificar o desempenho do modelo quanto mais próxima a AUC estiver de 1, melhor o modelo está discriminando entre as classes, esse comportamento pode ser avaliado na figura 01. A análise da curva ROC é especialmente útil em contextos onde as classes estão desbalanceadas, permitindo uma avaliação mais precisa da qualidade do modelo.

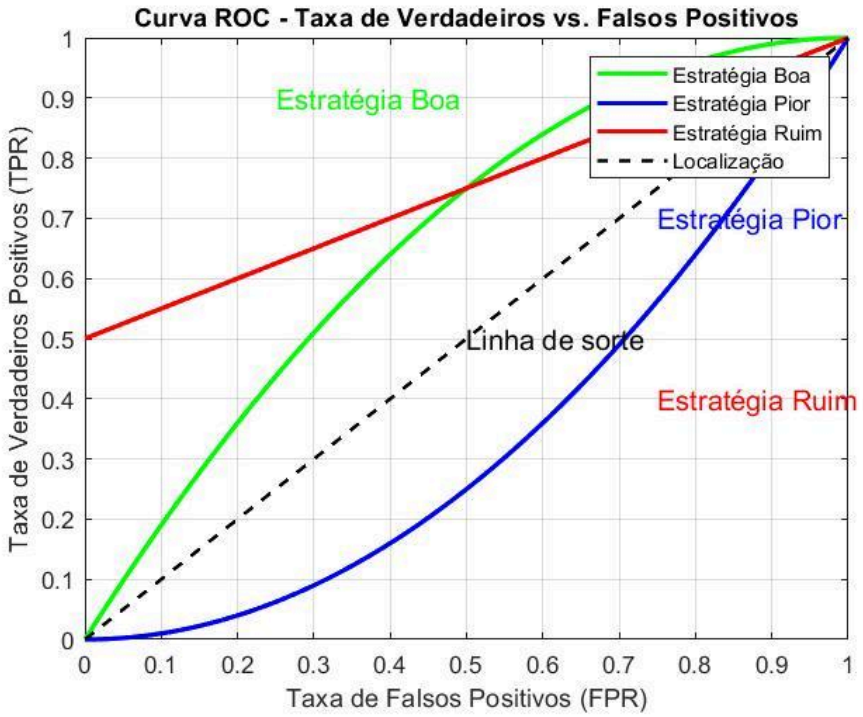



Figura 01- Autoria Própria

## 3. DESENVOLVIMENTO

### 3.1. TRATAMENTO DA BASE DE DADOS

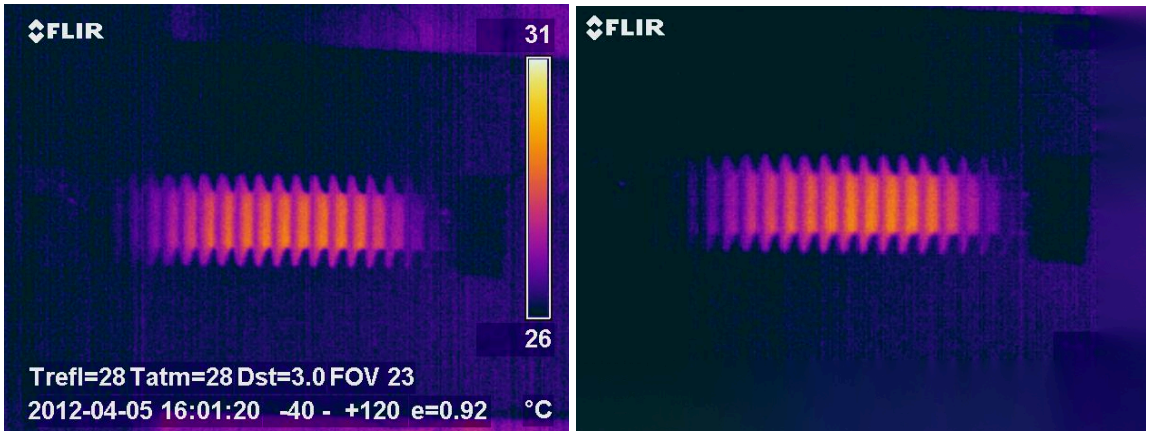
O presente trabalho busca melhorar o desempenho de sistemas de classificação de imagens térmicas de para-raios, neste contexto, o desafio inicial é a inexistência de grandes e amplas bases de dados que seriam necessárias para a formulação de tais modelos. Inicialmente serão utilizadas bases de dados produzidas no Laboratório de Alta Tensão da UFCG, contudo, tal base de dados ainda terá um volume bastante restrito, em função da dificuldade e dispêndio de tempo necessário para realização dos experimentos.

Neste contexto, um dos principais desafios associados à produção de um modelo inteligente confiável a partir de bases de dados restrita é conhecido como "*overfitting*". Entende-se que o *overfitting*, ou "sobre ajuste" em português, é um fenômeno comum em aprendizado de máquina e ocorre quando um modelo é excessivamente ajustado aos dados de treinamento, captando não apenas os padrões verdadeiros nos dados, mas também as características específicas dos dados de treinamento que não são generalizáveis.

Em síntese, um modelo com overfitting se adapta adequadamente durante seu treinamento, mas tem um desempenho ruim ao ser exposto a novos dados. Isso

acontece porque o modelo se torna especializado em memorizar os exemplos específicos do conjunto de treinamento, não aprendendo os padrões gerais que são realmente úteis para fazer previsões em dados novos. Uma abordagem comum para evitar esse problema é a separação do conjunto de dados em 70% para treinamento e 30% para teste. No entanto, essa divisão pode não ser suficiente, principalmente em função de sucessivas tentativas de treinamento, que podem influenciar o modelo a se adaptar ao conjunto de testes. Para mitigar o overfitting, além de uma divisão adequada, técnicas como regularização (L1, L2), *early stopping*, *data augmentation* e validação cruzada (*cross-validation*) são frequentemente utilizadas.

Com o intuito de reduzir a probabilidade de ocorrer esse tipo de fenômeno ("*overfitting*"), se fez necessário o tratamento da base de dados, através da programação em *python,* com o intuito de padronizar as imagens da base de dados. Neste contexto, a primeira medida adotada foi realizar cortes em letras e palavras, deixando apenas a imagem termográfica do para-raio, como apresentado na figura 02.




(a) Imagem Original. (b) Imagem com Tratamento.

Figura 02- Autoria Própria.

### 3.2. DESENVOLVIMENTO DE DATA ARGUMENTATION

Ainda visando evitar o *overfitting*, nos modelos em desenvolvimento nesta pesquisa, foi implementado um algoritmo para a ampliação de dados. Para a implementação do modelo foi usada a linguagem de programação Python, assim como, as seguintes bibliotecas*: numpy, pandas, matplotlib, keras* e *tensorflow*. Onde são configurados os parâmetros do `*ImageDataGenerator*`, função da biblioteca

*keras,* que controlam as transformações a serem aplicadas às imagens durante o processo de aumento. Essas transformações incluem rotação, deslocamento horizontal e vertical, cisalhamento, zoom e espelhamento horizontal. Essa variedade de transformações permite criar uma ampla gama de variações das imagens originais, o que enriquece o conjunto de dados de treinamento. Este tipo de expansão da base de dados é usualmente referida como *Data Augmentation*.

A base de dados que foi utilizada a função, foi desenvolvida por pesquisadores do Laboratório de Alta Tensão (LAT), inicialmente a base continha 104 imagens termográficas para os para-raios bons, como também, 104 imagens termográficas para os para-raios defeituosos, como apresentado na figura 03.

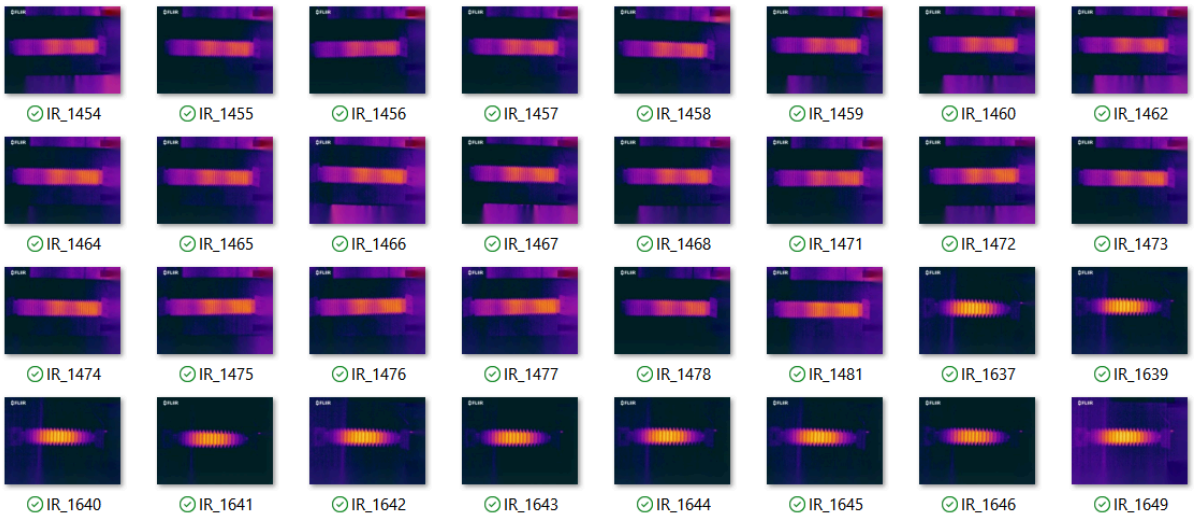



Figura 03- Fração das Imagens Iniciais.

Aplicou-se uma série de transformações às imagens originais, como rotação, deslocamento, corte, zoom, entre outras, resultando em um novo conjunto de dados que contém agora 336 imagens cada, como apresentado na figura 03.

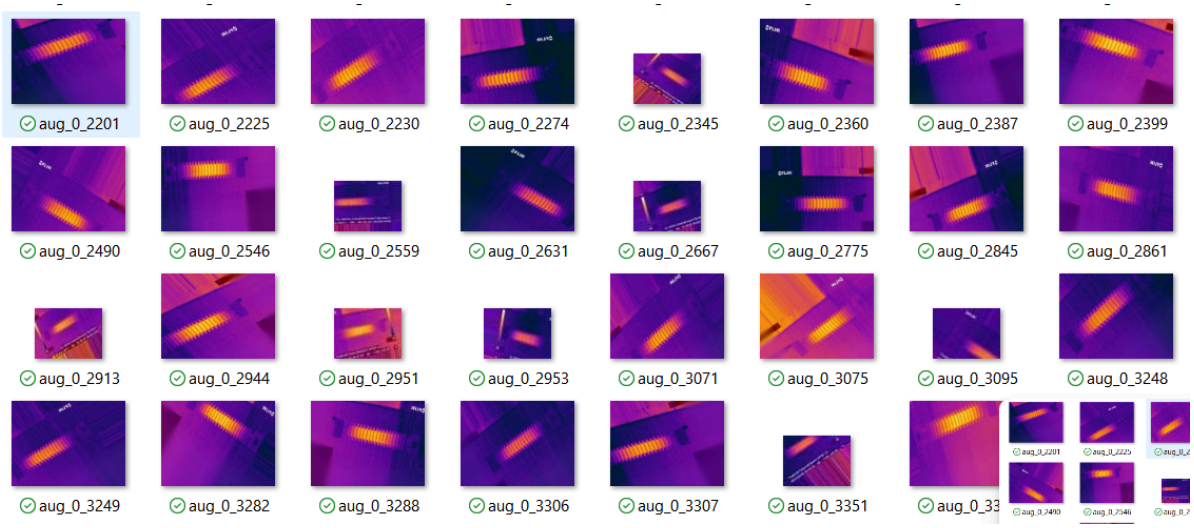



Figura 04- Fração das Imagens Geradas com o *ImageDataGenerator.*

Ao analisar os elementos apresentados nas Figuras 01 e 02 sob uma perspectiva humana, pode-se imaginar que tratam-se das mesmas imagens e por tanto, não estaria sendo agregada informação para evitar *overfitting*, contudo, podemos observar que, por exemplo, agora o classificador tem uma indicação clara de que a inclinação do para-raios na imagem não deve ser considerada no diagnóstico, por tanto, de um ponto de vista matemático, a base de dados tornou-se mais ampla.

## 4. RESULTADOS E DISCUSSÕES

### 4.1. DESENVOLVIMENTO DE CLASSIFICADORES INTELIGENTES

As Redes Neurais Convolucionais (CNNs) são uma técnica poderosa de aprendizado de máquina amplamente utilizada para tarefas de classificação de imagens, detecção de objetos e reconhecimento de padrões. Inspiradas no funcionamento dos neurônios biológicos, as CNNs se destacam por sua capacidade de aprender automaticamente características importantes das imagens. Elas fazem isso utilizando camadas convolucionais que detectam atributos visuais, começando com características simples, como bordas, e avançando para padrões mais complexos nas camadas subsequentes.

Com isso, o primeiro modelo de CNN desenvolvido teve o objetivo de identificar a presença de defeito no para-raio (atribuindo-lhe valor 0) ou identificar um para-raios sem defeitos (atribuindo-lhe valor 1) .

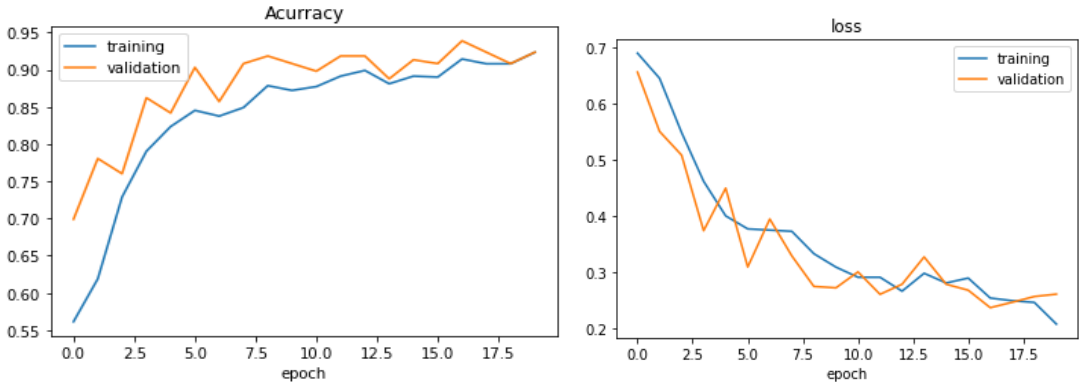
A criação do modelo começa com a API Sequencial do Keras, que permite adicionar camadas de forma sequencial. Ela começa com uma camada convolucional 2D (‘Conv2D’) com 32 filtros e um tamanho de kernel de (5, 5). Esta camada processa a imagem de entrada, aplicando a operação de convolução para extrair características. Em seguida, é adicionada uma camada de *max-pooling* (‘*MaxPooling2D*’) com um tamanho de pool de (2, 2). Essa camada reduz a dimensionalidade da imagem, preservando as características mais importantes. Depois, é adicionada outra camada convolucional 2D com 64 filtros e um tamanho de kernel de (3, 3), seguida por outra camada de *max-pooling*. Por fim, as camadas são achatadas (‘*Flatten*’) para converter a saída das camadas convolucionais em um vetor unidimensional, e duas camadas totalmente conectadas (`*Dense*`) são adicionadas. A função de ativação ReLU (`*activation*='relu'`) é usada em todas as camadas, exceto na camada de saída, todos os passos para a criação do modelo podem ser vistos na figura 05.

```
# Criar modelo
def myModel():
    model = Sequential()
    model.add(Conv2D(32, kernel_size=(5, 5), input_shape=(imageDimesions[0], imageDimesions
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Dropout(0.5))

    model.add(Flatten())
    model.add(Dense(128, activation='relu'))
    model.add(Dense(noOfClasses, activation='softmax'))

    # Compilar modelo
    model.compile(Adam(lr=0.001), loss='categorical_crossentropy', metrics=['accuracy'])
    return model
```

Figura 05 – Fração do Código para Implementação do Modelo de Treinamento.

Em resumo, a função utilizada para a criação da CNN cria uma arquitetura com várias camadas convolucionais, usando a ‘*API Sequencial*’ da biblioteca *Keras*. Essa arquitetura é adequada para classificação de imagens e é compilada com configurações apropriadas para treinamento, tratamento e avaliação.

As figuras 6a e 6b a seguir, apresentam os gráficos de acurácia e de perdas do modelo para os dados de treino e de teste. A figura 07 a seguir apresenta a matriz de confusão para o treinamento do modelo.




(a) Gráfico da Acurácia do Modelo (b) Gráfico de Perdas do Modelo

Figura 06 – Autoria Própria.

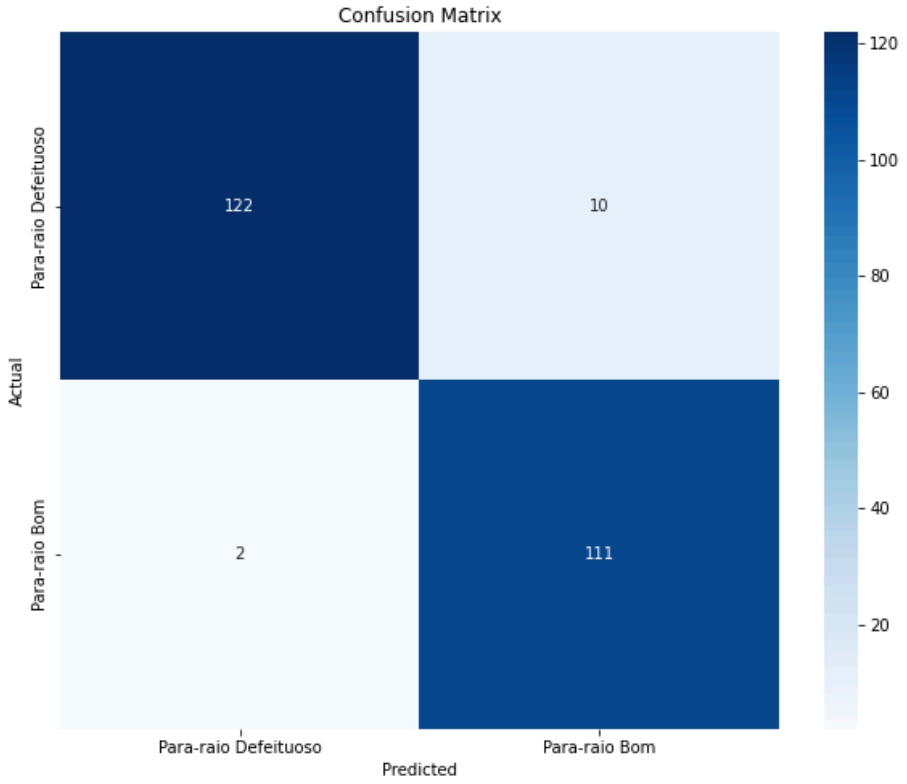



Figura 07– Matriz de Confusão do Modelo.

As figuras apresentadas acima demonstram que os resultados obtidos foram satisfatórios, onde como panorama geral de métricas do modelo, foi obtida uma taxa de acertos dentre as previsões possíveis: 0.9357 ou 93.57%.

### *4.2. INCEPTION*

A *InceptionV3* é uma evolução significativa no campo das CNNs, desenvolvida para lidar com a complexidade e a diversidade dos padrões em

imagens. Ela é caracterizada pelo uso de módulos *Inception*, que realizam convoluções com diferentes tamanhos de filtro em paralelo, permitindo que a rede capture padrões em várias escalas simultaneamente. Isso é especialmente útil em cenários onde as características das imagens variam em tamanho e complexidade.

No desenvolvimento do modelo baseado na arquitetura InceptionV3, foi utilizado um modelo pré-treinado. Um modelo pré-treinado é uma rede neural que já foi treinada em um grande conjunto de dados, como o *ImageNet*, contendo milhões de imagens categorizadas. Esses modelos já possuem pesos ajustados que capturam padrões visuais gerais, como texturas e formas, o que acelera o processo de desenvolvimento, pois a rede já tem um entendimento inicial dos dados. O modelo pré-treinado foi carregado sem as camadas superiores para permitir a personalização para a tarefa específica. Novas camadas densas foram adicionadas, seguidas por uma camada de saída com função de ativação sigmóide para a classificação binária. Inicialmente, as camadas base foram congeladas, e posteriormente foi aplicado o *fine-tuning*, descongelando algumas das camadas finais para um ajuste mais refinado.

Este modelo foi compilado utilizando o otimizador Adam, com uma taxa de aprendizado que foi ajustada entre as fases de treinamento inicial e *fine-tuning* para garantir uma convergência eficaz. Além disso, foram utilizados *callbacks* como *Early Stopping* para monitorar o desempenho do modelo e evitar o overfitting. Durante a avaliação, o modelo gerou previsões sobre o conjunto de validação, as quais foram comparadas com os rótulos verdadeiros para gerar uma matriz de confusão e calcular a acurácia, como ilustrados nas figuras 08 e 09.

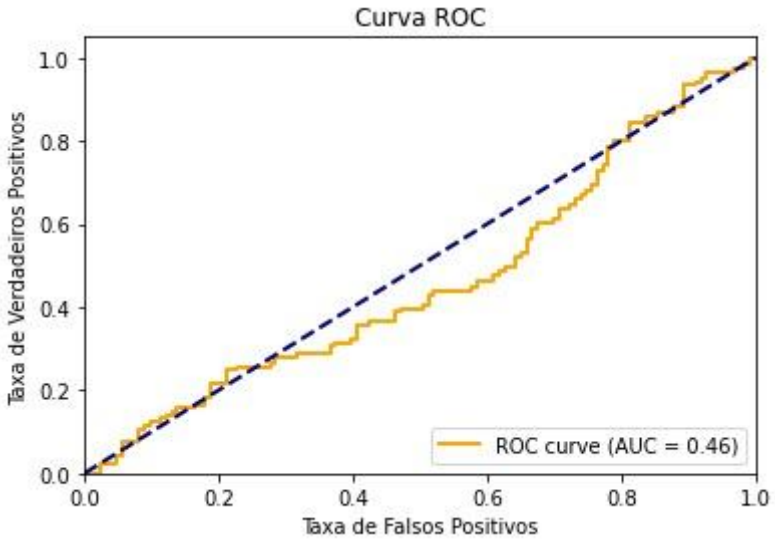



Figura 08– Curva ROC para o Modelo.

```
Avaliação no conjunto de validação:
8/8 ━━━━━━━━━━━━━━━━━━━━ 14s 2s/step - accuracy: 0.7781 - loss: 0.3372
Loss: 0.3454, Accuracy: 0.7828

Gerando previsões no conjunto de validação...
8/8 ━━━━━━━━━━━━━━━━━━━━ 20s 2s/step

Matriz de Confusão:
[[84 39]
 [86 35]]
```

Figura 09– Matriz de Confusão do Modelo.

### *4.3. XCEPTION*

A Xception é uma arquitetura que representa uma extensão da Inception, propondo uma abordagem mais eficiente através do uso de convoluções separáveis em profundidade. O modelo Xception foi projetado para capturar características complexas de maneira mais eficaz, dividindo o processo de convolução em duas partes: a convolução espacial e a convolução de profundidade. Isso permite um modelo mais leve e rápido, sem comprometer a precisão.

O desenvolvimento de um classificador utilizando a arquitetura Xception seguiu uma abordagem semelhante às anteriores. Primeiro, os diretórios de imagens para as classes "defeituoso" e "bom" foram combinados para criar um único conjunto

de dados de treinamento e validação. Em seguida, o modelo Xception pré-treinado foi carregado, sem as camadas superiores, permitindo a adição de novas camadas densas e uma camada de saída *sigmóide* para a classificação binária. O treinamento foi realizado com técnicas de *data argumentation* e *callbacks*, incluindo *Early Stopping* para evitar o overfitting.

A avaliação do modelo foi conduzida tanto no conjunto de validação quanto no de teste, com as métricas de acurácia e perda sendo calculadas. Além disso, foram geradas previsões no conjunto de teste, as quais foram usadas para criar um relatório de classificação e uma matriz de confusão, visualizada através de um heatmap. Para completar a análise de desempenho, foi gerada uma Curva ROC, ilustrada na figura 10, que permite avaliar a capacidade do modelo em discriminar entre as duas classes, resultando em um AUC (Área Sob a Curva) que fornece uma medida global da performance do modelo.

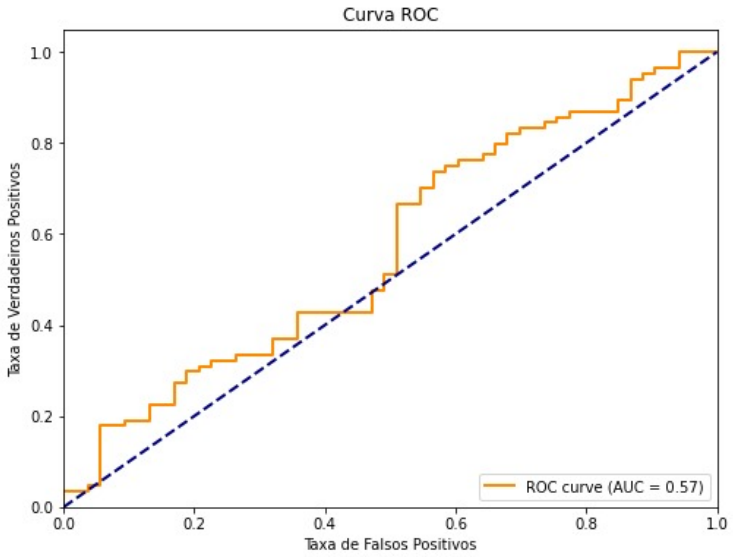



Figura 10– Curva ROC para o Modelo.

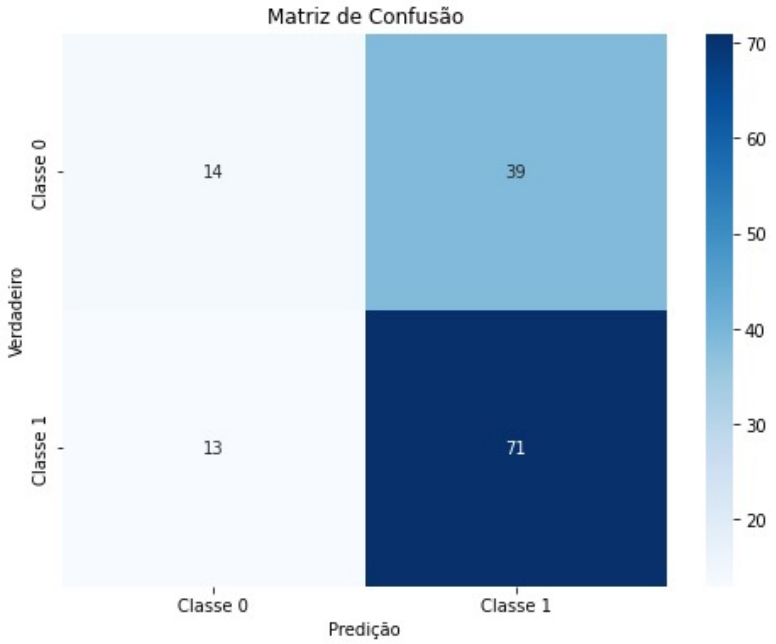


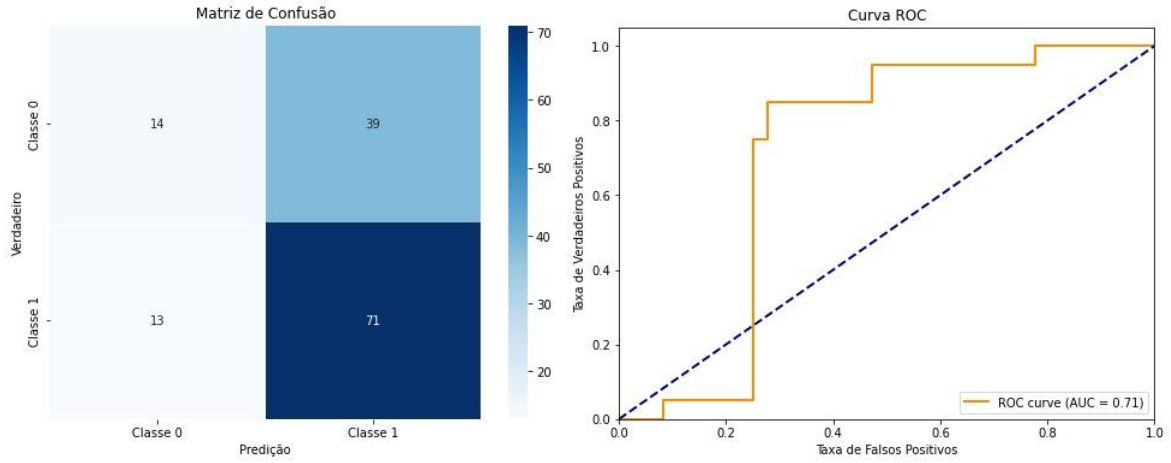

Figura 11– Matriz de Confusão do Modelo.

### *4.4. VGG16 E VGG19*

As arquiteturas VGG16 e VGG19, desenvolvidas pelo *Visual Geometry Group* (VGG) da Universidade de Oxford, são conhecidas por sua simplicidade e profundidade. Ambas utilizam camadas convolucionais pequenas (3x3) empilhadas sequencialmente, seguidas por camadas de pooling e, ao final, camadas totalmente conectadas. A principal diferença entre VGG16 e VGG19 é o número de camadas ponderadas: a VGG16 possui 16 camadas (13 convolucionais e 3 totalmente conectadas), enquanto a VGG19 tem 19 camadas (16 convolucionais e 3 totalmente conectadas).

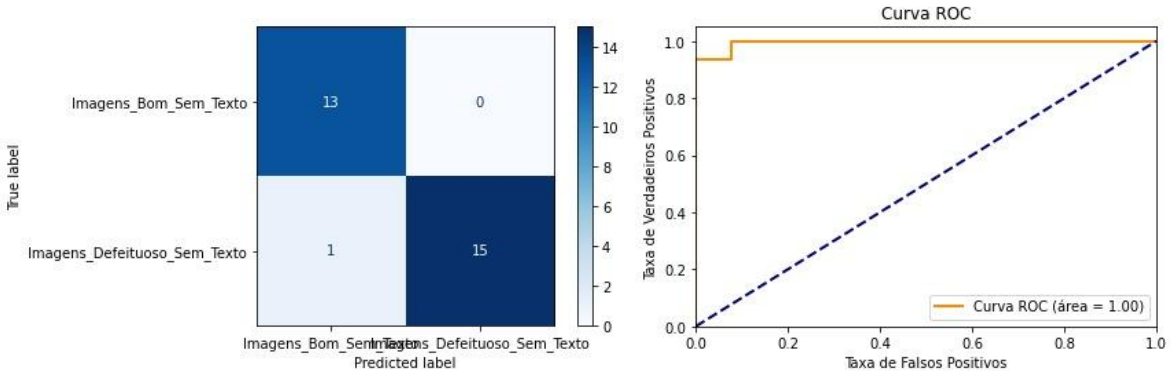
Para desenvolver classificadores baseados nas arquiteturas VGG16 e VGG19, o processo começou com o carregamento dos modelos pré-treinados na *ImageNet*, excluindo suas camadas superiores para permitir a adição de novas camadas personalizadas. As camadas do modelo base foram congeladas para evitar o ajuste dos pesos durante o treinamento inicial. Novas camadas densas foram adicionadas para adaptar o modelo à tarefa de classificação binária de imagens, que foi realizada utilizando um gerador de dados que aplica técnicas de data augmentation para melhorar a robustez do modelo. Após o treinamento inicial, foi realizado o fine-tuning, descongelando algumas camadas do modelo base para um

ajuste mais fino com uma taxa de aprendizado menor e consegue-se obter os seguintes parâmetros para os modelos, nas figuras 12 e 13.




(a) Matriz de Confusão do Modelo (b) Gráfico de ROC do Modelo.

Figura 12– Modelo VGG16.




(a) Matriz de Confusão do Modelo (b) Gráfico de ROC do Modelo.

Figura 13– Modelo VGG19.

## 4.5. COMPARAÇÃO ENTRE MODELOS

**Tabela 2. Comparação Para Os Modelos Implementados.**

| Característica | VGG16 | VGG19 | XCEPTION | INCEPTION | CNN |
|---|---|---|---|---|---|
| **Arquitetura Geral** | 16 camadas de pesos treináveis | 19 camadas de pesos treináveis | Separação Convolucional e Profundidade *Depthwise* | Módulos Inception com convoluções 1x1, 3x3 e 5x5 em paralelo | Convoluções seguidas de pooling, geralmente com poucas camadas |
| **Número Total de Camadas** | 13 convoluções + 3 FC = 16 | 16 convoluções + 3 FC = 19 | 36 camadas (de convolução, pooling e outras) | 42 camadas profundas | Variável, geralmente 3-10 camadas |
| **Pontos Fortes** | Simplicidade e eficácia em *transfer learning* | Melhor detalhamento devido à profundidade adicional | Alta eficiência e desempenho com menos parâmetros | Altamente eficiente em extração de características com diferentes escalas | Fácil de entender e implementar, boa para problemas simples |
| **Desvantagens** | Modelo grande, caro em termos de armazenamento e inferência | Mais pesado que o VGG16 sem grande ganho de performance | Mais complexo de entender e implementar | Arquitetura mais complicada de implementar e entender | Pouca eficiência para problemas complexos e grande volume de dados |

Fonte: Autoria Própria

## 4.6. ESTUDO DO AQUECIMENTO DE VARISTOR DE ÓXIDO DE ZINCO

Nesse estudo dos para-raios de óxido de zinco (ZnO), o COMSOL *Multiphysics* foi utilizado na análise do comportamento térmico desses dispositivos. Através da simulação computacional, é possível obter uma maior quantidade de dados, detalhados, sobre a distribuição de temperatura ao longo dos para-raios, identificar pontos de aquecimento excessivo e avaliar o desempenho térmico do dispositivo, agregando mais imagens para o treinamento da CNN.

Como forma de familiarização com o *software* foram desenvolvidos modelos introdutórios, como o apresentado na Figura 14. Ao modelar o para-raios no COMSOL, são considerados diversos parâmetros, como geometria do dispositivo, propriedades térmicas dos materiais utilizados, condições de contorno (tais como temperatura ambiente e taxa de transferência de calor para o ambiente) e as condições operacionais esperadas (incluindo corrente elétrica e tensão aplicada). A

partir dessas informações, é possível realizar simulações que reproduzem fielmente o comportamento térmico do para-raios em situações reais.

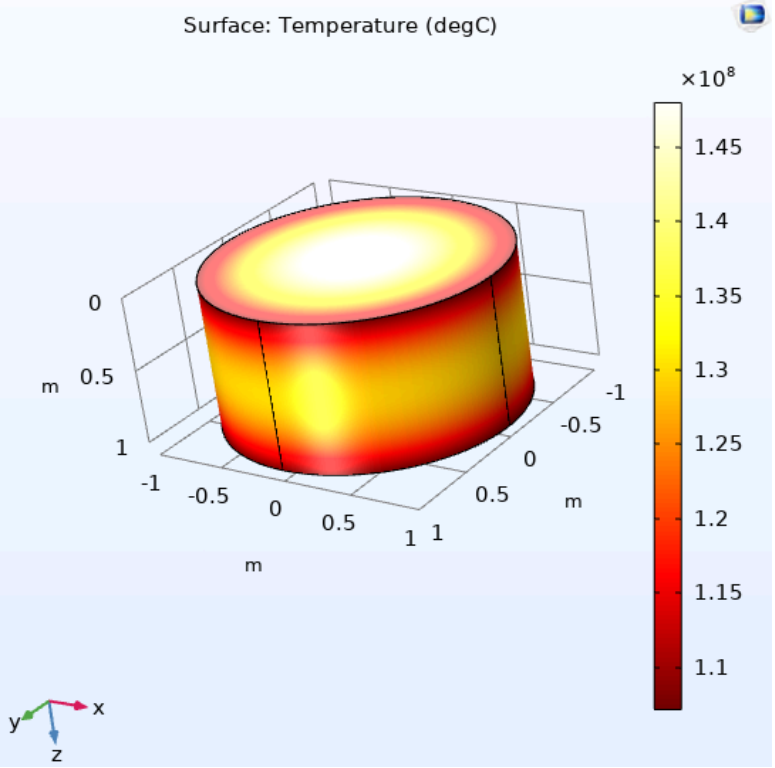



Figura 14- Modelo Introdutório Para Estudo.

Para definição de parâmetros referentes às características físicas intrínsecas ao modelo mais complexo, como por exemplo, a condutividade dos varistores de óxido de zinco. Foram necessários a extração de pontos de uma curva de um modelo real do para-raios, valores como tensão e corrente, para que adicionalmente a características geométricas do para-raio fosse possível determinar um cálculo aproximado para a condutividade do material, conforme mostrado nas equações a seguir.

$$\rho = (R \,.\, L)/\, A\ [\Omega{\cdot}m] \qquad (1)$$

$$R = V/\, I \qquad (2)$$

substituindo (2) em (1), obtém-se (3):

$$\rho = (V.\ L)/\,(A\,.\ I) \qquad (3)$$

e a condutividade sendo definida como,

$$\sigma = 1/\,\rho\ [(\Omega{\cdot}m) - 1]$$

onde ρ é a resistividade do material; σ corresponde a condutividade do material; V é a tensão no para-raios; I é a corrente elétrica no para-raios, e , por fim, A e L são, respectivamente, a área e o comprimento de cada bloco de varistor. Obtém-se a seguinte curva para a condutividade, apresentado na figura 15.

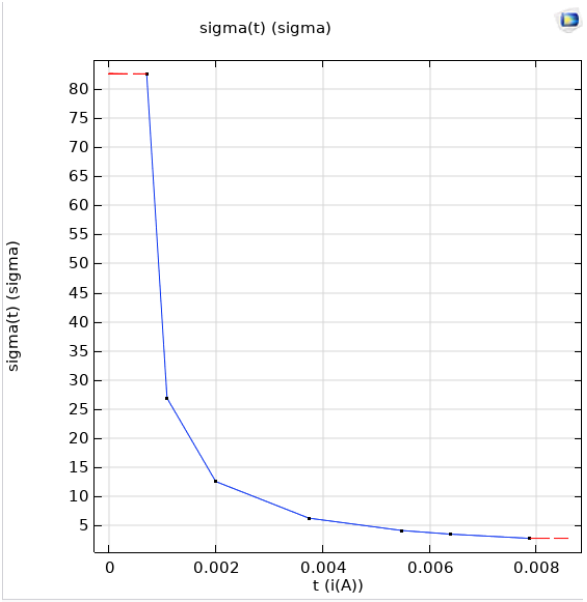



Figura 15- Condutividade dos Varistores

Em síntese, a geometria foi inicialmente desenvolvida no *AutoCad* e importada para o COMSOL, sendo possível definir um eixo de simetria para o equipamento com a finalidade de definir os materiais e os parâmetros necessários para a simulação, como apresentado a seguir na figura 16.

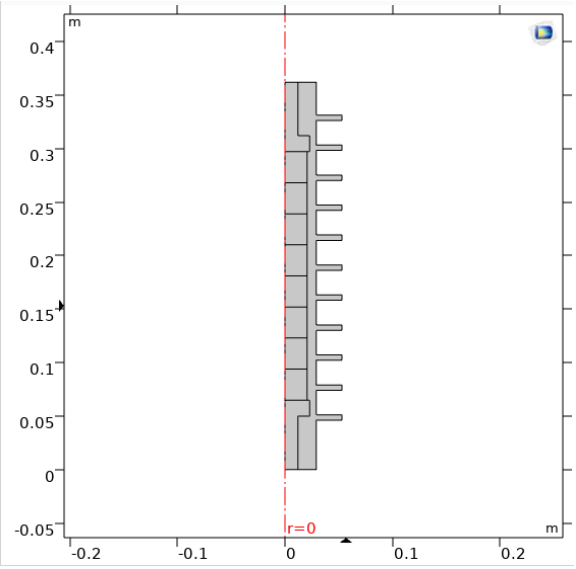


Figura 16- Modelo Final Para Estudo

Em resumo, o uso do COMSOL Multiphysics complementa as técnicas de termografia e aprendizado de máquina propostas neste estudo, oferecendo uma abordagem completa para avaliar e classificar o funcionamento dos para-raios de óxido de zinco. Essa integração entre simulação computacional, análise térmica e técnicas de inteligência artificial contribui significativamente para o avanço da pesquisa e desenvolvimento de dispositivos de proteção contra descargas atmosféricas, melhorando a segurança e a confiabilidade das redes elétricas.

Na continuação deste trabalho, buscar-se-á definir os parâmetros dos elementos da simulação de forma que possa ser validada em relação a medições reais, posteriormente, podem ser introduzidas pequenas modificações na simulação como forma de criar novos dados para a base disponível.

## 5. CONCLUSÃO

Ao implementar diferentes arquiteturas de redes neurais convolucionais para classificar defeitos em para-raios de óxido de zinco a partir de imagens termográficas, observou-se uma variação significativa nos desempenhos das redes. Com uma base de dados limitada, composta por imagens de para-raios bons e defeituosos, a análise do desempenho de cada modelo revelou os seguintes pontos:

A CNN clássica apresentou resultados satisfatórios em termos de acurácia e *loss*, mostrando que redes convolucionais tradicionais conseguem identificar padrões nas imagens, mesmo com uma quantidade de dados restrita. O gráfico de acurácia revelou uma evolução estável entre o treinamento e a validação, com o modelo não apresentando sinais evidentes de *overfitting*. Contudo, sua matriz de confusão indicou alguns erros na distinção entre para-raios bons e defeituosos, revelando que a simplicidade dessa arquitetura, embora eficaz, ainda enfrenta desafios no reconhecimento de características mais complexas.

A arquitetura Inception, projetada para ser mais eficiente e menos suscetível a *overfitting*, também apresentou resultados promissores. Entretanto, seus gráficos de acurácia e perda mostraram uma ligeira discrepância entre o treinamento e a validação, o que sugere que o modelo teve dificuldade em generalizar adequadamente. A matriz de confusão indicou uma boa capacidade de classificação para uma base de dados limitada, mas o desempenho geral ainda não atingiu níveis ótimos, possivelmente devido à falta de dados.

O modelo Xception teve o pior desempenho entre as arquiteturas avaliadas, com uma curva ROC (AUC = 0.57), e uma matriz de confusão que revelou uma alta taxa de erro. O modelo classificou erroneamente uma grande quantidade de para-raios bons como defeituosos (39 erros) e vice-versa. Essa arquitetura, que é bastante complexa e poderosa para grandes volumes de dados, parece não ser adequada para o conjunto de dados restrito utilizado, resultando em uma baixa capacidade de generalização. O *overfitting* pode ter ocorrido, já que o modelo não foi capaz de capturar padrões robustos nos dados de validação.

O modelo VGG16 apresentou um desempenho razoável, com uma Curva ROC (AUC = 0.71), indicando uma capacidade moderada de discriminar entre as classes. Embora tenha capturado bem os padrões do conjunto de dados, a matriz de confusão mostrou uma quantidade considerável de erros na classificação de para-raios bons e defeituosos, especialmente na identificação de para-raios bons. Isso sugere que o modelo foi um pouco mais eficiente que a Xception, mas ainda se deparou com limitações em relação ao tamanho do conjunto de dados.

A arquitetura VGG19 foi a que apresentou o melhor desempenho, com uma Curva ROC (AUC = 1.0), classificando quase perfeitamente os para-raios bons e defeituosos, com apenas um erro registrado na matriz de confusão. No entanto, esse desempenho excepcional deve ser interpretado com cautela, pois uma AUC de 1.0 pode ser um indicativo de *overfitting* no conjunto de validação. Isso sugere que, embora o modelo tenha se ajustado muito bem aos dados de validação, sua generalização para novos dados pode ser limitada.

A implementação de diferentes arquiteturas de redes neurais para a classificação de para-raios a partir de imagens termográficas evidenciou a importância da escolha correta da arquitetura em relação ao volume de dados disponíveis.

Com base nesses resultados, é possível concluir que, em situações onde há restrição de dados, arquiteturas mais simples e menos complexas, como a CNN clássica e o modelo Inception, podem ser mais eficazes. Redes mais elaboradas, como VGG19 e Xception, exigem bases de dados maiores para atingir seu pleno potencial, caso contrário, estão sujeitas ao risco de *overfitting*. Portanto, é necessário considerar a quantidade de dados e a complexidade da tarefa ao escolher o modelo mais adequado para as classificações.

## 6. AGRADECIMENTOS

Agradeço a todos os membros do Laboratório de Alta Tensão (LAT) da Universidade Federal de Campina Grande (UFCG) pelo valioso apoio na construção deste trabalho. Expresso minha gratidão, em especial, ao meu orientador, professor Dr. Pablo Bezerra Vilar, pela orientação e pelo esclarecimento das dúvidas ao longo do desenvolvimento desta pesquisa. Estendo meus agradecimentos a todos que, de maneira direta ou indireta, contribuíram para a realização deste estudo, incluindo todos aqueles que deram início a esta pesquisa. Este trabalho, vinculado ao PIBIC/CNPq-UFCG, contou com o apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq.

7. **REFERÊNCIAS**

8. LIRA, G. R. S. Monitoramento de para-raios de Óxido de zinco com base na medição da corrente de fuga total. *Tese de Doutorado*, Departamento de Engenharia Elétrica. Universidade Federal de Campina Grande – UFCG, 2012. Páginas
9.
10. R. Álvares C. Diagnóstico de falhas em para-raios utilizando termografias.

    Programa de Pós-graduação em Engenharia Elétrica - UFMG, p. 1–119, 2008.

    Páginas.
11.
12. SILVA, I. N. D. e. a. Artificial neural networks. *Cham: Springer International Publishing*, Springer, 2017. páginas
13.
14. DSA, E. Deep Learning Book. [S.l.]: Data Science Academy- Disponível em . Acesso em: 11 Março., 2022. páginas