

# Detecção de falhas em equipamentos por meio de inspeção visual

Iniciação Científica Voluntária

Murilo Capozzi dos Santos

Orientadora: Profa. Dra. Lilian Berton

*Instituto de Ciência e Tecnologia - UNIFESP*

São José dos Campos, Brasil

[murilo.capozzi@unifesp.br](mailto:murilo.capozzi@unifesp.br)

**Abstract**—Neste estudo, exploramos o potencial do aprendizado de máquina supervisionado, com ênfase nas redes neurais convolucionais (CNNs), para a detecção de falhas em equipamentos por meio da análise visual de superfícies danificadas. Investigamos o desempenho de diferentes modelos pré-treinados de CNNs e a aplicação de técnicas de aumento de dados (*Data Augmentation*) para aprimorar a capacidade de identificar falhas. Avaliamos esses modelos em termos de acurácia na detecção de falhas e a capacidade de previsão. Esta pesquisa tem o objetivo de contribuir para a automatização da detecção de falhas em processos industriais, visando melhorias na eficiência operacional e na redução de custos relacionados a falhas não detectadas precocemente.

**Keywords:** Detecção de falhas, Deep Learning, CNN, Transfer Learning

## I. INTRODUÇÃO

Segundo Norvig e Russell, a Inteligência Artificial (IA) permite que máquinas e sistemas aprendam com dados, tomem decisões autônomas e ajustem suas operações de acordo com as condições em tempo real. No contexto da produção, isso se traduz em melhorias significativas na eficiência, qualidade e flexibilidade. Ao usar técnicas de IA, as empresas podem desenvolver sistemas inteligentes capazes de monitorar, diagnosticar e até mesmo prever problemas em suas linhas de produção.

Com o avanço da tecnologia, diversas atividades que antes eram estritamente manuais e cansativas começaram a ser automatizadas. No entanto, aquelas que envolvem algum tipo de perigo quando realizadas de maneira incorreta continuaram a ser feitas manualmente, ou seja, mesmo com tanta tecnologia, o ser humano ainda depende da realização de tarefas manuais. Além disso, antes da popularização e eficácia das redes neurais convolucionais, outras tecnologias não eram tão precisas nem confiáveis a ponto de eliminar completamente a necessidade de interação humana.

Nesse cenário, quando pensamos em uma fábrica de equipamentos com capacidade de produção gigantesca ou em equipamentos que são usados por um longo período e exigem manutenção, as falhas só são detectadas quando já é tarde

demais, o que pode causar prejuízos. Em vez disso, a mão-de-obra humana é usada para verificar constantemente essas peculiaridades. Isso se torna um trabalho cansativo e monótono, no qual também ocorrem falhas, especialmente devido aos chamados falsos-positivos, nos quais uma falha não é identificada, ou quando não há falhas e erroneamente é detectado que houve.

É importante ressaltar que a Indústria 4.0, ou Quarta Revolução Industrial, está mudando a forma como as coisas são feitas na produção, e um dos motivos por trás disso é a Inteligência Artificial. De acordo com Ustundag e Cevikcan, essa quarta revolução industrial é descrita como uma mistura de Internet das Coisas (IoT), *Big Data* e IA para otimizar os processos de produção. Vale a pena ressaltar que a IA não apenas automatiza coisas, mas também torna as máquinas inteligentes, capazes de aprender e tomar decisões por conta própria. Isso significa eficiência, qualidade e flexibilidade melhores na fabricação, tudo graças ao cooperação entre a Indústria 4.0 e a IA [2].

O objetivo desse trabalho é realizar um comparativo entre diferentes modelos de redes neurais profundas na identificação de falhas visuais em imagens, a fim de investigar a eficiência da automatização de detecção de falhas em processos industriais.

## II. CONCEITOS FUNDAMENTAIS

É intrínseco que o leitor esteja familiarizado com estes tópicos, que serão apresentados logo após:

- A- Aprendizado de máquina supervisionado
- B- Perceptron e as redes neurais artificiais
- C- Processamento de imagem
- D- Redes neurais convolucionais
- E- Elementos essenciais para o treinamento

### A. Aprendizado de máquina supervisionado

Para Solange Rezende (1999, p. 39-43): "...é uma área de IA cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática." Especificadamente para o aprendizado supervisionado: "...extrair

um bom classificador a partir de um conjunto de exemplos rotulados". Dito isso, será desenvolvido um sistema inteligente que irá observar diversos exemplos/amostras da nossa base de dados (diversas imagens), esses sendo rotulados, e obter conhecimento.

### B. Perceptron e as redes neurais artificiais

O perceptron, termo criado em 1958 por Frank Rosenblatt, é o desenvolvimento hipotético de um sistema nervoso ou de uma máquina, designado para ilustrar as propriedades fundamentais de sistemas inteligentes em geral, sendo simples de entender e análogo aos sistemas biológicos que aprendemos na escola.

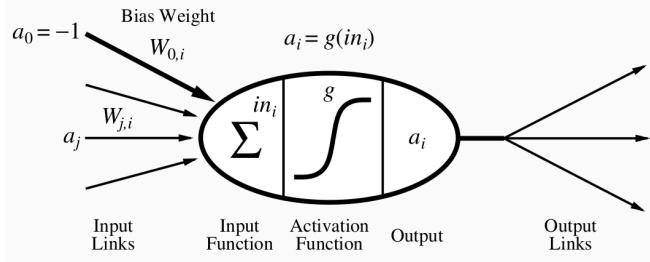


Fig. 1. O perceptron. Fonte: [4]

Consiste no tipo mais simples de rede neural artificial e, como vemos na Fig. 1, possui entradas  $in_i$  e seus respectivos pesos  $W_i$ . Adiante, há uma somatória do produto das entradas com seus pesos mais um termo constante  $b$  chamado de "bias", o resultado disso passa por uma função de ativação que irá nos proporcionar uma saída  $a_i$ . O aprendizado então tange sobre a descoberta dos melhores pesos para cada entrada.

O problema é a simplicidade extrema do perceptron, sendo efetivo apenas para problemas linearmente separáveis (que conseguimos traçar uma reta dividindo). Surgiram então as redes neurais artificiais (RNAs) com camadas ocultas, muito mais complexas e custosas mas que resolvem problemas não-linearmente separáveis. Observa-se na Fig. 2 que todos os nós de cada camada ligam-se a cada um dos próximos nós, isso é chamado de *Fully Connected* (FC). Cada aresta possui um peso próprio dada uma entrada associada a ele. A partir daqui, chamaremos-as apenas de RNAs.

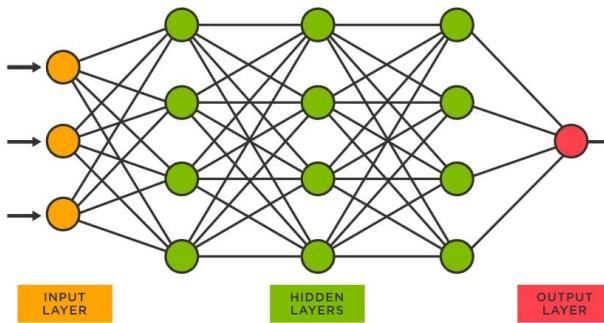


Fig. 2. Redes neurais artificiais. Fonte: [5]

### C. Processamento de imagem

Neste trabalho, iremos utilizar imagens de equipamentos como base de dados e, como vimos, para o sistema ser capaz de obter conhecimento, ele precisa de diversas amostras para isso, ou seja, quanto mais melhor. Dito isso, existem técnicas de processamento de imagens capazes de nos proporcionar variações da mesma imagem dada, tais como:

- Rotacionar
- Cortar
- Espelhar
- Recolorir

Chamamos esse processo de *Data Augmentation* e com ele conseguimos aumentar ainda mais o número de imagens disponíveis para o sistema aprender. Veja exemplo na Fig. 3.

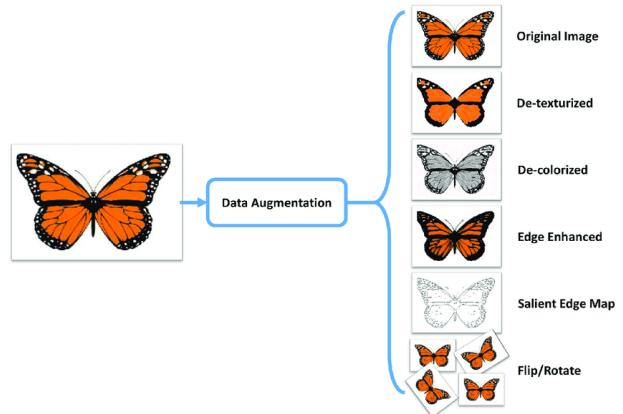


Fig. 3. Exemplos de técnicas de Data Augmentation. Fonte: [7]

### D. Redes neurais convolucionais

O produto das RNAs e o processamento de imagens trouxe ao mundo as CNN (*Convolucional Neural Networks*), uma arquitetura de Aprendizado Profundo (o famoso *Deep Learning*), e nela para cada pixel na imagem de entrada, codificamos algum valor, como por exemplo a intensidade do pixel associado na camada de entrada. O problema é que uma imagem compõe dezenas, centenas ou milhares de pixels, multiplicados pelo número de camadas, nós e arestas nos deparamos com um problema computacional exponencial. Afim de contornar a situação, o algoritmo da CNN atribui importância de "blocos" ou pequenas partes dentro da imagem, isto é, aplicam filtros que geram mapas de características os quais extraem o mais importante de cada pedacinho da imagem.

De maneira mais focada, o algoritmo recebe imagens e então a partir de filtros, geram mapas de características e esses serão transformados em um vetor para que entrem em uma RNA, também chamada de Rede Densa. Somadas ao *Data Augmentation*, criam-se estruturas poderosas para reconhecimento de padrões em imagens. Veja na Fig. 4 o tamanho que o sistema pode alcançar facilmente para apenas uma imagem.

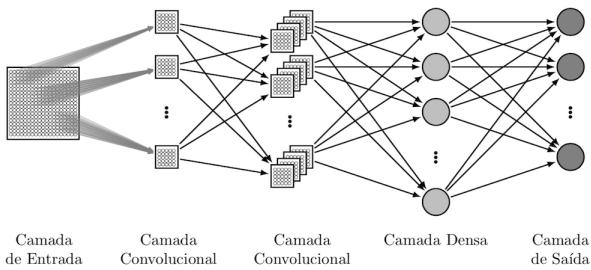


Fig. 4. Redes neurais convolucionais. Fonte: [8]

Há também a técnica chamada de *Transfer Learning*, ou Transferência de Aprendizado, que aproveita modelos pré-treinados, ou seja, algoritmos que já possuem os pesos totalmente treinados, disponíveis para o uso de qualquer outro objetivo adiante. Esses modelos treinaram em uma base de dados chamada ImageNet que contém mais de 14 milhões de imagens e 20.000 classes associadas às elas, porém, você pode importar este modelo e adaptar ao seu próprio interesse, podendo até mesmo "melhorar" camadas já treinadas (treinar elas novamente para o seu próprio modelo) chamado de *Fine Tuning* ou Ajuste Fino, descongelando camadas anteriores e instaurando outra base de dados específica.

Neste trabalho, utilizaremos os seguintes modelos pré-treinados, mostrando o porte do modelo e a quantidade de pesos ajustáveis dele:

TABLE I  
MODELOS PRÉ-TREINADOS

Modelo	Camadas	Parâmetros
MobileNetV2	154	2,257,984
VGG16	19	14,714,688
Xception	132	20,861,480
InceptionV3	311	21,802,784

Cada camada de um modelo tem uma função específica e contribui para o processamento progressivo dos dados de entrada. As camadas comuns em uma CNN incluem camadas de convolução, camadas de *pooling*, camadas de ativação e camadas totalmente conectadas (*Fully Connected* ou FC). A profundidade da rede, ou seja, o número de camadas, pode variar desde algumas camadas até várias dezenas de camadas. Já o número de parâmetros em uma CNN refere-se à quantidade de pesos ajustáveis que a rede precisa aprender durante o treinamento. Redes mais profundas com mais parâmetros têm a capacidade de capturar características de níveis mais elevados, mas também são mais propensas a problemas como o próprio *overfitting* e requerem mais recursos computacionais para treinamento e predição.

A fim de curiosidade, a Fig. 5 mostra um gráfico comparando diversos modelos pré-treinados existentes pela quantidade

de operações e a acurácia.

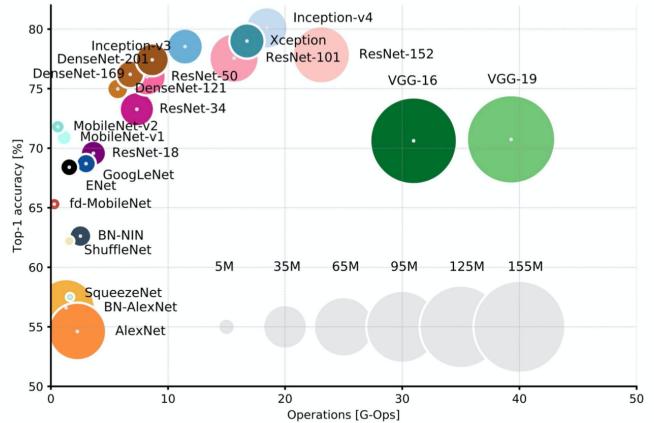


Fig. 5. Gráfico comparando modelos. Fonte: [10]

#### E. Elementos essenciais para o treinamento e avaliação

Existem diversos conceitos fundamentais que tangem o treinamento de modelos e a avaliação do desempenho em relação à classificação, abordaremos sobre eles a seguir:

- Tamanho do lote (*Batch*)
  - Dados são divididos em lotes (*batches*) de tamanho fixo. Cada lote é processado pelo modelo, e as atualizações dos parâmetros são realizadas com base nas amostras contidas no lote.
- Super-ajuste (*Overfitting*)
  - Ajuste excessivamente aos dados de treinamento, resultando em um desempenho ruim na generalização para novos dados. Perde a capacidade de generalizar e se torna excessivamente especializado nos exemplos específicos do conjunto de treinamento.
- Poda (*Dropout*)
  - Técnica de regularização para evitar *overfitting*, que consiste em remover/anular aleatoriamente um certo número de nós em uma determinada camada durante cada passo de treinamento. Força o modelo a aprender representações mais robustas.
- Épocas (*Epochs*)
  - O número de vezes que todo o conjunto de dados é percorrido durante o treinamento.
- Optimizador (*Optimizer*)
  - Responsável por ajustar os parâmetros do modelo com base na função de perda, buscando minimizá-la, existem diversas técnicas, as mais conhecidas são: SGD (*Stochastic gradient descent*), Adam e RMSProp. Utilizaremos do Adam.
- Função de perda (*Loss function*)
  - Medida de quão bem o modelo está performando em uma tarefa específica, como classificação ou

regressão. Utilizaremos da função de perda apropriada para classificação binária, *Binary Crossentropy*, e multiclasse, *Categorical Crossentropy*.

- Validação cruzada (*Crossvalidation*)

- Técnica utilizada para avaliar o desempenho de um modelo. Envolve a divisão dos dados disponíveis em conjuntos de treinamento e validação, onde o modelo é treinado em uma parte dos dados e avaliado em outra parte. Utilizaremos duas abordagens, sendo elas:

- *Hold-out*

- \* Envolve dividir os dados em diferentes conjuntos, sendo um conjunto para treinamento e outros conjuntos para validação e teste. Os conjuntos são estáticos, isto é, usados do início ao final do treinamento, sem variedade.

- *K-Fold*

- \* Envolve dividir a base de dados em k consecutivos de conjuntos de treino e teste, na qual cada *fold*, ou "pasta", é usado para validação e os  $k-1$  outros conjuntos usados para o treinamento. A vantagem da combinação entre os k conjuntos é que traz variabilidade ao treino, ou seja, diferentes cenários são explorados dada a mesma base de dados, trazendo confiança para posteriormente avaliar precisamente o modelo.

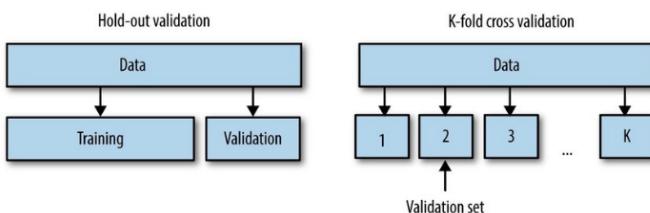


Fig. 6. Visualização do *Hold-out* e *K-Fold*. Fonte: [13]

- Métricas de avaliação

- Matriz de confusão

- \* Ferramenta fundamental para avaliar o desempenho de um modelo de classificação. Resumese por uma tabela que compara as previsões feitas por um modelo com as classes reais dos exemplos no conjunto de dados de teste. Ela é composta por quatro elementos principais:

- a) Verdadeiro Positivo (*True Positive - TP*): Representa os casos em que o modelo previu corretamente a classe positiva (a classe que estamos interessados) e a amostra realmente pertence a essa classe.
- b) Falso Positivo (*False Positive - FP*): Representa os casos em que o modelo previu erroneamente a classe positiva quando a amostra pertence à classe negativa. Isso indica que o modelo fez uma classificação incorreta, identificando um exemplo

como pertencente à classe positiva quando não era.

- c) Verdadeiro Negativo (*True Negative - TN*): Representa os casos em que o modelo previu corretamente a classe negativa, ou seja, a amostra não pertence à classe positiva e o modelo acertou essa previsão.
- d) Falso Negativo (*False Negative - FN*): Representa os casos em que o modelo previu erroneamente a classe negativa quando a amostra pertence à classe positiva (um erro do tipo II). Isso indica que o modelo não conseguiu identificar corretamente um exemplo que realmente pertence à classe positiva.

		Classe Preditiva	
		Positivo	Negativo
Classe Verdadeira	Verdadeiro	Verdadeiro	Falso
	Positivo (VP)	Negativo (FN)	
Negativo	Falso	Falso	Verdadeiro
	Positivo (FP)	Negativo (VN)	

Fig. 7. Conjuntos da matriz de confusão. Fonte: [14]

- Acurácia (*Accuracy*)

- \* Com base nos valores da matriz de confusão, podemos calcular a acurácia, métrica utilizada para avaliar o desempenho de um modelo de aprendizado de máquina, especialmente em tarefas de classificação. Ela fornece uma medida de quanto bem o modelo é capaz de fazer previsões corretas em relação ao número total de exemplos de teste. A fórmula para calcular a acurácia é bastante simples:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

- \* É uma das métricas mais importantes ao avaliar um modelo, no entanto, ela pode não ser a métrica ideal em todos os casos, especialmente quando os dados são desequilibrados (ou seja, uma classe tem muito mais exemplos do que outra). Em tais situações, outras métricas, como a precisão, a sensibilidade (recall) e a *F1-score*, podem ser mais informativas.

- *Grad-CAM* [15]

- Também conhecido como *Gradient-weighted Class Activation Mapping*, é uma técnica de visualização da saída das redes neurais convolucionais (CNNs). Essa abordagem é particularmente útil para entender quais partes específicas de uma imagem contribuem significativamente para a classificação realizada pela rede neural.

A característica distintiva do *Grad-CAM* é sua capacidade de mapear as regiões ativadas na imagem

de entrada sem a necessidade de acesso direto às camadas convolucionais internas da rede neural. Isso torna o *Grad-CAM* uma técnica mais genérica e aplicável a várias arquiteturas de CNNs.

Para criar esses mapas de ativação, *Grad-CAM* utiliza os gradientes de ativação da camada de saída da rede em relação à classe de interesse. Esses gradientes são ponderados com base na importância relativa de cada canal de ativação e, em seguida, somados para gerar um mapa de calor que destaca as regiões da imagem que influenciaram a classificação. O *Grad-CAM* tem diversas aplicações em áreas como medicina, visão computacional e análise de imagens, proporcionando uma compreensão mais intuitiva do funcionamento das CNNs e auxiliando na interpretação das características relevantes para a classificação de uma imagem.

Vale ressaltar que o tamanho do *batch*, o número de épocas, a porcentagem do *dropout* e outros são considerados hiperparâmetros, ou seja parâmetros de modelos que devem ser definidos antes de treinar o modelo, ajustáveis por execução ou experimento.

#### F. Arquitetura dos modelos

A fim de explorar mais o uso dos modelos propostos, será mostrado suas particularidades, isto é, suas arquiteturas. Com isso, veremos como foram construídas e a profundidade que elas podem alcançar.

- MobileNetV2 [16]

- A rede, como o próprio nome induz, é uma proposta para utilização em plataformas móveis e embarcados, ou seja, com recursos limitados. À respeito da sua arquitetura, é baseado em uma estrutura residual invertida, onde as conexões residuais estão entre as camadas de gargalo. A camada de expansão intermediária utiliza convoluções depthwise leves para filtrar características como fonte de não linearidade. Como um todo, a arquitetura do MobileNetV2 contém a camada convolucional inicial com 32 filtros, seguida por 19 camadas de gargalo residual.
- É usado o ReLU6 como não linearidade devido à sua robustez quando usado com computação de baixa precisão, além de um tamanho de kernel de  $3 \times 3$ , que é padrão para redes modernas, dropout e normalização em lote.

- VGG16 [17]

- A ideia surgiu do Grupo de Geometria Visual (em inglês Visual Geometry Group, por isso o nome VGG) e baseia-se na utilização de blocos e consiste em uma sequência de camadas convolucionais, seguido por uma camada de pooling máxima para downsampling espacial. Simonyan e Zisserman utilizaram convoluções com kernel  $3 \times 3$  com preenchimento de 1 e  $2 \times 2$  pool máximo com passo de 2, o

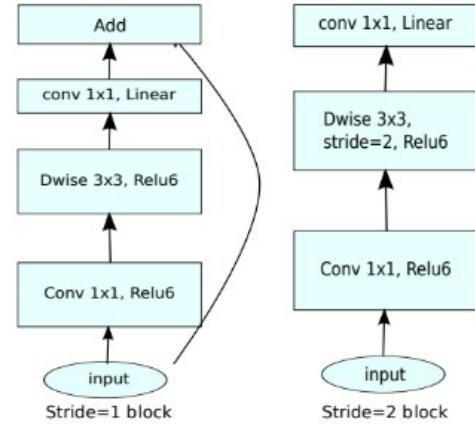


Fig. 8. Arquitetura MobileNetV2. Fonte: [16]

que reduz pela metade a resolução após cada bloco. Os blocos convolucionais da rede conectam os blocos VGG seguintes.

- A rede original, a VGG-11, possuía 5 blocos convolucionais, sendo uma camada convolucional para as duas primeiras e duas camadas convolucionais para as três restantes possuindo 64, 128, 256, 512 e 512 canais de saída respectivamente.

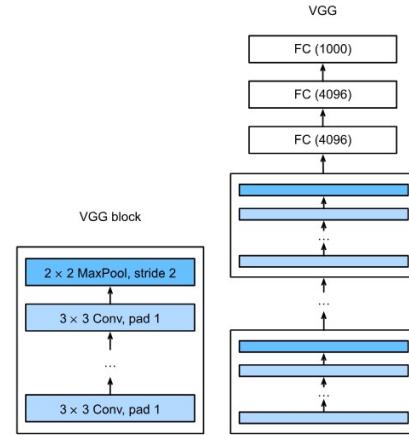


Fig. 9. Arquitetura VGG16. Fonte: [18]

- InceptionV3 [19]

- A ideia principal da arquitetura Inception é considerar como uma estrutura local esparça ideal de uma rede neural convolucional de visão pode ser aproximada e coberta por componentes densos prontamente disponíveis, sendo que a rede é construída a partir de blocos de construção convolucionais.
- Em geral, uma rede Inception é uma rede composta por módulos do tipo acima empilhados uns sobre os outros, com camadas de max-pooling ocasional com passo 2 para reduzir pela metade a resolução da grade. Por razões técnicas (eficiência de memória durante o treinamento), pareceu benéfico começar a

usar módulos Inception apenas em camadas superiores, mantendo as camadas mais baixas no estilo convolucional tradicional. Isso não é estritamente necessário, refletindo simplesmente algumas inefficiências infraestruturais em nossa implementação atual.

- Um aspecto útil dessa arquitetura é que ela permite aumentar significativamente o número de unidades em cada etapa sem um aumento descontrolado na complexidade computacional nas etapas posteriores. Isso é alcançado pelo uso ubíquo de redução de dimensionalidade antes das convoluções caras com tamanhos de patch maiores.

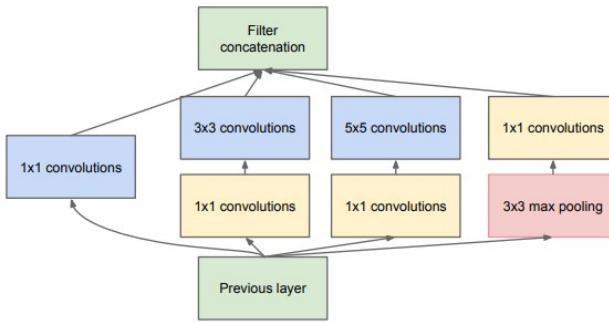


Fig. 10. Arquitetura InceptionV3. Fonte: [20]

- Xception [21]

- Baseia-se no conceito de redes convolucionais em cascata ao introduzir o conceito de convoluções separáveis em profundidade, fazendo com que convoluções sejam aplicadas separadamente em cada canal de entrada antes de serem combinadas. Além disso, possui técnicas de agrupamento denso, que visam reduzir a perda de informação durante o processo de downsampling, preservando características úteis para as camadas profundas.
- O mapeamento de correlações entre canais cruzados e correlações espaciais nos mapas de características de redes neurais convolucionais pode ser completamente desacoplado. Como essa hipótese é uma versão mais forte da hipótese subjacente à arquitetura Inception, a arquitetura proposta foi nomeada de Xception, que significa "Inception Extrema".
- Possui 36 camadas convolucionais que formam a base de extração de características da rede e estão estruturadas em 14 módulos, todos os quais têm conexões residuais lineares ao redor deles, exceto pelos módulos inicial e final.
- Em resumo, a arquitetura Xception é uma pilha linear de camadas de convolução separáveis em profundidade com conexões residuais. Isso torna a arquitetura muito fácil de definir e modificar.

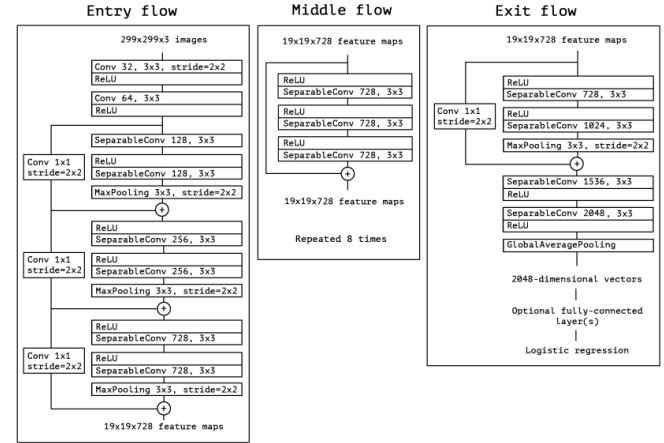


Fig. 11. Arquitetura Xception. Fonte: [21]

### III. TRABALHOS RELACIONADOS

Alguns artigos relacionados à detecção ou previsão de falhas em equipamentos:

- *Equipment fault detection using spatial signatures* [22]
  - O artigo também envolve detectar falhas em equipamentos, neste caso com foco em semicondutores, baseando-se em comparações e cálculos matemáticos para a predição sobre as superfícies dos mesmos.
- *Using SVM based method for equipment fault detection in a thermal power plant* [23]
  - Baseia-se na detecção de equipamentos derivados de usinas termelétricas e usa o modelo de máquinas de vetores de suporte (SVM), também supervisionado, com base em parâmetros das turbinas, e não imagens.
- *Fault Detection in Power Equipment via an Unmanned Aerial System Using Multi Modal Data* [24]
  - Busca identificar falhas em sistemas elétricos, como cabos, pratos e correntes danificados ou faltantes por meio da inspeção visual, isto é, por imagens. Funciona por meio de 2 modelos, na detecção de objetos e hot spots da imagem e também pelo modelo de CNN para classificação.
- *Fault Detection and Classification in Plasma Etch Equipment for Semiconductor Manufacturing e-Diagnostics* [25]
  - Desenvolve um sistema de detecção e classificação, chamado de FDC, para um equipamento para fabricação de semicondutores, o Plasma Etch. Utiliza de redes neurais modulares (MNN) e da teoria de Dempster-Shafter para associar o incerto a diagnósticos de danificações.
- *Visual dynamic model based on self-organizing maps for supervision and fault detection in industrial processes* [26]
  - Utiliza de técnicas de mineração de dados visuais para gerar um modelo de transições baseado em clusters e com isso um método de computação residual

- para detecção e identificação de falhas em uma planta industrial real.
- A CNN-based network failure prediction method with logs [27]
    - Diferencia os modelos de CNN, *Long Short Term Memory* (LSTM) e *Gated Recurrent Units* (GRU) para a predição de falhas a partir dos logs que simulam um sistema de telecomunicação wireless.
  - Fault mode recognition of planetary gears based on CNN and transfer learning [28]
    - O artigo utiliza também de CNN para o treinamento, com foco em um modelo pré-treinado para predição de falhas em engrenagens planetárias. Além disso, utiliza outras redes auxiliares GAN e DCNN.
  - CNN-Based Hybrid Optimization for Anomaly Detection of Rudder System [29]
    - Baseia-se na otimização híbrida para detecção de anomalias em sistemas de leme, ou seja, propõe um modelo otimizado da CNN chamado HPSOGWO-CNN baseado nos algoritmos *Particle Swarm Optimization* (PSO) e *Grey Worf Optimizer* (GWO).
  - Application of Deep CNN-LSTM Network to Gear Fault Diagnostics [30]
    - Utiliza de redes CNN juntamente com uma rede de memória de curto prazo, chamada de LSTM (CNN-LSTM), para classificação de falhas de dados de vibração de um sistema de maquete de caixa de engrenagens de helicóptero.
  - CNN based Gearbox Fault Diagnosis and Interpretation of Learning Features [31]
    - Diferentemente dos outros artigos, esse se aprofunda em, além de utilizar de CNN para classificar falhas de caixa de engrenagens, visualizar os recursos de aprendizado dos filtros criados pela CNN para entender os fenômenos de diagnóstico de falhas físicas, ou seja, permite observar a relação entre as frequências características e os filtros da CNN.
  - An Investigation Into Fault Diagnosis of Planetary Gearboxes Using A Bispectrum Convolutional Neural Network [32]
    - De forma semelhante a outro citado, estuda a predição de falhas em engrenagens planetárias com CNN e modelos pré-treinados, mas este artigo se aprofunda na análise do Bispectrum, transformando sinais de vibrações 1D para imagens 2D, que serão utilizadas como entrada para o modelo.
  - Triplet-Graph Reasoning Network for Few-Shot Metal Generic Surface Defect Segmentation [33]
    - Abordando literaturas com banco de imagens relacionado ao deste artigo, neste os autores utilizam de imagens também com defeitos na superfície de metais, mas são imagens diferentes. Utilizam da estratégia de TGRNet (*Triplet-Graph Reasoning Network*) com codificador triplo. O objetivo do estudo foi concentrado em realizar segmentações de áreas de foco na imagem, ou seja, encontrar os pontos importantes/relevantes de análise.
  - An End-to-End Steel Surface Defect Detection Approach via Fusing Multiple Hierarchical Features [34]
    - Neste artigo, os autores realizaram experimentos utilizando a mesma base de dados de superfície metálicas, as mesmas imagens e classes definidas e, além disso, abordaram sobre aprendizagem profunda, empregando também redes neurais convolucionais. Utilizam também de redes de fusão multi-níveis (MFN) e redes de propostas regionais (RPN) para gerar regiões de interesse. Com isso, chegaram a resultados surpreendentes, que variam em 99%.
  - A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects [35]
    - Os autores deste também usufruiram do mesmo banco de imagens de superfícies metálicas e classes definidas, com foco no estudo e aplicação de um método robusto de ruído baseado em padrões binários locais completos, com os modelos AE-CLBPs, CLBP e variações de LBP. Com elas, conseguiram alcançar acurácia maiores que 95% em todos os modelos.
- Visto todos estes trabalhos, e reforçando a ideia, fica explícito que o diferencial deste artigo em questão será o uso da CNN com *Transfer Learning* para predição de falhas em equipamentos e buscar pelos experimentos quais os melhores modelos pré-treinados para a aplicação.

#### IV. OBJETIVO

A seguir serão descritos os objetivos gerais e específicos:

##### A. Gerais

O objetivo deste trabalho é testar a eficiência e desempenho dos modelos pré-treinados em bases de dados de equipamentos defeituosos e determinar se são capazes de resolver o problema da identificação.

##### B. Específicos

- Avaliar os modelos pré-treinados no treinamento e teste nas determinadas bases de dados.
- Avaliar se os modelos são capazes de identificar corretamente.
- Avaliar qual modelo obtém acurácia mais alta.

#### V. METODOLOGIA EXPERIMENTAL

##### A. Base de dados

Fizemos experimentos com duas bases de dados, sendo a primeira de classificação binária e a outra categórica (ou multiclasse).

- 1) Este conjunto de dados se refere à fabricação de produtos de fundição. A fundição é um processo de fabricação no qual um material líquido é geralmente despejado em um molde, que contém uma cavidade oca com a forma

desejada, e depois é permitido que se solidifique. A razão para coletar esses dados são os defeitos na fundição. Defeito de fundição é uma irregularidade indesejada no processo de fundição de metal. Existem muitos tipos de defeitos na fundição, como porosidade, pequenos orifícios, rebarbas, defeitos de contração, defeitos no material do molde, defeitos no metal despejado, defeitos metalúrgicos, etc. Defeitos são algo indesejado na indústria de fundição. Para eliminar esses produtos defeituosos, todas as indústrias têm seu departamento de inspeção de qualidade. Mas o principal problema é que esse processo de inspeção é realizado manualmente. É um processo muito demorado e, devido à precisão humana, não é 100% preciso. Isso pode resultar na rejeição de todo o pedido, causando um grande prejuízo para a empresa. [36]

Todas essas fotos são vistas de cima do rotor da bomba submersível. O conjunto de dados contém um total de 7348 dados de imagem, divididos entre produtos "ok\_front", sem falhas, e "def\_front", defeituosos. Todas elas têm o tamanho de imagens em escala de cinza de 300x300 pixels.

- *ok\_front*



Fig. 12. Exemplo *ok\_front*.

- *def\_front*

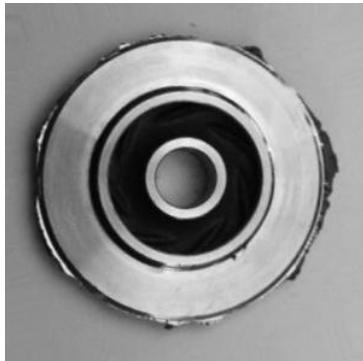


Fig. 13. Exemplo *def\_front*.

- 2) Por consequência da dificuldade para encontrar um banco de imagens propriamente ditas de equipamentos com

falhas, utilizaremos imagens de superfícies danificadas proporcionadas pela Universidade Northeastern [37]. No banco de dados de defeitos de superfície da Universidade Northeastern (NEU), são coletados seis tipos de defeitos de superfície típicos de tiras de aço laminadas a quente, ou seja, escama enrolada (*Rolled-in-scale*), remendos (*Patches*), trincas (*Crazing*), superfície picada (*Pitted-surface*), inclusões (*Inclusion*) e arranhões (*Scratches*). O banco de dados inclui 1.800 imagens em escala de cinza: 300 amostras de cada um dos seis tipos diferentes de defeitos de superfície típicos.

As Figuras 5,6,7,8,9 e 10 mostram as imagens dos seis tipos de defeitos de superfície típicos, a resolução original de cada imagem é de 200x200 pixels. Os defeitos intraclasses apresentam grandes diferenças na aparência, por exemplo, os arranhões (*Scratches*) podem ser arranhões horizontais, verticais e inclinados, entre outros. Ao mesmo tempo, os defeitos interclasses têm aspectos semelhantes, como escama enrolada, trincas e superfície picada. Além disso, devido à influência da iluminação e das mudanças de material, a escala de cinza das imagens com defeitos intraclasses varia. Em resumo, o banco de dados de defeitos de superfície da NEU apresenta dois desafios difíceis, ou seja, os defeitos intraclasses têm grandes diferenças na aparência, enquanto os defeitos interclasses têm aspectos semelhantes, e as imagens de defeitos são afetadas pela iluminação e pelas mudanças de material.

- *Crazing*

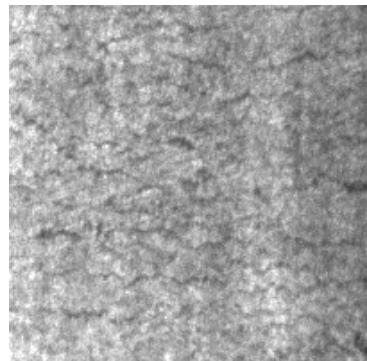


Fig. 14. Exemplo *crazing*.

- *Inclusion*

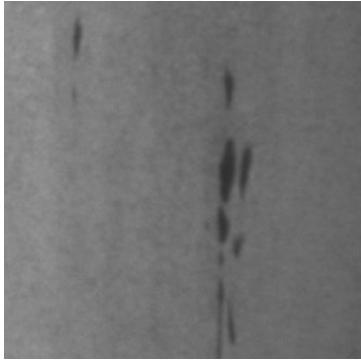


Fig. 15. Exemplo *inclusion*.

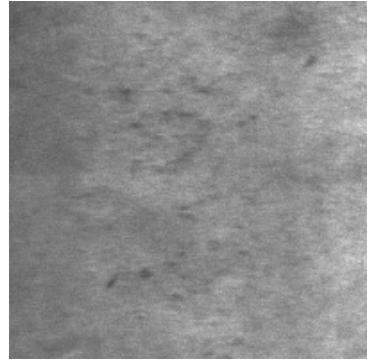


Fig. 18. Exemplo *rolled-in-scale*.

– *Patches*

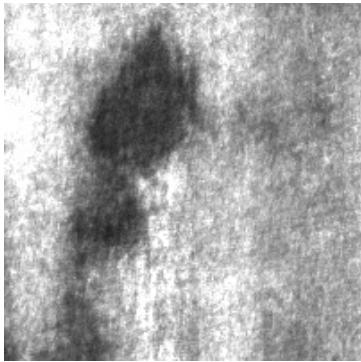


Fig. 16. Exemplo *patches*.

– *Pitted-surface*

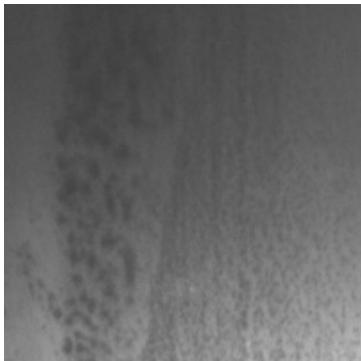


Fig. 17. Exemplo *pitted-surface*.

– *Rolled-in-scale*

– *Scratches*

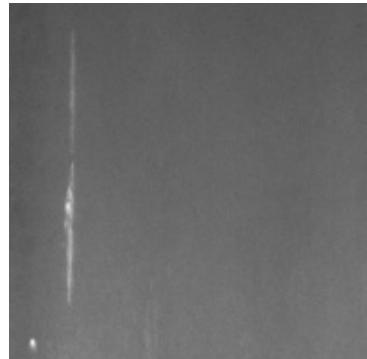


Fig. 19. Exemplo *scratches*.

### B. Protocolo de validação

Técnica muito utilizada em aprendizado de máquina para avaliar a variabilidade de um conjunto de dados e a confiabilidade dos modelos treinado com esses dados.

Mais especificadamente, utilizamos o *Hold-Out*, que irá repartir toda as imagens em 3 diretórios: treino, validação e teste na escala 70, 20, 10 % respectivamente. Ao longo do treinamento, iremos avaliar o desempenho das redes com a base de validação e, após finalizado o treinamento, a base de teste será utilizada para verdadeiramente avaliar o desempenho do modelo treinado. As duas bases foram testadas neste protocolo.

Além disso, utilizamos o *K-Fold* com k recebendo 5, ou seja, o conjunto de treinamento será dividido em 5 *folds*. Com isso, cada um irá treinar com 4 *folds* e testar (durante treinamento) com o outro restante. Ao final é calculada a média de acurácia entre os 5 treinamentos. Apenas a base de dados 2, disponibilizada pela NEU, foi testada neste protocolo.

### C. Experimento

Adotaremos os seguintes passos para os experimentos dados pelo diagrama de blocos do funcionamento dos experimentos na Fig. 20. Será repetido para cada modelo que testaremos

O código será desenvolvido na plataforma do Google Colab em Python, utilizando a biblioteca do TensorFlow juntamente com a do Keras para criação e treinamento dos modelos,

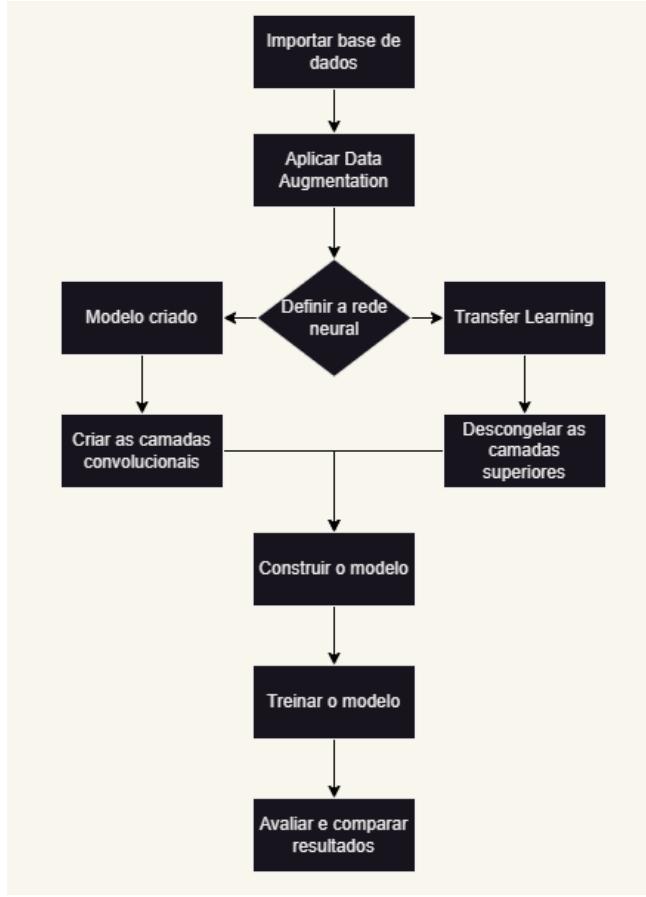


Fig. 20. Diagrama de blocos do funcionamento.

utilizando o *Transfer Learning* disponibilizado pelo próprio Keras (os modelos já citados anteriormente, o MobileNetV2, VGG16, Xception e o InceptionV3). Serão usadas funções da biblioteca do SciKit-Learn que fazem a divisão da base de dados em treino e teste, além de funções para avaliação tais como acurácia e matriz de confusão. Além destas, bibliotecas auxiliares, quase que básicas do Python serão usadas, como Numpy, Matplotlib, etc.

A respeito da medida de avaliação, será usada a acurácia e perda durante o treinamento e no teste. Além da matriz de confusão.

## VI. RESULTADOS E ANÁLISE

Antes de apresentar os resultados dos experimentos, na Tabela II, podemos visualizar os resultados obtidos pelas literaturas citadas que utilizaram o *dataset* 2.

TABLE II  
RESULTADO LITERATURA NO *dataset* 2 DE SUPERFÍCIES

Referência	Modelo	Acurácia (%)
HE at al. [34]	MFN+RPN	99.67
SONG, YAN [35]	CLP+AECLP+LBP	98.98+-0.63

Apresentaremos então os resultados, divididos pelas estratégias de validação cruzada: *Holdout* e *KFold*. É importante

frisar que apenas o segundo *dataset* será submetido ao *KFold*. Após mostrados, serão analisados.

### A. Holdout

No *dataset* 1, apenas foram submetidos para experimento a rede neural criada, o MobileNetV2 e o VGG16 pois, como visto, é um problema de classificação binária, os modelos, com pouco treinamento já são capazes de aprenderem a diferença de uma classe para outro. Pela quantidade enorme de imagens presentes do *dataset* e a resolução um pouco maior, foi utilizado *batch size* de 256 imagens, além da escolha de 30 épocas. Como já descrito, *loss function* utilizada foi a *Binary Crossentropy*, o otimizador Adam com *learning rate* de 0.001 e *dropout* de 30% para evitar *overfitting*. As Fig. 21 até 26 mostram os gráficos e matrizes de confusão de cada modelo após o treinamento no *dataset* 1. A Tabela III expõe a acurácia, em porcentagem, e a *loss* após o teste dos modelos na base de validação.

- Rede neural criada

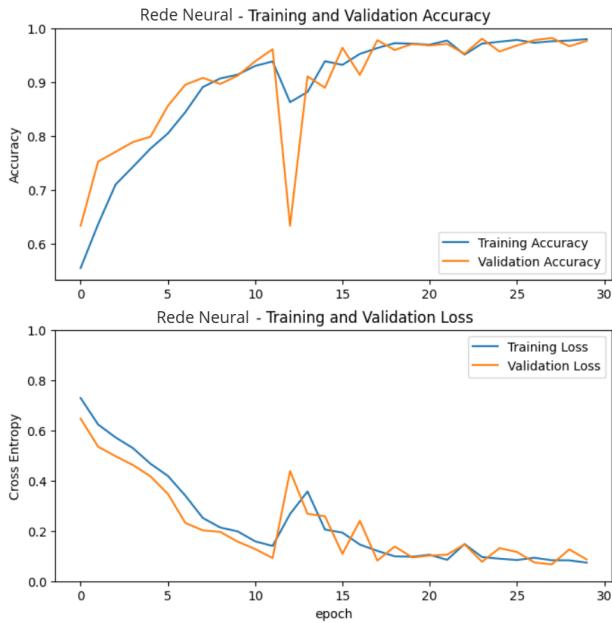


Fig. 21. Gráfico rede neural criada. (Dataset 1)

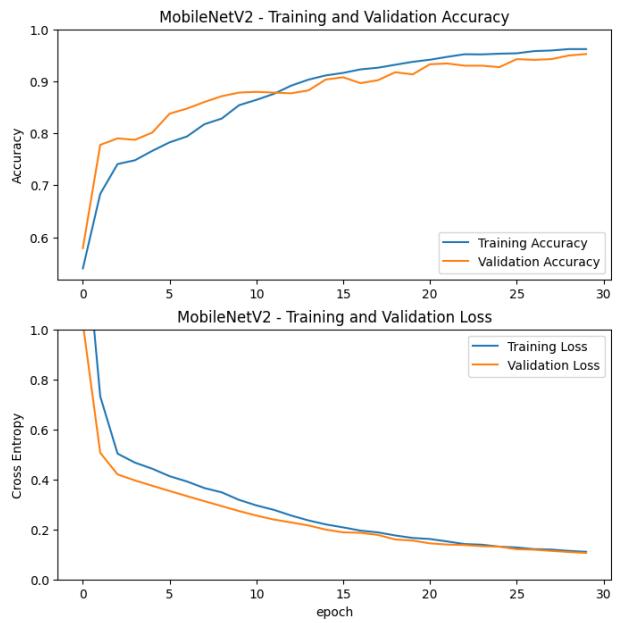


Fig. 23. Gráfico MobileNetV2. (Dataset 1)

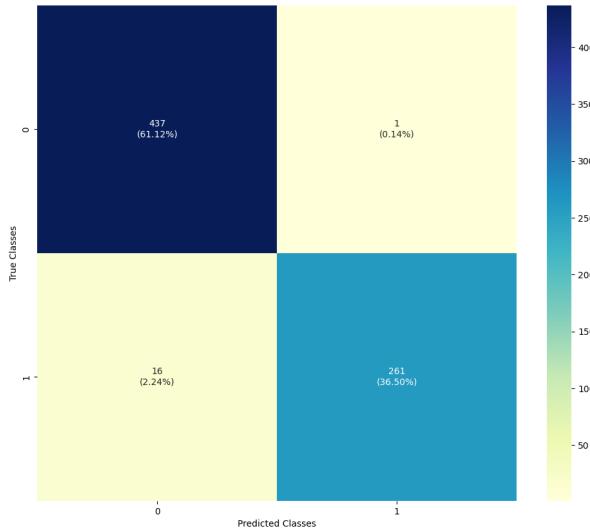


Fig. 22. Matriz de confusão rede neural criada. (Dataset 1)

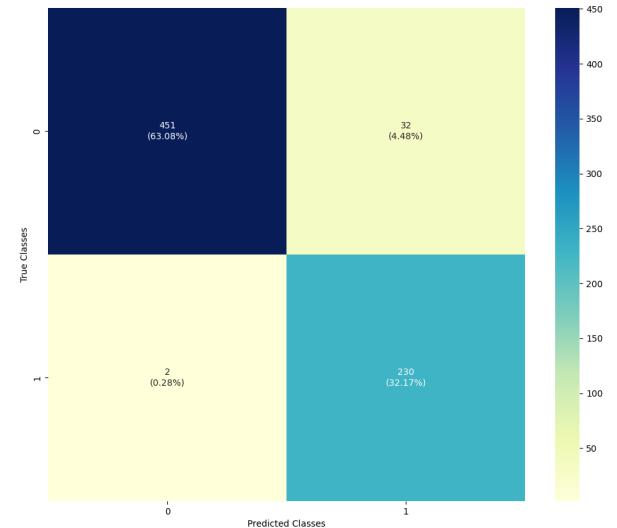


Fig. 24. Matriz de confusão MobileNetV2. (Dataset 1)

- MobileNetV2

- VGG16

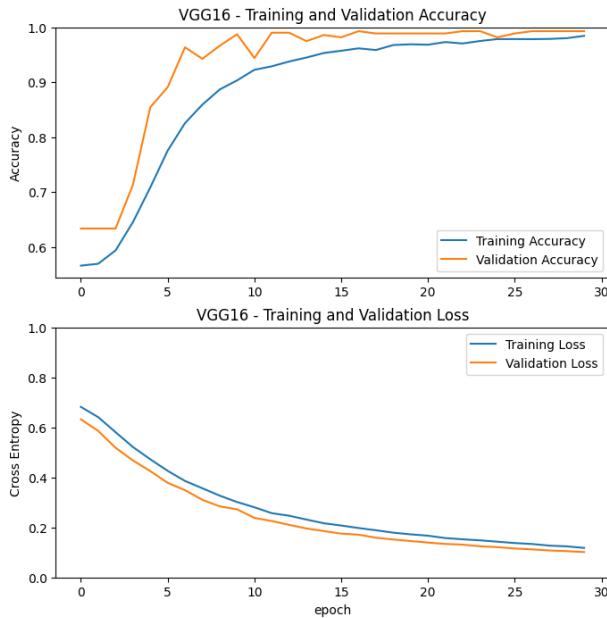


Fig. 25. Gráfico VGG16. (Dataset 1)

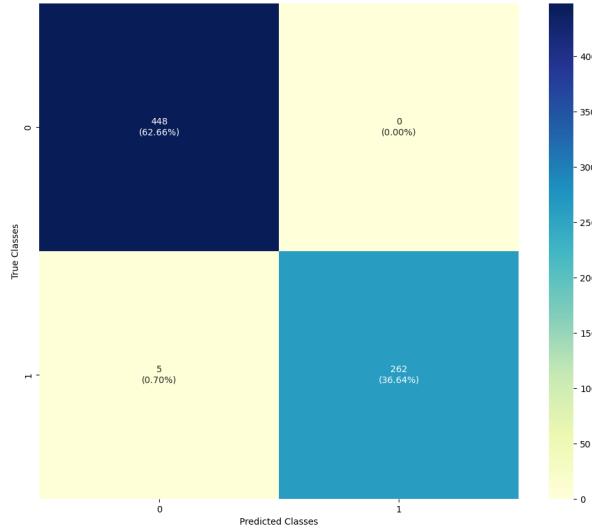


Fig. 26. Matriz de confusão VGG16. (Dataset 1)

TABLE III  
RESULTADOS DO TREINAMENTO Holdout APPLICADOS NO dataset 1, NA BASE DE VALIDAÇÃO

Modelo	Acurácia (%)	Loss
RedeNeural	97,62	0,133
MobileNetV2	95,24	0,140
VGG16	99,30	0,157

Para o *dataset 2*, por conta da menor quantidade de imagens e resolução, foi utilizado *batch size* com 32 imagens, com 100 épocas de treinamento, *loss function* sendo a *Categorical Crossentropy*, também aplicando o mesmo otimizador, Adam, *learning rate* de 0.001 e *dropout* de 30%. As Fig. 27 até 36 mostram os gráficos e matrizes de confusão de cada modelo

após o treinamento no *dataset 2*. A Tabela IV expõe a acurácia, em porcentagem, e a *loss* após o teste dos modelos na base de validação.

- Rede neural criada

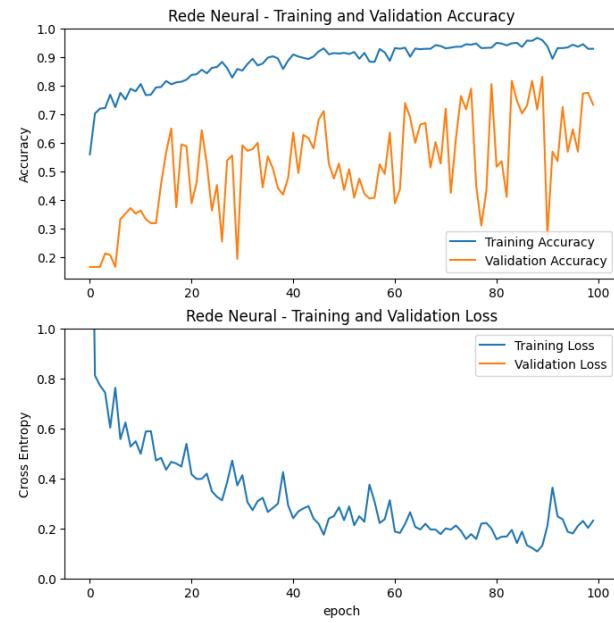


Fig. 27. Gráfico rede neural criada. (Dataset 2)

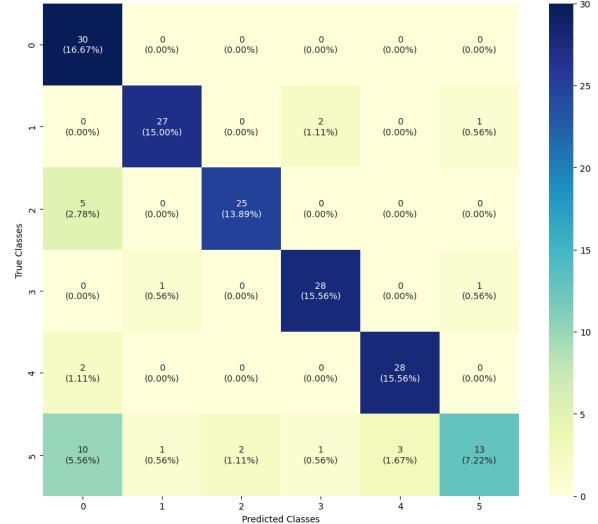


Fig. 28. Matriz de confusão rede neural criada. (Dataset 2)

- MobileNetV2

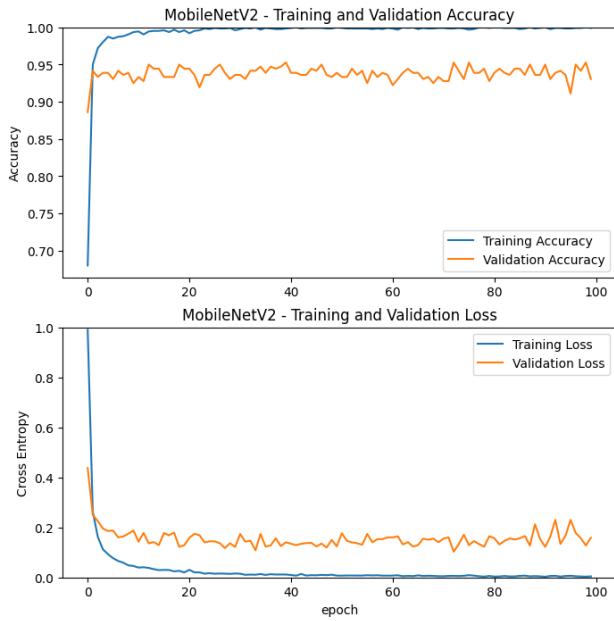


Fig. 29. Gráfico MobileNetV2. (Dataset 2)

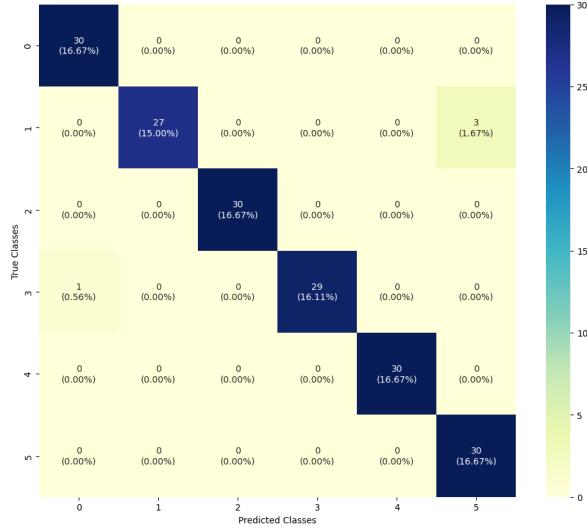


Fig. 30. Matriz de confusão MobileNetV2. (Dataset 2)

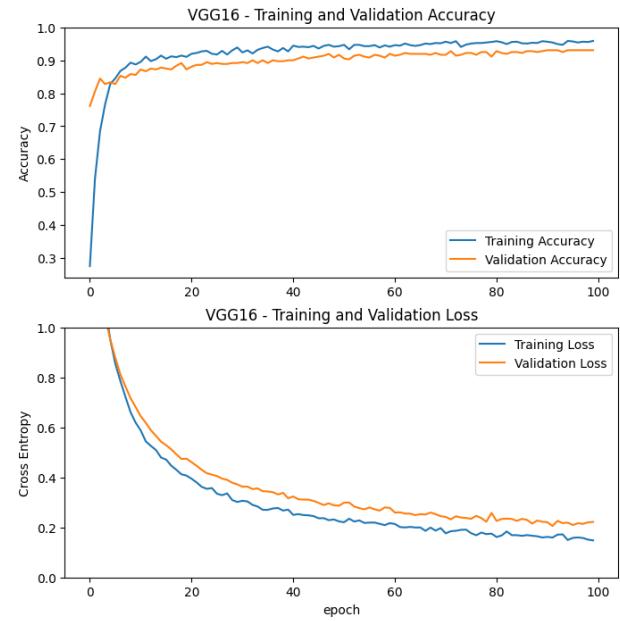


Fig. 31. Gráfico VGG16. (Dataset 2)

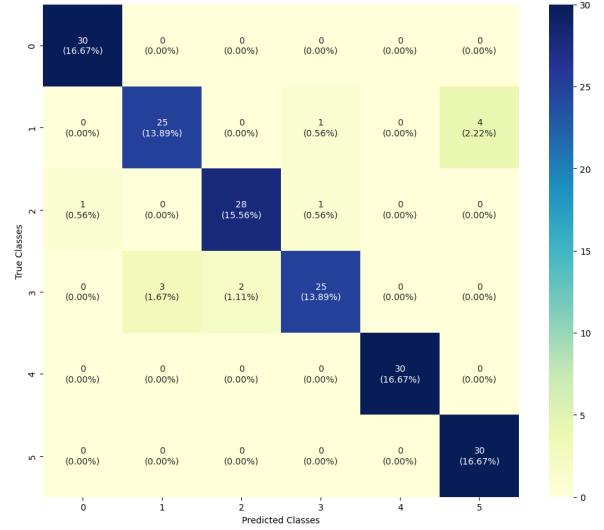


Fig. 32. Matriz de confusão VGG16. (Dataset 2)

- VGG16

- InceptionV3

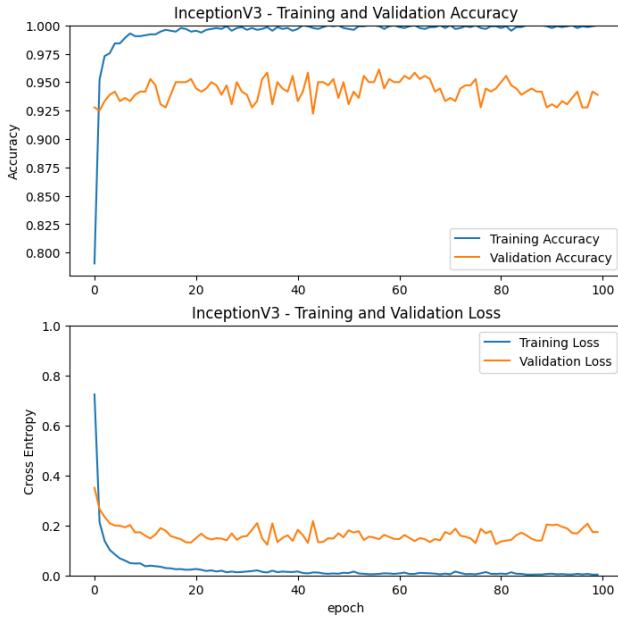


Fig. 33. Gráfico InceptionV3. (Dataset 2)

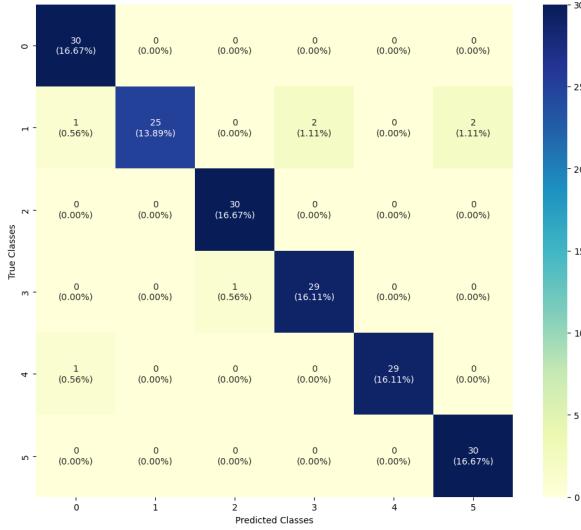


Fig. 34. Matriz de confusão InceptionV3. (Dataset 2)

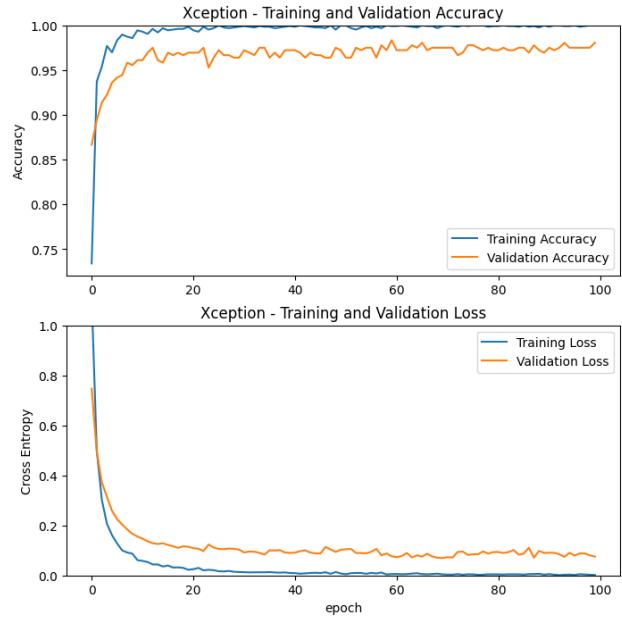


Fig. 35. Gráfico Xception. (Dataset 2)

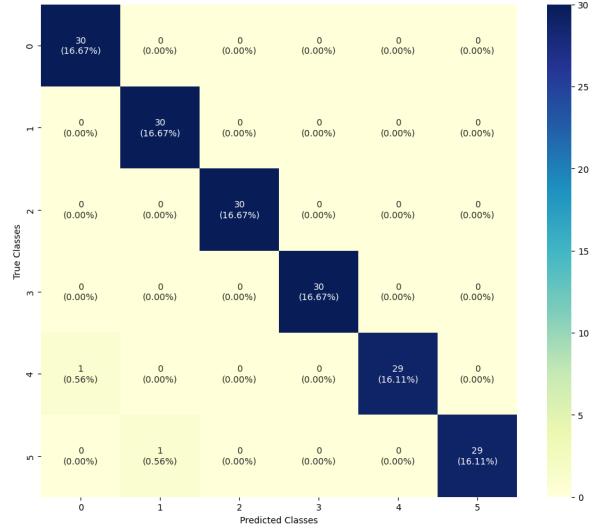


Fig. 36. Matriz de confusão Xception. (Dataset 2)

TABLE IV  
RESULTADOS DO TREINAMENTO Holdout APLICADOS NO dataset 2, NA BASE DE VALIDAÇÃO

Modelo	Acurácia (%)	Loss
RedeNeural	83,33	1.7817
MobileNetV2	97,77	0.0913
VGG16	93,33	0.1900
Xception	97,77	0.0437
InceptionV3	96,11	0.0814

### B. KFold

Como dito, apenas o *dataset 2* foi submetido ao *KFold* a fim de avaliarmos mais a fundo os modelos. Para isso, foi determinado que o número de *folds* seria 5, ou seja,

- Xception

4 *folds* para treinamento e 1 para teste (durante próprio treinamento). Apenas o número de épocas variou em relação aos hiperparâmetros, sendo 50 épocas para cada *fold*. As Fig. 37 até 46 mostram os gráficos e matrizes de confusão de cada modelo após o treinamento no *dataset* 2. As Tabela V, VI, VII e VIII expõem respectivamente a acurácia durante treinamento (aplicado ao *fold* restante), a *loss* durante treinamento, a acurácia de cada *fold* após treinamento aplicados à base de validação e a *loss* deste teste na base de validação.

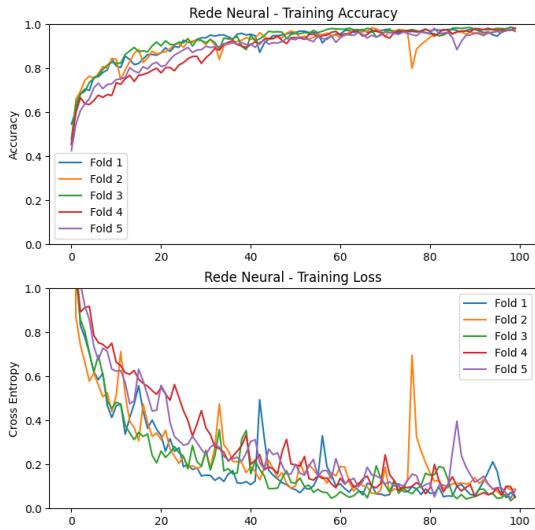


Fig. 37. Gráfico dos *folds* da Rede Neural.

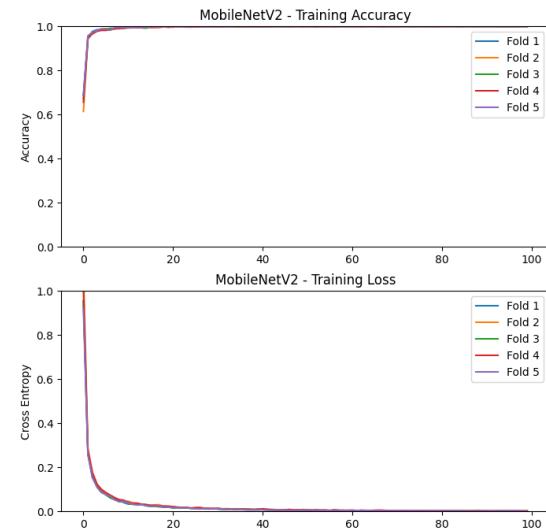


Fig. 39. Gráfico dos *folds* do MobileNetV2.

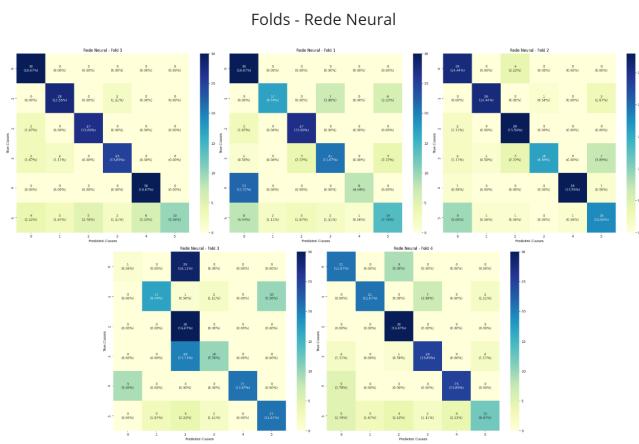


Fig. 38. Matriz de confusão *folds* da Rede Neural.

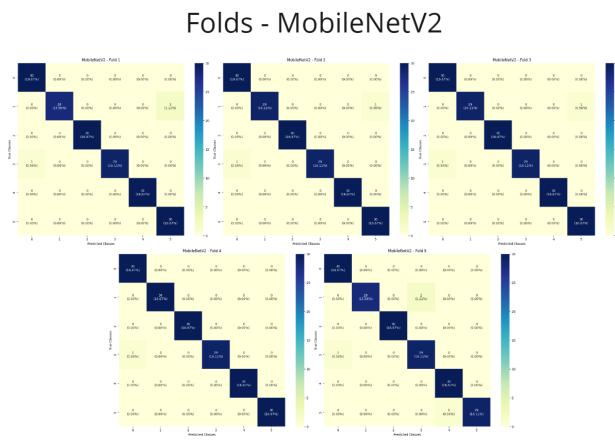


Fig. 40. Matriz de confusão *folds* do MobileNetV2.

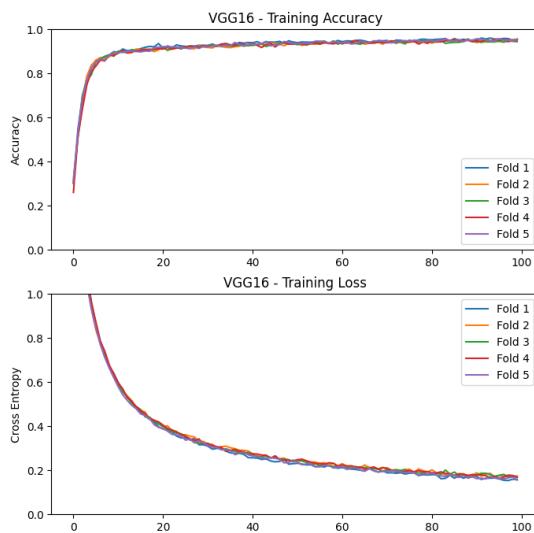


Fig. 41. Gráfico dos *folds* do VGG16.

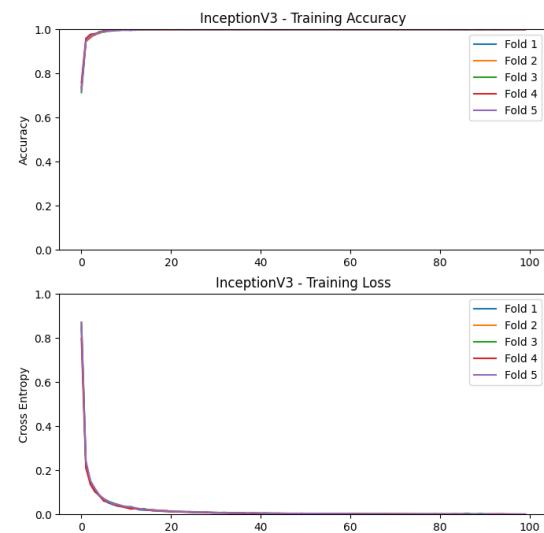


Fig. 43. Gráfico dos *folds* do InceptionV3.



Fig. 42. Matriz de confusão *folds* do VGG16.

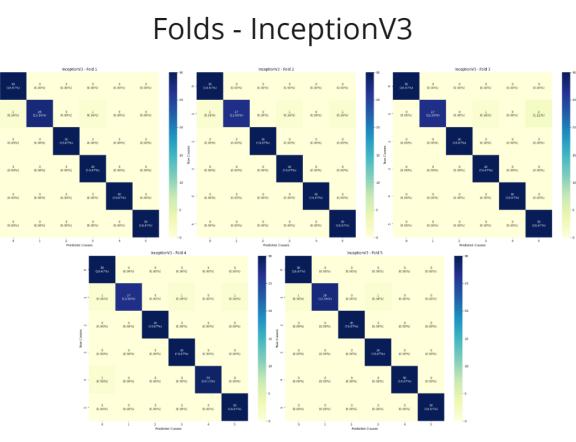


Fig. 44. Matriz de confusão *folds* do InceptionV3.

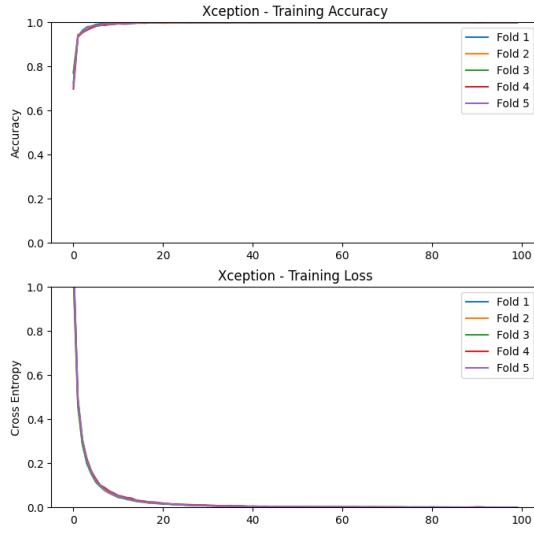


Fig. 45. Gráfico dos folds do Xception.

TABLE VI  
Loss DURANTE TREINAMENTO DO KFold APPLICADOS NO fold DE TESTE

Modelo	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Média
RedeNeural	1,400	1,318	11,824	0,563	1,047	3,2304
MobileNetV2	0,029	0,006	0,034	0,004	0,008	<b>0,0162</b>
VGG16	0,175	0,110	0,128	0,123	0,148	0,1368
Xception	0,022	0,033	0,011	0,006	0,036	0,0216
InceptionV3	0,060	0,010	0,017	0,025	0,021	0,0266

TABLE VII  
ACURÁCIA (%) DO KFold APPLICADOS NO dataset DE VALIDAÇÃO

Modelo	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Média
RedeNeural	64,99	78,88	55,55	74,44	83,33	71,43
MobileNetV2	98,33	98,88	98,88	99,44	97,77	98,66
VGG16	95,55	93,88	95,55	96,11	93,88	94,99
Xception	100,00	100,00	100,00	100,00	99,44	<b>99,88</b>
InceptionV3	98,88	98,33	98,33	97,77	98,88	98,43

Folds - Xception

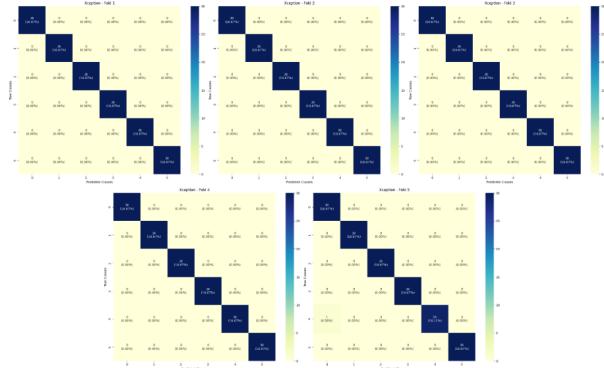


Fig. 46. Matriz de confusão folds do Xception.

TABLE VIII  
Loss DO KFold APPLICADOS NO dataset DE VALIDAÇÃO

Modelo	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Média
RedeNeural	6,366	5,524	10,556	2,298	2,062	5,361
MobileNetV2	0,040	0,054	0,044	0,040	0,051	0,045
VGG16	0,146	0,151	0,145	0,147	0,154	0,148
Xception	0,012	0,010	0,007	0,012	0,011	<b>0,010</b>
InceptionV3	0,047	0,074	0,043	0,037	0,034	0,047

### C. Grad-CAM

TABLE V  
ACURÁCIA (%) DURANTE TREINAMENTO DO KFold APPLICADOS NO fold DE TESTE

Modelo	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Média
RedeNeural	84,56	84,25	44,44	89,81	89,19	78,45
MobileNetV2	99,07	99,69	98,76	100,00	99,69	<b>99,44</b>
VGG16	94,13	97,22	97,22	97,53	95,06	96,23
Xception	99,07	99,07	99,69	99,69	99,07	99,31
InceptionV3	98,45	99,69	99,38	99,07	99,38	99,19

Com o intuito de visualizar quais partes da imagem cada modelo considerava relevante para a classificação, utilizaremos o *Grad-CAM* nos dois *datasets*, coletando uma amostra de cada classe aleatoriamente e analisando com sua versão com o mapa de calor. Além disso, apenas os modelos com as melhores acurácia serão aplicados, para visualizar os pontos realmente distinguem para fazer a classificação, com isso, para o *dataset 1*, apenas o modelo VGG16 foi submetido (Fig. 47) e, no *dataset 2*, o MobileNetV2 (Fig. 48) e o Xception (Fig. 49).



Fig. 47. Grad-CAM gerado pelo VGG16 em amostras do *dataset* 1.

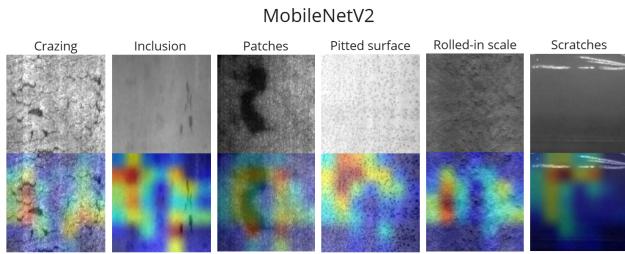


Fig. 48. Grad-CAM gerado pelo MobileNetV2 em amostras do *dataset* 2.

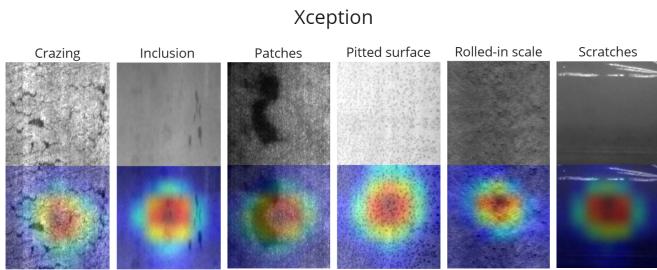


Fig. 49. Grad-CAM gerado pelo Xception em amostras do *dataset* 2.

#### D. Análise dos resultados

- *Dataset 1*

- Ao analisar os resultados obtidos pelo experimento, vemos que todos os modelos atingiram mais de 95% de acurácia, além da VGG16 atingir resultado maior que 99%. Vemos então que, pela simplicidade do problema, estes modelos tiveram ótimo desempenho mesmo com poucos parâmetros e/ou camadas em suas arquiteturas, incluindo a rede neural criada chegar em resultados tão altos e equiparando aos outros modelos neste *dataset*.
- Observando o mapa de calor gerado pelo VGG16, as áreas de relevância circulam a imagem, no entorno do rotor da bomba submersível e, na imagem da classe defeituosa, os arranhões foram pontos

muito importantes na classificação. Ou seja, não é necessário que toda a imagem seja verificada, o padrão para distinguir das classes ok e defeituosas é simples, não necessitando de modelos com mais parâmetros para isto.

- Resumidamente, a classificação binária pode ser considerada trivial para modelos pré-treinados, já que são construídos para lidarem com complexidades maiores.

- *Dataset 2*

- No experimento com o protocolo *Holdout*, notamos que a rede neural criada, que no *dataset* 1 tinha obtido 97% de acurácia, neste *dataset*, com mais classes e mais complexo, alcançou apenas 83,33% e com *loss* alta, além da instabilidade nos testes com a base de validação. Em contrapartida, os modelos MobileNetV2 e Xception, obtiveram igualmente a maior acurácia, de 97,77% e *loss* extremamente baixas, com foco na Xception que teve vantagem chegando a 0.0437. Vemos então, com a complexidade maior de classificação, que os modelos com mais parâmetros obtiveram melhores resultados, principalmente no quesito de estabilidade no treinamento. O VGG16 não obteve acurácia tão alta quanto no outro *dataset* também, mas manteve-se acima dos 90% e *loss* baixa com alta estabilidade durante treinamento e validação, podendo ter resultados melhores com mais épocas. O InceptionV3, modelo um pouco menos robusto que o Xception, obteve acurácia maior que 96% e *loss* muito baixa mas ainda com certa instabilidade durante o processo de validação.

- Já no experimento com o protocolo *KFold*, o resultado se manteve, na qual o MobileNetV2 e Xception mantiveram-se no topo porém, afirmou-se a superioridade do modelo mais robusto, o Xception, na base de validação, obtendo média de 99,88% e *loss* de 0.01 apenas. Em comparação quando em treinamento, na qual o MobileNetV2 mostrou-se ligeiramente melhor, mas acabou que adequando-se mais à base de treinamento e não manteve o desempenho na validação. A rede neural criada teve números ainda piores dos resultados do *Holdout*, afirmando que não é preciso o suficiente para chegar ao nível dos modelos pré-treinados, além da extrema instabilidade entre os *folds* durante treinamento. A VGG16 teve um ótimo desempenho apesar no número de camadas neste *dataset* mas, como esperado, foi menos preciso que os outros modelos pré-treinados. Neste protocolo, o InceptionV3 mostrou-se melhor e mais estável, obtendo médias maiores que 99% de acurácia no treinamento e 98% na validação.
- Vemos no mapa de calor gerado pelo MobileNetV2 e no Xception uma diferença interessante para a classificação dos modelos. No MobileNetV2, os pontos relevantes para classificar as imagens estavam aos

- redores, buscando as extremidades, enquanto que no Xception o foco sempre era o centro. Mesmo com a estratégia do MobileNetV2, não foi suficiente para superar a estratégia "simples" do Xception.
- Em resumo, apenas a rede neural criada não obteve resultados, o que era esperado pela complexidade do *dataset*, todos os modelos pré-treinados obtiveram ótimo desempenho. Com isso, prova-se a capacidade dos modelos da detecção e classificação, com destaque para o Xception.

## VII. CONCLUSÃO

Com o avanço do aprendizado de máquina e consequentemente do *Deep Learning*, técnicas de CNN conseguem substituir a mão-de-obra humana em tarefas repetitivas e monótonas como o reconhecimento de falhas em equipamentos. Com modelos e parâmetros corretos, além de uma base de dados suficientemente grande, são suficientes para criar modelos que serão muito eficientes e confiáveis, tanto em detecções simples como vistas no *dataset* 1 de existência ou não de defeitos, que pelos gráficos e matrizes de confusão, vimos que o VGG16 teve ótimo desempenho para esta tarefa, quanto no *dataset* 2 na classificação correta da falha, na qual o Xception obteve o melhor resultado na análise.

Além disso, é importante notar a necessidade de conduzir diferentes tipos de testes suplementares com conjuntos de dados mais extensos e variados, abrangendo diferentes variações de falhas e até mesmo outros modelos mais robustos.

## REFERENCES

- [1] NORVIG, P; RUSSELL, S. Inteligência Artificial. 3 ed, GEN LTC 2013. 1016 p.
- [2] USTUNDAG, A., CEVIKCAN, E.; Industry 4.0: Managing the Digital Transformation. Springer, 2018. 286 p.
- [3] REZENDE, Solange O., MONARD, Maria Carolina, CARVALHO, André C. P. L.. Sistemas Inteligentes para Engenharia: Pesquisa e Desenvolvimento. Anais III Workshop de Sistemas Inteligentes para Engenharia. Belo Horizonte: Editora UFMG, 1999.
- [4] ROSENBLATT, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 386–408. <https://doi.org/10.1037/h0042519>
- [5] Tibco. What is a Neural Network?. Disponível em: <https://www.tibco.com/reference-center/what-is-a-neural-network>. Acesso 28 set. 2023.
- [6] DE ALBUQUERQUE, Márcio Portes; DE ALBUQUERQUE, Marcelo Portes. Processamento de imagens: métodos e análises. Rio de Janeiro, Brasil, v. 12, 2000.
- [7] ResearchGate. Data augmentation using semantic-preserving transformation for SBIR. Disponível em: [https://www.researchgate.net/figure/Data-augmentation-using-semantic-preserving-transformation-for-SBIR\\_fig2\\_319413978](https://www.researchgate.net/figure/Data-augmentation-using-semantic-preserving-transformation-for-SBIR_fig2_319413978). Acesso em 28 set. 2023.
- [8] Sakurai. Implementando a estrutura de uma Rede Neural Convolucional utilizando o MapReduce do Spark. Disponível em: <https://www.sakurai.dev.br/cnn-mapreduce/>. Acesso em 28 set. 2023.
- [9] Medium. Tutorial — Transfer Learning aplicado no reconhecimento de flores, Disponível em: <https://medium.com/ensina-ai/tutorial-transfer-learning-3972cac5e9b5>. Acesso em 29 maio 2023.
- [10] CANZIANI, A., PASZKE, A., CULURCIELLO, E.; An Analysis of Deep Neural Network Models for Practical Applications. CoRR, abs/1605.07678, 2016. <http://arxiv.org/abs/1605.07678>.
- [11] Microsoft. Modelo de validação cruzada. Disponível em: <https://learn.microsoft.com/pt-br/azure/machine-learning/component-reference/cross-validate-model?view=azureml-api-2>. Acesso em 13 de jun de 2023.
- [12] Medium. Otimizando os hiperparâmetros. Disponível em: <https://medium.com/data-hackers/otimizando-os-hiperpar%C3%A2metros-621de5e9be37>. Acesso em 22 de jun de 2023.
- [13] Secure Learning para detección de Android Malware - Scientific Figure on ResearchGate. Disponível em: [https://www.researchgate.net/figure/Figura-44-Hold-out-y-K-fold-cross-validation-5\\_fig1\\_334119803](https://www.researchgate.net/figure/Figura-44-Hold-out-y-K-fold-cross-validation-5_fig1_334119803). Acesso em 2 out. 2023.
- [14] ResearchGate. ResiDI: Um Sistema de Decisão Inteligente para Infraestruturas Residenciais via Sensores e Atuadores Sem Fio - Scientific Figure on ResearchGate. Disponível em: [https://www.researchgate.net/figure/Figura-4-Medidas-de-desempenho-calculadas-a-partir-da-matriz-de-confusao\\_fig4\\_321162277](https://www.researchgate.net/figure/Figura-4-Medidas-de-desempenho-calculadas-a-partir-da-matriz-de-confusao_fig4_321162277). Acesso em 18 out. 2023.
- [15] SELVARAJU, R. R. et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision
- [16] SANDLER, M., HOWARD, A., ZHU, M., ZHMOGINOV A., CHEN, L.; MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381. DOI 10.48550/arXiv.1801.04381.
- [17] SIMONYAN, K., ZISSERMAN, A. (2014); Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. DOI 10.48550/arXiv.1409.1556.
- [18] Dive into Deep Learning. Redes Usando Blocos (VGG). Disponível em: [https://pt.d2l.ai/chapter\\_convolutional-modern/vgg.html](https://pt.d2l.ai/chapter_convolutional-modern/vgg.html). Acesso em 28 set. 2023.
- [19] SZEGEDY C. et al.; Going deeper with convolutions, 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.
- [20] Google Cloud. Guia avançado do InceptionV3. Disponível em: <https://cloud.google.com/tpu/docs/inception-v3-advanced?hl=pt-br>. Acesso em 23 out. 2023.
- [21] CHOLLET, F.; Xception: Deep Learning with Depthwise Separable Convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2017. p. 1251-8.
- [22] GARDNER et al.; "Equipment fault detection using spatial signatures" in IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part C, vol. 20, no. 4, pp. 295-304, Oct. 1997, doi: 10.1109/3476.650961.
- [23] CHEN, K.-Y., CHEN, L.-S., CHEN, M.-C., LEE, C.-L.; Using SVM based method for equipment fault detection in a thermal power plant, Computers in Industry, Volume 62, Issue 1, 2011, Pages 42-50, ISSN 0166-3615, doi: 10.1016/j.compind.2010.05.013. (<https://www.sciencedirect.com/science/article/pii/S0166361510000862>)
- [24] JALIL, B.; LEONE, G.R.; MARTINELLI, M.; MORONI, D.; PASCALI, M.A.; BERTON, A. Fault Detection in Power Equipment via an Unmanned Aerial System Using Multi Modal Data. Sensors 2019, 19, 3014. <https://doi.org/10.3390/s19133014>
- [25] HONG, S. J., LIM, W. Y., CHEONG, T., MAY, G. S.; "Fault Detection and Classification in Plasma Etch Equipment for Semiconductor Manufacturing e -Diagnostics," in IEEE Transactions on Semiconductor Manufacturing, vol. 25, no. 1, pp. 83-93, Feb. 2012, doi: 10.1109/TSM.2011.2175394.
- [26] FUERTES, J., DOMÍNGUEZ, M., REGUERA, P., PRADA, M., DÍAZ, I., CUADRADO, A.; Visual dynamic model based on self-organizing maps for supervision and fault detection in industrial processes, Engineering Applications of Artificial Intelligence, Volume 23, Issue 1, 2010, Pages 8-17, ISSN 0952-1976, <https://doi.org/10.1016/j.engappai.2009.06.001>.
- [27] JI, W., DUAN, S., CHEN, R., WANG, S., LING, Q.; "A CNN-based network failure prediction method with logs," 2018 Chinese Control And Decision Conference (CCDC), Shenyang, China, 2018, pp. 4087-4090, doi: 10.1109/CCDC.2018.8407833.
- [28] TANG, J., ZOU, Y., YU, J., ZHANG, Y.; "Fault mode recognition of planetary gears based on CNN and transfer learning," 2020 39th Chinese Control Conference (CCC), Shenyang, China, 2020, pp. 7235-7240, doi: 10.23919/CCC50068.2020.9188941.
- [29] WANG, W., YANG, R., GUO, C., QUIN, H.; "CNN-Based Hybrid Optimization for Anomaly Detection of Rudder System," in IEEE Access, vol. 9, pp. 121845-121858, 2021, doi: 10.1109/ACCESS.2021.3109630.
- [30] HAJ MOHAMAD, T., ABBASI, A., KIM, E., NATARAJ, C.; "Application of Deep CNN-LSTM Network to Gear Fault Diagnostics," 2021 IEEE International Conference on Prognostics and Health Man-

- agement (ICPHM), Detroit (Romulus), MI, USA, 2021, pp. 1-6, doi: 10.1109/ICPHM51084.2021.9486591.
- [31] LAL SENANAYAKA, J. S., VAN KHANG, H., ROBBERSMVR, K. G.; "CNN based Gearbox Fault Diagnosis and Interpretation of Learning Features," 2021 IEEE 30th International Symposium on Industrial Electronics (ISIE), Kyoto, Japan, 2021, pp. 1-6, doi: 10.1109/ISIE45552.2021.9576257.
- [32] PANG, X., XUE, X., JIANG, W., LU, K.; "An Investigation Into Fault Diagnosis of Planetary Gearboxes Using A Bispectrum Convolutional Neural Network," in IEEE/ASME Transactions on Mechatronics, vol. 26, no. 4, pp. 2027-2037, Aug. 2021, doi: 10.1109/TMECH.2020.3029058.
- [33] BAO, Y., SONG, K., LIU, J., WANG, Y., YAN, Y., YU, H., LI, X.; Triplet-Graph Reasoning Network for Few-Shot Metal Generic Surface Defect Segmentation, in IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1-11, 2021, Art no. 5011111, doi: 10.1109/TIM.2021.3083561.
- [34] HE, Y., SONG, K., MENG, Q., YAN, Y.; An End-to-End Steel Surface Defect Detection Approach via Fusing Multiple Hierarchical Features, in IEEE Transactions on Instrumentation and Measurement, vol. 69, no. 4, pp. 1493-1504, April 2020, doi: 10.1109/TIM.2019.2915404.
- [35] SONG, K., YAN, Y.; A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects, Applied Surface Science, 285, 858-864. doi:10.1016/j.apsusc.2013.09.002.
- [36] KANTESARIA et al. Casting product image data for quality inspection. Disponível em <https://www.kaggle.com/datasets/ravirajsinh45/real-life-industrial-dataset-of-casting-product/data>.
- [37] Northeatern University. Base de dados de defeitos de superfície NEU. Disponível em [http://faculty.neu.edu.cn/songkechen/zh\\_CN/zhyt/263269/list/index.htm](http://faculty.neu.edu.cn/songkechen/zh_CN/zhyt/263269/list/index.htm).