ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ARTIFICIAL INTELLIGENCE APPLIED TO QUALITY DEFECTS CLASSIFICATION FOR TYRES

INTELIGÊNCIA ARTIFICIAL APLICADA A CLASSIFICAÇÃO DE DEFEITOS DE QUALIDADE EM PNEUS

ALEX DA SILVA ALVES
https://orcid.org/0009-0006-9596-8883 / https://lattes.cnpq.br/4726734954179008 / asalves19899@gmail.com
FATEC SBC - Adib Moisés Dib, São Bernardo, São Paulo.

MICHAEL SAMPAIO DOS SANTOS
michaelsampaio1993@gmail.com
FATEC SBC - Adib Moisés Dib, São Bernardo, São Paulo.

GEDEANE KENSHIMA
https://orcid.org/0000-0002-0095-7529 / http://lattes.cnpq.br/1456607470616155 / gedeanekenshima@gmail.com
FATEC SBC - Adib Moisés Dib, São Bernardo, São Paulo

Recebido em: 14/06/2023
Aprovado em:26/03/2024
Publicado em: 22/04/2024

RESUMO

A manufatura de pneus radiais é complexa, e por este motivo alguns defeitos inevitavelmente acabam aparecendo no processo de produção. Atualmente, a detecção de defeitos comumente usada é a detecção manual, entre estas técnicas está o teste de ultrassom, detecção térmica, detecção por Raio-X, holografia digital, entre outros. O Raio-X ou radiografia técnica permite a análise de todos os componentes internos do pneu, como a estrutura de talões e as diferentes camadas de borracha e de cordas, sejam elas metálicas ou têxteis. A detecção de defeitos por Raio-X é simples e intuitiva, então é fácil para um humano detectar e classificar defeitos apenas observando as imagens, todavia a tarefa de detectar defeitos utilizando imagens de Raio-X em tons de cinza é feita de forma totalmente manual, e considerando a variabilidade das condições fisiológicas e técnicas dos operadores, há perda de tempo e custos para a empresa. O objetivo deste trabalho é a digitalização do processo convencional de análise de falhas de qualidade em pneus radiais metálicos utilizando algoritmos desenvolvidos com recursos de Inteligência Artificial, capaz de identificar e classificar defeitos presentes em imagens de Raio-X. Este trabalho apresenta uma breve introdução sobre o que é o

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

pneu, bem como o desenvolvimento do algoritmo proposto, utilizando técnicas clássicas de segmentação de imagens com uma biblioteca de Visão Computacional e duas estruturas de redes neurais convolucionais diferentes. Por intermédio da comparação, verificou-se que o algoritmo baseado em U-Net obteve resultados mais relevantes, porém, o algoritmo baseado em Mask R-CNN também se mostrou promissor, podendo ser útil em novos trabalhos, com uma base de dados maior e diferentes parâmetros de configurações.

Palavras-chave: defeito de qualidade, detecção, inteligência artificial, qualidade de pneus.

ABSTRACT

The manufacture of radial tires is complex, and for this reason some defects inevitably end up appearing in the production process. Currently the commonly used defect detection is manual detection, among these techniques are ultrasound testing, thermal detection, X-ray detection, digital holography, among others. The X-ray or technical radiography allows the analysis of all the internal components of the tire, such as the bead structure and the different layers of rubber and cords, whether metallic or textile. The task of detecting defects using X-ray images is done manually, which causes loss of time and costs for the company. Furthermore, it is a subjective, inefficient, time-consuming and even biased process as it requires a high level of concentration and focus. The objective of this work is the digitization of the conventional process of analysis of quality failures in metallic radial tires using algorithms developed with Artificial Intelligence resources, capable of identifying and classifying defects present in X-Ray images. This work presents a brief introduction about what the tire is, as well as the development of the proposed algorithm, using classic image segmentation techniques with a Computer Vision library and two different convolutional neural network structures. Through the comparison, it was verified that the algorithm based on U-Net obtained more relevant results, however, the algorithm based on Mask R-CNN was also promising and could be useful in new works, with a larger database and different settings parameters.

Keywords: artificial intelligence, quality defect, detect, tyre quality.

1 INTRODUÇÃO

A manufatura de todos os tipos de pneus radiais é caracterizada por uma série de processos complexos, e por isso alguns defeitos inevitavelmente surgem no processo de produção (ZHENG *et al.*, 2018). A fabricação desses pneus requer a conformidade estrita com limites e tolerâncias, a fim de garantir que o produto final atenda às especificações técnicas e normas regulamentadoras. Cada país costuma ter seu órgão regulamentador local que rege as especificações técnicas a serem atendidas, por exemplo para venda no Mercado Americano deve-se seguir os padrões indicados na

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

TRA (*The Tire and Rim Association*) (GENT e WALTER, 2005), já para vender produtos no Brasil deve-se seguir os padrões indicados na ALAPA (Associação Latino-Americana de Pneus e Aro) e consequentemente obter a certificação do INMETRO (Instituto Nacional de Metrologia, Qualidade e Tecnologia).

Nesse contexto, para assegurar uma ótima qualidade do pneu, a inspeção de qualidade, antes do pneu deixar a fábrica, é uma etapa essencial do processo de manufatura do pneu (LI *et al.*, 2021). Os sistemas de detecção de defeitos mais comuns incluem ultrassom, detecção térmica, holografia digital, detecção por Raio-X, entre outras. Dentre estes processos o mais conhecido nas indústrias de pneus para detecção de defeitos estruturais do pneu é o Raio-X.

Nas companhias de manufatura de pneus a tarefa de detectar defeitos utilizando imagens de Raio-X é feita manualmente por um funcionário, e isso causa perda de tempo e custos para a empresa. Além disso trata-se de um processo subjetivo, e até tendencioso pois requer um alto nível de concentração e foco. A maioria dos métodos de detecção de defeitos de Raio-X em pneus apresentados em literatura enfrentam dois tipos de dificuldades. A primeira é que as imagens dos Raio-X dos pneus são muito variáveis, pois existem centenas de especificações e desenhos diferentes. A outra dificuldade é que além das falhas distintas, falhas da mesma classe apresentam variações entre elas mesmo, dificultando ainda mais a identificação e classificação de defeitos (SALEH *et al.*, 2023).

A solução proposta visa substituir esse modelo de análise manual a olho nu, realizando uma análise com alto processamento computacional com a utilização de Inteligência Artificial (IA), gerando alertas ao identificar os padrões de defeitos previamente definidos e induzidos. O objetivo deste trabalho é aumentar a precisão de detecção usando as tecnologias de CNN (*Convolutional Neural Network*) (PAHINKAR *et al.*, 2021), utilizando especificamente as arquiteturas Mask R-CNN (*Mask Regional Convolutional Neural Network*) e U-Net para validação. Assim como funciona para o homem, o computador será capaz de aprender a classificar os defeitos, utilizando como referência uma grande base de dados, técnica conhecida como Aprendizagem Supervisionada.

Sendo assim, um determinado defeito foi definido como objeto a ser identificado e classificado e a partir dessa definição foram criados três conjuntos de dados diferentes, necessários para treinar

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

o algoritmo, sendo o conjunto de treinamento, conjunto de validação e conjunto de teste. O conjunto de treinamento é composto pelas imagens originais disponibilizadas no processo de Raio-X. É utilizando essas imagens que a rede neural aprenderá os pesos ideais para realizar as inferências de modo preciso. O conjunto de validação é composto por máscaras, que são os elementos das imagens do conjunto de treinamento que devem ser segmentados, ou seja, o defeito. Portanto, para cada imagem do conjunto de treinamento, existe uma máscara no conjunto de validação. Normalmente as máscaras são imagens binarizadas (somente com pixels branco e pretos, onde o defeito é representado por pixels brancos e todo o resto da imagem por pixels pretos) ou então são criadas através das coordenadas de cada ponto do defeito, representados nos eixos X e Y, técnica conhecida como anotação. Finalmente, o conjunto de testes também possui imagens originais, porém elas não fazem parte do conjunto de treinamento. Essas imagens serão submetidas ao algoritmo, com o intuito de verificar a sua acurácia. Para comparar o percentual de acerto da rede neural, a cada época de treinamento, as imagens de validação são submetidas à rede, sendo possível comparar a segmentação realizada pela rede neural com a máscara segmentada que possui as respostas desejadas. Após o treinamento da rede neural, as imagens do conjunto de testes são submetidas para que a rede as classifique (MALEKI *et al.*, 2020).

2. FUNDAMENTAÇÃO TEÓRICA

As seções a seguir apresentam sobre o uso de Raio-X e suas características, além das redes neurais convolucionais U-NET e Mask R-CNN.

2.1 Uso de Raio-X na classificação de pneus

O processo de Raio-X é um processo convencional que apresenta uma necessidade tecnológica de alta importância. A tecnologia de detecção de defeitos por Raio-X é simples e intuitiva, então é fácil para um humano detectar e classificar defeitos apenas observando as imagens (ZHENG *et al.*, 2018), todavia o classificador final é o ser humano, que está sujeito a apresentar falhas dependendo de suas condições fisiológicas e técnicas.

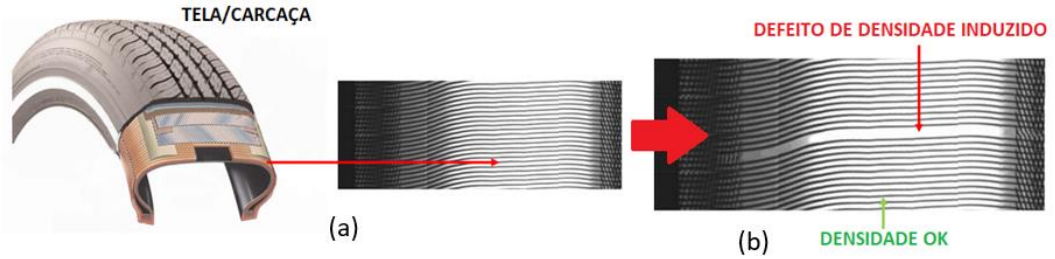
O Raio-X ou radiografia técnica permite a análise de todos os componentes internos do pneu, como a estrutura de talões e as diferentes camadas de borracha e de cordas, sejam elas metálicas ou têxteis. Durante o processo de análise, são obtidas imagens em três dimensões de alta

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

resolução, contraste e resolução espacial, o que permite diferenciar até compostos de borracha. Estes ensaios permitem a detecção de vários tipos de anomalias existentes em qualquer parte da estrutura do pneu analisado.

Permitem ainda, verificar a regularidade do espaçamento entre cordas, a regularidade das viradas da tela, o posicionamento, a qualidade das uniões e descentragens de cintas e a concentricidade dos feixes de arames dos talões (GENT e WALTER, 2005). Na Figura 01 é possível observar uma imagem isométrica de um pneu com as indicações dos componentes internos, juntamente com a imagem de Raio-X parcial da carcaça do pneu:

Figura 01 – Estrutura do pneu (a) representação da tela/carcaça do pneu (b) defeito induzido



Fonte: Adaptado de Gent e Walter (2005)

Ao observar a Figura 01, nota-se que a tela do Raio-X na Figura 01(a), por não ter interferência de outros componentes metálicos na região do flanco, possui uma continuidade e uniformidade nas linhas das cordas. Este é o resultado que se deseja obter na manufatura de um pneu bem projetado, seja por parte de produto ou por parte de processo. Já na Figura 01 (b) é possível notar que há um espaçamento significativo entre as linhas das cordas, caracterizando o defeito de variação na escala ou densidade da tela mencionado. O defeito indicado na Figura 01 (b) foi induzido para que fosse possível validar o sistema proposto.

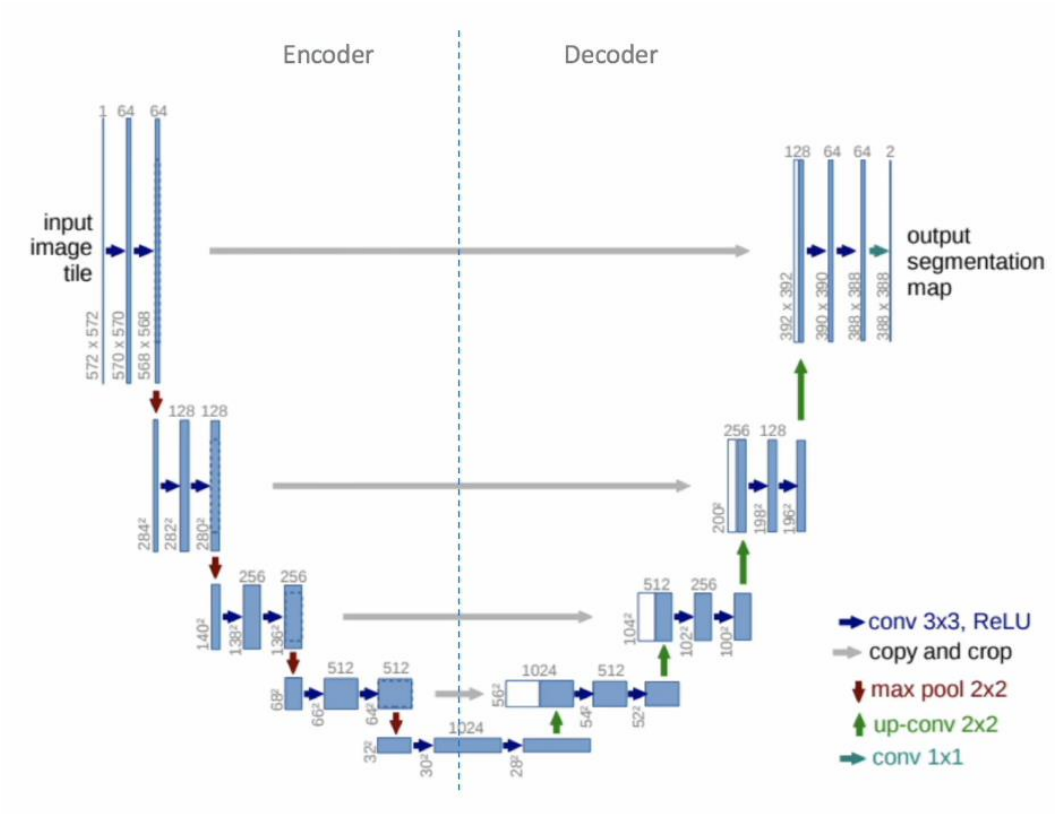
2.2 Arquitetura U-Net

U-Net é uma das estruturas de segmentação semântica mais importantes para uma rede neural convolucional (CNN). É amplamente utilizada no domínio da análise de imagens médicas

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

para segmentação de lesões, segmentação anatômica e classificação. A vantagem dessa estrutura de rede é que ela pode não apenas segmentar com precisão o alvo do recurso desejado e processar e avaliar de forma objetiva as imagens, mas também ajudar a melhorar a precisão no diagnóstico por imagens (DU *et al.*, 2020). Conforme a Figura 02, esta arquitetura é composta por duas partes, sendo a parte inicial denominada *Encoder* e a parte final denominada *Decoder*:

Figura 02 – Estrutura da arquitetura U-Net



Fonte: Ronneberger, Fischer e Brox (2015).

A Figura 02 exibe como a estrutura U-Net funciona. Através do *Encoder* a imagem original composta por uma matriz 572x572 é enviada para a arquitetura, onde são aplicadas camadas convolucionais de *Max Pooling,* com o objetivo de obter a imagem compactada, representada na parte central da estrutura. A imagem compactada é composta por um vetor de 1024 *pixels*. Na sequência, o *Decoder* realiza o processo inverso, aplicando camadas convolucionais de

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

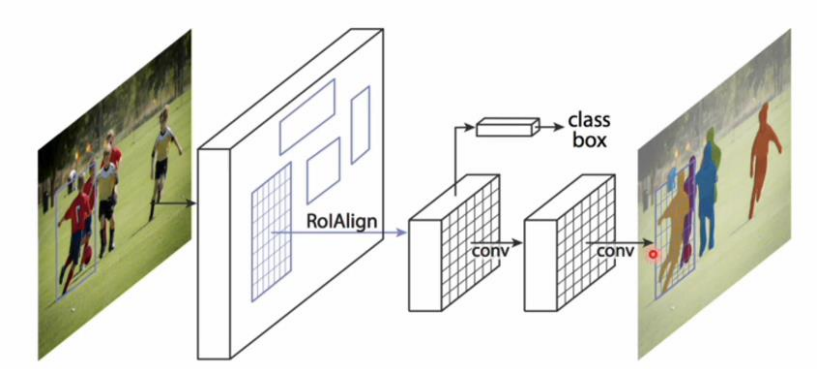
Upsampling. Basicamente o objetivo do processo de decodificação é projetar semanticamente as características discriminativas aprendidas da imagem pelo codificador. Ou seja, como se o decodificador validasse o que o codificador aprendeu (RONNEBERGER, FISCHER e BROX, 2015).

2.3 Arquitetura Mask R-CNN

A Mask R-CNN (*Mask Regional Convolutional Neural Network*) é uma arquitetura baseada em uma rede convolucional que foi introduzida em 2017 por cientistas de dados e pesquisadores do *Facebook AI research (FAIR).* A rede Mask R-CNN é basicamente uma extensão da popular arquitetura de detecção de objetos Faster R-CNN, por isso é conhecida por ser um modelo baseado na arquitetura original R-CNN (*R-CNN Baded Model*) (HE *et al*., 2017).

Na arquitetura Mask R-CNN, inicialmente uma imagem é enviada para que seja classificada e consequentemente segmentada. Na etapa seguinte, é realizada a extração das principais regiões da imagem onde há a probabilidade de obter o objeto desejado através de um *bounding box* (uma caixa delimitadora que extrai e contorna a área desejada). Posteriormente, diversas camadas convolucionais são aplicadas na região extraída, resultando na camada final da rede neural, que retorna a imagem segmentada com a classe do objeto (o nome do objeto segmentado), o valor de probabilidade de aquilo que foi selecionado, de fato, ser o objeto de interesse e finalmente, suas coordenadas delimitadas pelo *bounding box* (HE *et al*., 2017). A Figura 03 exibe a Arquitetura da rede:

Figura 03 – Arquitetura Mask R-CNN



Fonte: HE *et al.* (2017).

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

A Figura 03 exibe o fluxo utilizado pela arquitetura Mask R-CNN, onde a partir de uma imagem de entrada, diversas camadas convolucionais são utilizadas para obter o objeto desejado. Os principais componentes desta rede como o *Backbone*, FPN (Feature Pyramid Network), RPN (*Region Proposal Network*), Classificador de ROI (Region of Interest), ROI *Pooling* e Máscara de segmentação, serão descritos a seguir:

- **Rede Backbone:** basicamente uma rede neural convolucional padrão utilizada para fazer a extração das características de baixo nível como por exemplo as bordas e contornos da imagem, enquanto as camadas posteriores da rede neural detectam as características de maior nível, com por exemplo as características especificas do objeto desejado. Ao passar pela rede *Backbone*, a imagem original, tem suas dimensões originais convertidas para um mapa de características, que por sua vez, será repassado para os estágios seguintes (HU, HE e GUO, 2023).
- **Feature Pyramid Network (FPN):** De forma breve, o FPN é uma técnica para representar melhor os objetos em múltiplas escalas. Conforme o nome da arquitetura indica, através de uma pirâmide as características de alto nível são extraídas e passadas para uma nova camada, que por sua vez, realiza as previsões. Desta forma é possível que essas pirâmides tenham acesso as características de alto nível (HU, HE e GUO, 2023).
- ***Region Proposal Network* (RPN):** É utilizado um conceito chamado de janela deslizante, ou seja, existem diversos *bounding boxes* na imagem, de modo que o *bounding box* que possuir mais informações sobre o objeto de interesse e menos informações sobre o fundo da imagem, será utilizado pelo algoritmo (KAUSHIK, RAMAN e RAO, 2020).
- **Classificador de ROI**: Executado após o RPN este estágio gera duas saídas para cada ROI. A primeira saída é a classe a qual o objeto pertence, ou seja, se o objeto é uma pessoa, carro, cachorro, gato ou uma falha considerando a aplicação desenvolvida. A segunda saída é a segmentação do que de fato é o objeto na imagem e o que é o fundo da imagem. Além disso, o classificador retorna também a caixa delimitadora, responsável por indicar onde de fato esse objeto está na imagem (HE *et al*., 2017).
- ***ROI Pooling:*** Um pequeno problema nesta arquitetura é que os classificadores não interpretam com precisão as variações no tamanho da imagem de entrada, normalmente

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

eles requerem uma imagem de tamanho fixa. Durante a geração dos *bounding boxes* as regiões de interesse podem ter diferentes tamanhos, sendo necessário aplicar a técnica do *Roi Pooling,* que consiste em cortar uma parte do Mapa de Características e redimensioná-lo para um tamanho fixo adequado (HE *et al*., 2017).

- **Máscara de Segmentação**: Após o resultado de saída do Classificador de ROI, uma máscara de segmentação é aplicada, finalmente fornecendo as regiões que de fato, possuem o objeto desejado (HE *et al*., 2017).

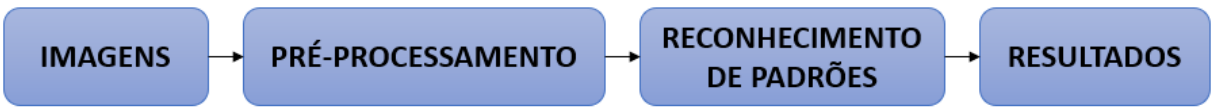
3. MATERIAIS E MÉTODOS

O trabalho proposto é baseado em experimentos com uso de ferramentas de IA. A natureza da pesquisa é secundária, trazendo informações através de trabalhos já publicados sobre o tema. A pesquisa é descritiva e busca dados mais consistentes sobre o tema do uso de IA aplicada a classificação de defeitos de qualidade em pneus. Busca-se oferecer subsídios para novas investigações sobre a mesma temática.

O desenvolvimento foi realizado utilizando uma base de dados com 60 imagens, na plataforma *Google Colaboratory*, que possibilita escrever e executar códigos em Python utilizando uma unidade de processamento gráfico gratuitamente, recurso fundamental durante o treinamento dos algoritmos. As imagens foram criadas com base no conteúdo do livro *The Pneumatic Tire* (GENT e WALTER, 2005) e validadas por um especialista de mercado.

Para um melhor entendimento do trabalho proposto, é apresentado abaixo na Figura 04 o fluxo macro baseado em Visão Computacional:

Figura 04 - Fluxo de um sistema baseado em Visão Computacional



Fonte: Autoria própria (2023).

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

É possível observar na Figura 04 que a primeira etapa de entrada são as imagens. Para as imagens de Entrada foram utilizadas imagens de Autoria própria. com defeitos induzidos para suportar a validação do projeto, e em seguida o pré-processamento e reconhecimento em Mask R-CNN e U-Net e a última etapa do processo que seriam os resultados obtidos em ambas as soluções.

3.1 Pré-processamento

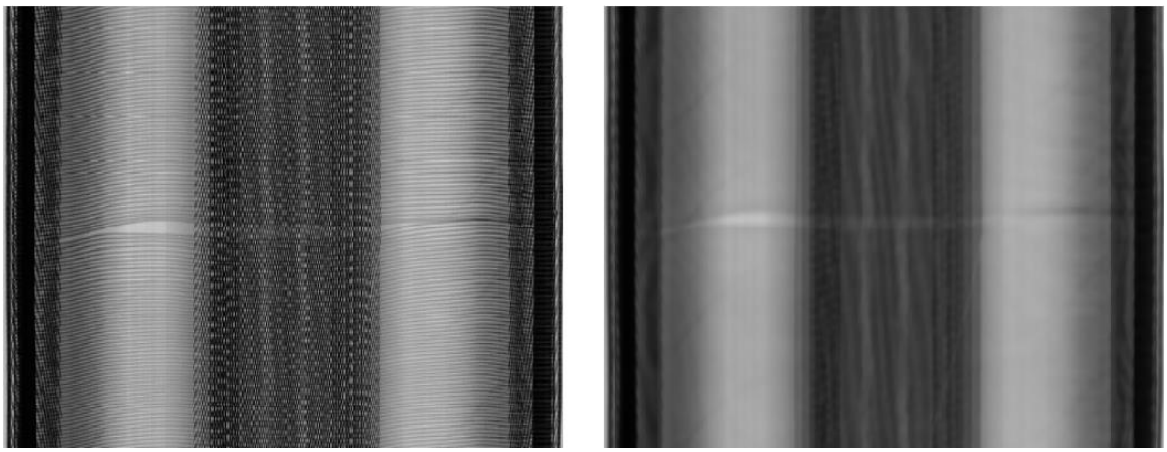
Utilizando técnicas de segmentação clássicas, com funções disponíveis na biblioteca para processamento de imagens OpenCV (OpenCV, 2023), uma biblioteca *open source*, foi possível segmentar o defeito e construir o conjunto de dados de validação. De modo experimental, inicialmente todas as imagens foram redimensionadas para o tamanho 1024 x 1024, através da função 'cv2.resize'. Se trata de uma adequação para um melhor desempenho dos algoritmos propostos.

Com as imagens padronizadas, as funções 'cv2.cvtColor' e 'cv2.COLOR_BGR2GRAY' foram utilizadas, para converter as cores RGB (*Red, Green e Blue*) para tons de cinza. Segundo Barelli (2018), a conversão para tons de cinza é uma técnica muito utilizada em visão computacional, pois permite que os algoritmos processem as imagens com mais eficiência, uma vez que as imagens em tons de cinza possuem menos informações para serem processadas, normalmente as características mais relevantes como bordas, manchas e junções. Além disso, a Limiarização (função utilizada posteriormente) obrigatoriamente requer que as imagens estejam em tons de cinza. Com a imagem convertida, o filtro *Gaussian Blur* foi utilizado, através da função 'cv2.GaussianBlur'. A finalidade deste filtro é reduzir os ruídos presentes na imagem, para obter um resultado melhor na etapa de Limiarização. Com o intuito de reforçar ou suprimir alguma característica, aumentando a nitidez ou desfocando a imagem, respectivamente, o filtro *Gaussian Blur* utiliza um *kernel* de convolução 2D, ou seja, uma matriz de duas dimensões que é sobreposta à imagem original com a finalidade de modificar o valor de cada *pixel* da imagem, pelo valor da média ponderada obtida na área definida pelo tamanho do kernel. Além do *kernel*, também é necessário especificar o parâmetro correspondente ao grau de suavização desejado. O filtro *Gaussian Blur* foi aplicado utilizando um *kernel* 17x17 e valor de suavização zero (correspondente a suavização mínima). A Figura 05 traz um comparativo entre a imagem original e a imagem obtida após a aplicação do filtro *Gaussian Blur*:

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

Figura 05 – Imagem original x Imagem com filtro *Gaussian Blur*

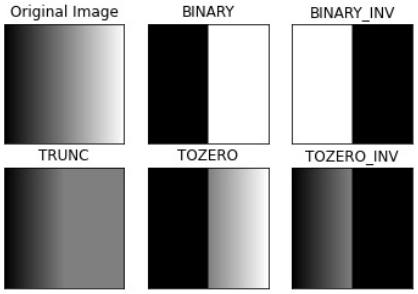


Fonte: Autoria própria (2023).

Com a comparação exibida na Figura 05, é possível observar que o conteúdo de alta frequência foi completamente suprimido, tornando a imagem mais suave.

Após a aplicação do filtro, a técnica de limiarização foi aplicada na imagem, utilizando a função 'cv2.adaptiveThreshold'. O conceito de limiarização, consiste em segmentar a imagem em tons de branco, preto e cinza, em função de um valor de referência previamente definido. Ou seja, cada *pixel* da imagem será comparado com o valor de limiar informado. Se o valor do *pixel* for menor do que o valor de limiar informado, o *pixel* terá o seu valor definido como zero. Caso o valor do *pixel* seja maior que o valor de limiar informado, o *pixel* terá o seu valor definido para o valor máximo, também informado previamente. A Figura 06 exibe alguns exemplos após a aplicação da Limiarização:

Figura 06 – Exemplos de Limiarização



ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

Fonte: OpenCV (2023)

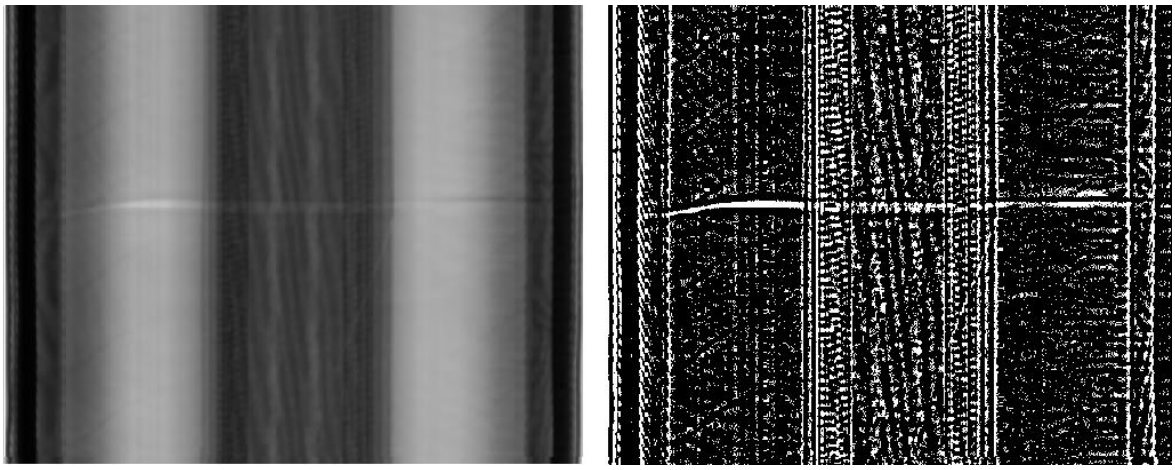
Para os exemplos da Figura 06, foram inseridos como parâmetros os valores de 127 para limiar e 255 para valor máximo. A imagem descrita '*Original Image*' como o próprio nome sugere, é a imagem original. A imagem '*BINARY*' é o resultado obtido onde todos os *pixels* com valores menores que 127 foram convertidos para o valor zero (preto) e todos os valores maiores que 127 foram convertidos para o valor máximo 255 (branco). A imagem '*BINARY_INV*' é o oposto do resultado obtido na imagem '*BINARY*'. A imagem '*TRUNC*' é o resultado onde todos os *pixels* com valores menores que 127 permanecem como estão e os *pixels* com valores maiores ou iguais a 127 são convertidos para o valor 127. A imagem '*TOZERO*' é o resultado obtido onde todos os *pixels* com valores menores que 127 são convertidos para zero (preto) e todos os *pixels* com valores maiores ou iguais a 127 ficam como estão. Finalmente, a imagem '*TOZERO_INV*' é o oposto do resultado anterior, onde todos os *pixels* com valores menores que 127 permanecem iguais e os *pixels* com valores maiores ou iguais a 127 são convertidos para zero (preto).

Existe também a Limiarização Adaptativa, que diferente do conceito anterior, não possui um valor fixo de limiar. Neste caso, o valor de limiar é dinâmico, sendo definido pela média do valor da área da vizinhança. Ou seja, o valor de cada *pixel* será determinado em função do valor médio de uma pequena região ao seu redor. Com esse método, é possível obter diferentes limiares, para diferentes regiões da imagem. Através do método experimental, observou-se que os resultados obtidos utilizando a Limiarização Adaptativa foram mais significativos. A Figura 07 exibe um antes e depois da aplicação da Limiarização Adaptativa:

Figura 07 – Antes e depois da Limiarização Adaptativa

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA



Fonte: Autoria própria (2023).

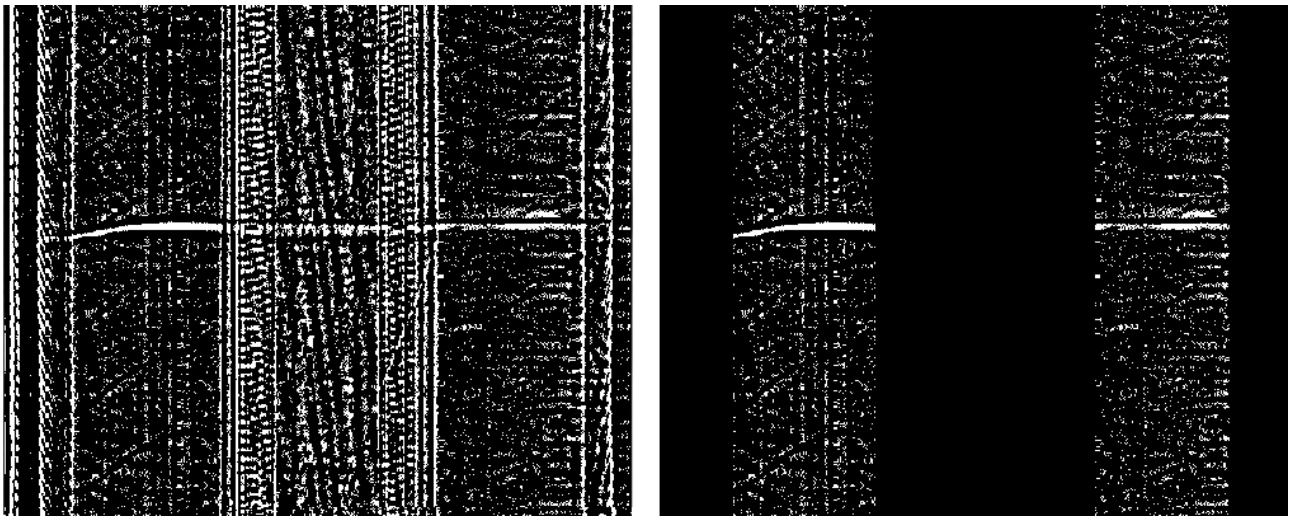
Através da Figura 07, é possível comparar a imagem de antes e depois do processo de limiarização. Após a limiarização, o defeito que desejamos segmentar está mais nítido, como sendo uma linha horizontal branca praticamente contínua. A limiarização foi aplicada através das funções ‘cv2.adaptiveThreshold’ e ‘cv2.THRESH_BINARY’.

Durante análise das imagens da base de dados, um padrão foi identificado. As imagens originais possuem três regiões mais escuras e duas regiões mais claras, sendo que os defeitos estão sempre presentes em pelo menos uma das duas regiões claras. Com o intuito de tornar a segmentação do defeito mais simples, removendo o excesso de informações na imagem, três retângulos pretos foram desenhados sobre as partes mais escuras da imagem, através da função ‘cv2.rectangle’, conforme a Figura 08:

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

Figura 08 – Antes e depois da remoção de excessos



Fonte: Autoria própria (2023).

Através do resultado exibido na Figura 08, é possível verificar que as regiões iniciais, centrais e finais, caracterizadas por linhas verticais fortes e bem definidas, foram eliminadas, facilitando a segmentação do defeito. Essas linhas, apesar de estarem em uma direção diferente (vertical, enquanto o defeito está na horizontal) dificultava o processo de segmentação.

Finalmente, a segmentação completa do defeito foi obtida aplicando um filtro de desfoque mediano, através da função 'cv2.medianBlur' e de outras duas funções que fazem parte de uma técnica conhecida como Operações Morfológicas, a função 'cv2.morphologyEX' e 'cv2.MORPH_OPEN'. O filtro de desfoque mediano, preserva os detalhes de alta frequência como bordas, contornos ou as linhas mais fortes da imagem. Por sua vez, as operações morfológicas costumam ser utilizadas em imagens binarizadas, ou seja, imagens em tons de preto e branco. As duas principais operações são a operação de dilatação e a operação de erosão. Utilizando a operação de dilatação, é possível realçar o objeto de interesse, por exemplo, aumentando as bordas do objeto ou então preenchendo alguns *pixels* que fazem parte do objeto, mas que por algum motivo não foram considerados. A operação de erosão faz o oposto. Com ela, é possível eliminar alguns ruídos que não fazem parte do objeto desejado ou então reduzir as bordas do objeto de interesse. A Figura 09 exibe algumas operações morfológicas:

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

Figura 09 – Operações morfológicas

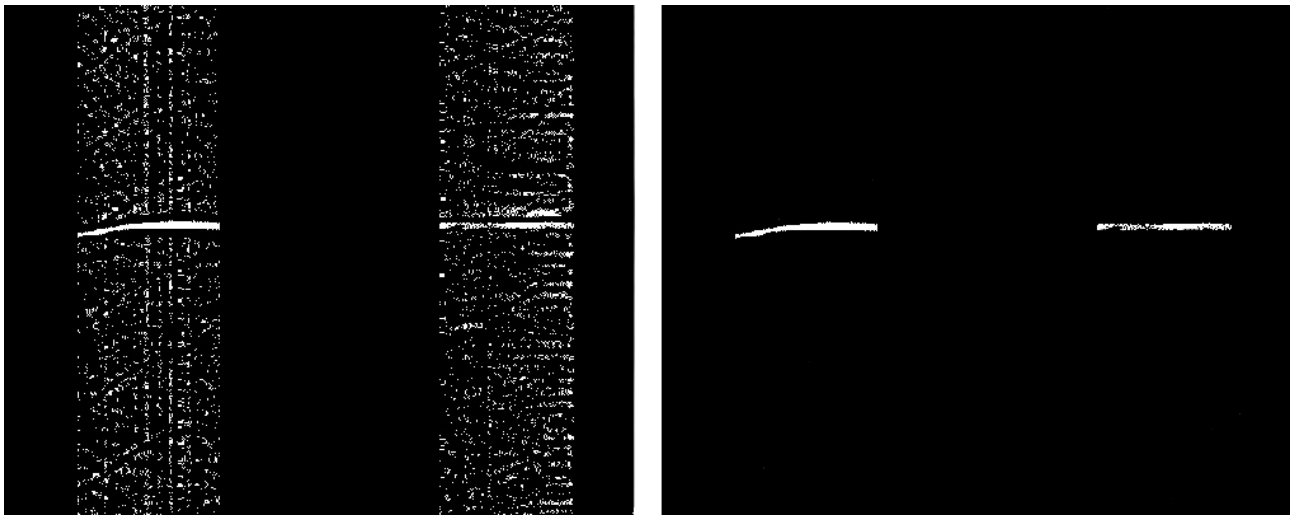
Fonte: OpenCV (2023)

A Figura 09 possui três operações morfológicas diferentes: à esquerda, é possível verificar a imagem original, referência para as imagens posteriores. Na segunda imagem da esquerda para a direita, é possível observar a operação de erosão, onde todas as bordas da imagem original foram reduzidas em função de um determinado *kernel.* A imagem central exibe a operação de dilatação, onde todas as bordas da imagem original foram ampliadas em função de um determinado *kernel*. Finalmente, a imagem à direita exibe a operação de abertura, que nada mais é do que a operação de erosão, seguida da operação de dilatação. O objetivo da operação de abertura é reduzir os ruídos da imagem. É possível verificar que os pequenos pontos brancos foram completamente suprimidos da imagem, tendo preservado o objeto desejado (letra j). Existe também a operação de fechamento, que tem como propósito o inverso da operação de abertura, ou seja, inicialmente é realizado a operação de dilatação e posteriormente a operação de erosão. A Figura 10 exibe a imagem original completamente segmentada:

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

Figura 10 – Imagem completamente segmentada



Fonte: Autoria própria (2023).

Através do resultado exibido na Figura 10, é possível verificar que o defeito objeto de interesse, foi completamente segmentado. Todas as 60 imagens disponíveis no conjunto de dados foram submetidas a essas etapas. O resultado obtido foi utilizado como conjunto de dados de validação na estrutura de Rede Convolucional U-Net.

3.2 Reconhecimentos de padrões com arquitetura U-Net

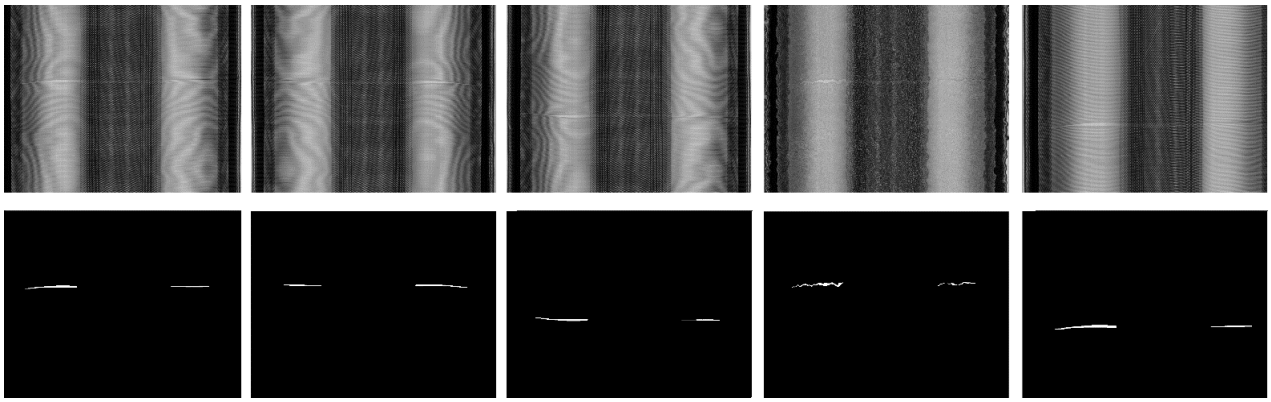
A estrutura U-Net foi desenvolvida utilizando as 60 imagens originais para o conjunto de treinamento e para o conjunto de validação as máscaras foram binarizadas, através do pré-processamento realizado. No entanto, para construir um modelo de rede neural satisfatório, é necessário ter uma base de dados consistente, com um número significativo de imagens. Em função da pequena quantidade de imagens disponíveis na base de dados, a biblioteca *Albumentation* foi importada e seus recursos foram utilizados, para gerar novas imagens a partir de transformações das imagens existentes. Portanto, das 60 imagens disponíveis, 40 imagens foram submetidas para as transformações '*HorizontalFlip*' que espelha a imagem horizontalmente, '*VerticalFlip*' que espelha a imagem verticalmente, '*ElasticTransform*' que gera distorções na estrutura da imagem produzindo um efeito de onda e '*GridDistortion*' que através de distorções na grade da imagem

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

produz o efeito de variações na distribuição ao longo da imagem. A Figura 11 exibe todas as distorções aplicadas.

Figura 11 – Transformações com recursos da biblioteca Albumentation



Fonte: Autoria própria (2023).

Através da Figura 11, é possível observar os resultados obtidos com as quatro transformações aplicadas, tanto nas imagens originais, quanto nas máscaras. Como esse recurso foi aplicado em 40 imagens, o resultado obtido foi de 200 imagens. Dessa forma o conjunto de treinamento foi definido com 200 imagens, o conjunto de validação com 200 máscaras e finalmente, o conjunto de testes com 20 imagens e 20 máscaras.

A estrutura da rede neural utilizada, como os blocos de convolução, os blocos de *encoder* e *decoder*, foi construída com os recursos da biblioteca *open source* TensorFlow. Quatro treinamentos com parâmetros diferentes foram realizados, dois treinamentos utilizando imagens de dimensões 1024x1024 (no primeiro enviando uma imagem por vez antes de ajustar os pesos do treinamento da rede e no segundo enviando duas imagens por vez) e dois treinamentos utilizando imagens de dimensões 512x512, inicialmente enviando uma imagem por vez e depois duas imagens por vez antes de ajustar os pesos (a função ‘cv2.resize’ novamente foi utilizada para utilizar converter as dimensões das imagens para 512x512.

3.3 Reconhecimentos de padrões com arquitetura Mask R-CNN

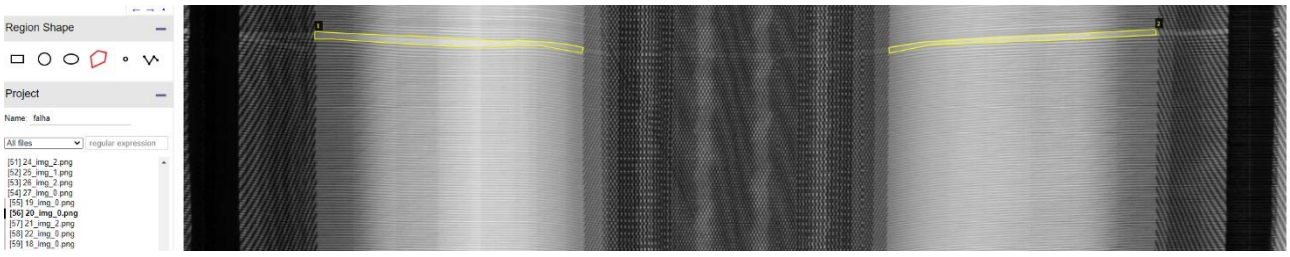
Para o desenvolvimento proposto, a estrutura da Mask R-CNN utilizada foi retirada de um repositório disponível no GitHub (MATTERPORT, 2023), sendo necessário apenas carregá-la no

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ambiente do Google Colaboratory e inserir os parâmetros correspondentes ao conjunto de imagens utilizado no presente trabalho. Além da estrutura da Mask R-CNN, também foi necessário importar a biblioteca do Tensorflow, uma das bibliotecas mais utilizadas para trabalhar com redes neurais e *deep learning*. Utilizou-se o conceito de transferência de aprendizagem, ou seja, uma técnica para não precisar realizar o treinamento da rede neural do zero. Dessa forma, para extrair características básicas como bordas e contornos foram utilizados os pesos de uma rede neural já existente. A rede em questão, foi criada utilizando um conjunto de dados da plataforma COCO (LIN *et al*., 2015). Com a base da rede neural pronta, algumas camadas convolucionais foram adicionadas para que fosse possível reconhecer as falhas desejadas.

O conjunto de dados de validação foi definido através da técnica de anotações, onde o algoritmo utiliza as coordenadas dos defeitos nos eixos X e Y como referência para realizar as previsões. As anotações foram geradas através da ferramenta de anotação de imagem VGG Image (VGG Image Annotator, 2023). Se trata de uma ferramenta muito utilizada para definir regiões de interesse em uma imagem e criar descrições textuais dessas regiões. A Figura 12 exibe uma anotação realizada em uma das imagens de treinamento.

Figura 12 – Anotação realizada com a ferramenta VIA.



Fonte: Autoria própria (2023).

Na Figura 12, é possível verificar que a ferramenta VIA possui algumas estruturas geométricas que permitem selecionar a imagem e contornar a região de interesse, atribuindo posteriormente, um nome a essa região. A região amarela são as falhas que devem ser segmentadas e consequentemente classificadas pelo algoritmo.

Após carregar todas as imagens do conjunto de dados de treinamento e realizar a anotação individual de cada uma delas, um arquivo na extensão ‘json’ é gerado, contendo todas as coordenadas X e Y das regiões que foram selecionadas. A Figura 13 exibe o arquivo em questão.

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

Figura 13 – Arquivo ‘json’ com as coordenadas X e Y

via_region_data.json


```
{"_via_settings":{"ui":{"annotation_editor_height":25,"annotation_editor_fontsize":0.8,"leftsidebar_width":18,"image_
"05_img_1.png8545154":{"filename":"05_img_1.png","size":8545154,"regions":[{"shape_attributes":{"name":"polygon","all_
"file_attributes":{"caption":"","public_domain":"no","image_url":""}},"08_img_2.png7254070":{"filename":"08_img_2.png"
[840,840,852,897,927,960,1000,965,937,908,871,840],"all_points_y":[266,275,273,271,269,267,264,265,265,265,265,267]},"
965,985,970,933,881],"all_points_y":[1527,1534,1535,1539,1540,1537,1537,1536,1533,1532]},"region_attributes":{"name":"
1887,1889,1891,1892,1892,1892,1891,1892,1896,1903,1896,1887,1885,1886]},"region_attributes":{"name":"falha","type":"un
"image_quality":{"good":true,"frontal":true,"good_illumination":true}}},{"shape_attributes":{"name":"polygon","all_poi
{"name":"polygon","all_points_x":[149,148,169,206,239,282,315,367,394,425,452,464,469,455,444,426,420,411,326],"all_po
970,966]},"region_attributes":{"name":"falha","type":"unknown","image_quality":{"good":true,"frontal":true,"good_illum
"image_quality":{"good":true,"frontal":true,"good_illumination":true}}},{"shape_attributes":{"name":"polygon","all_poi
1011,968,894,856,857,899,968,1030],"all_points_y":[1529,1537,1536,1535,1537,1541,1546,1539,1534,1532,1528]},"region_at
"good_illumination":true}}},{"shape_attributes":{"name":"polygon","all_points_x":[798,798,816,881,937,1003,1040,1091,1
```


Fonte: Autoria própria (2023).

As coordenadas exibidas na Figura 13, corresponde a cada ponto nos eixos X e Y que foram selecionados durante o processo de anotação. Essas coordenadas são utilizadas como referência, durante o aprendizado da rede. Isso significa que o arquivo ‘json’ é a base de validação. Além da anotação na base de dados de treinamento, também é necessário anotar a base de dados de testes, pois é através da anotação na base de dados de teste que será possível comparar o resultado previsto pela rede neural com as respostas reais, anotadas no VIA. É dessa forma que o desempenho do algoritmo é avaliado.

Foram utilizadas 120 imagens com 30 épocas de treinamento do algoritmo dando peso para cada uma das imagens em cada passagem por cada imagem.

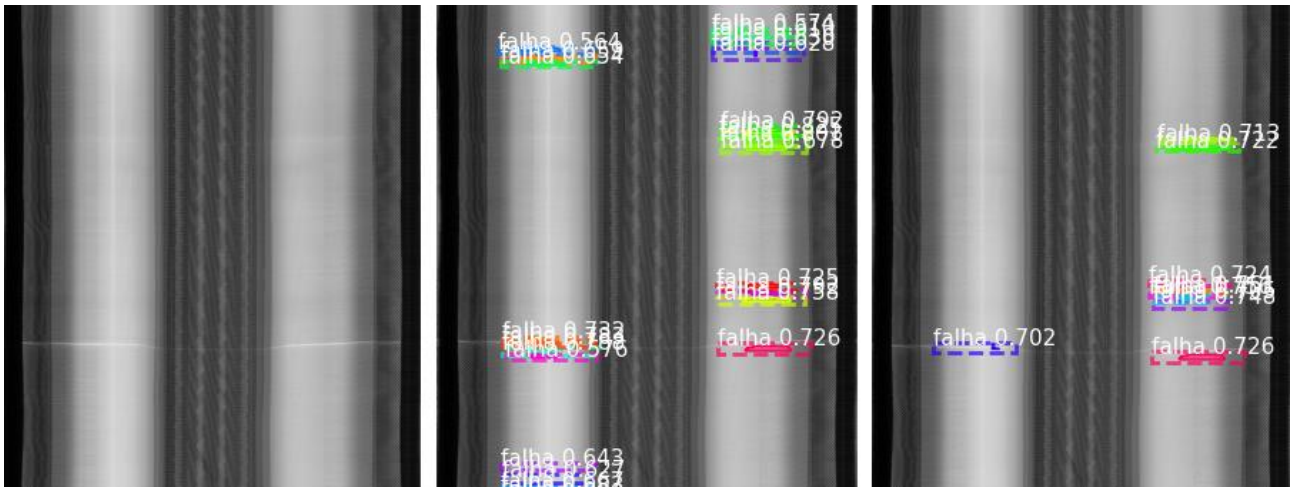
3.4 Resultados

Inicialmente, o primeiro algoritmo utilizando a arquitetura Mask R-CNN foi configurado com imagens de dimensões 1024x1024, porém, não foi possível concluir o treinamento pois o ambiente do Google Colaboratory sempre retornava um erro na parte final, possivelmente relacionado as dimensões das imagens. Sendo assim, todas as imagens foram redimensionadas para as dimensões 512x512 e enviadas para serem processadas pela GPU (*Graphics Processing Unit*) uma por uma, em um treinamento de 30 épocas, com percentual mínimo de detecções aceitáveis de 90%. No entanto, em função da pequena quantidade de imagens utilizadas, o algoritmo não foi capaz de identificar falhas. Para contornar este problema, alguns testes foram realizados, variando

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

o valor de percentual mínimo de detecções aceitáveis para 50% e posteriormente 70%, onde o melhor resultado foi obtido. A Figura 14 exibe os resultados dos três algoritmos:

Figura 14 – Resultados obtidos com arquitetura Mask R-CNN



Fonte: Autoria própria (2023).

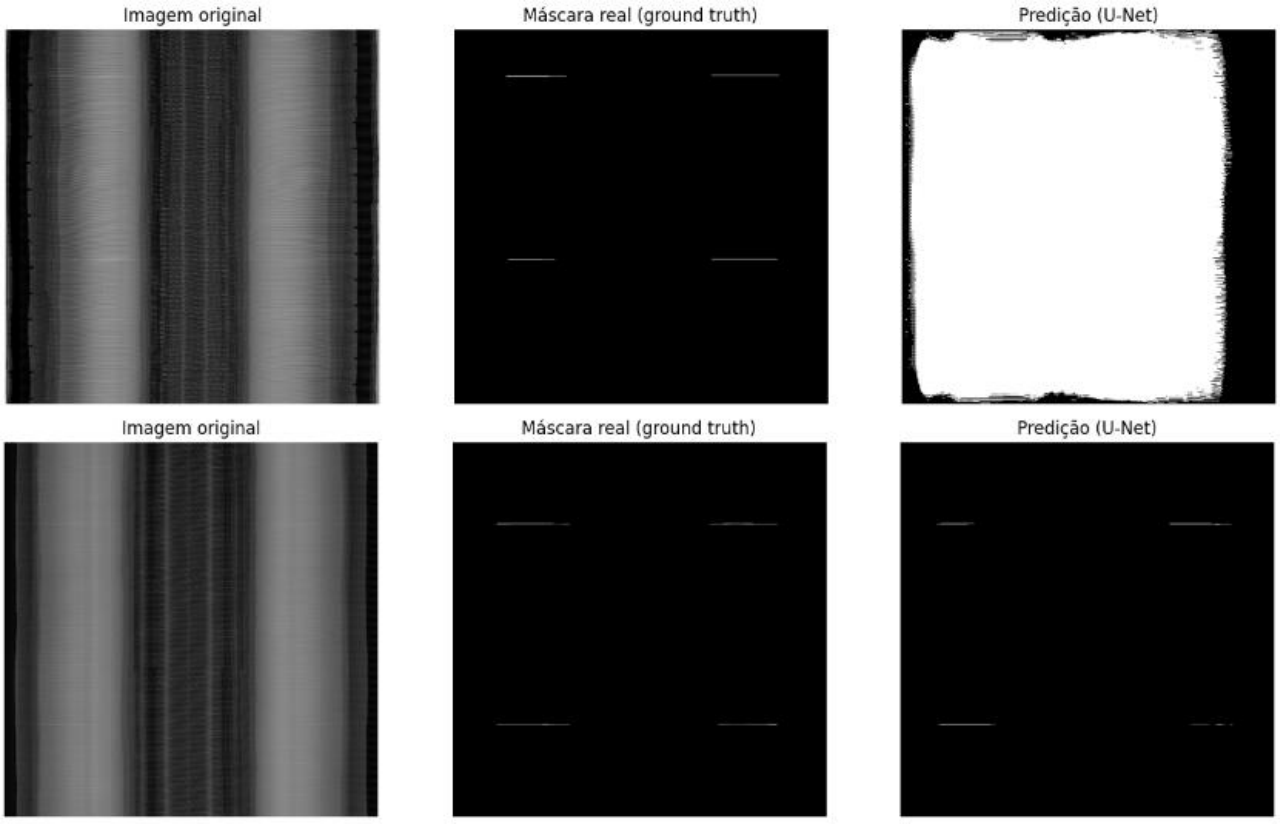
Através da Figura 14, é possível observar que variando o percentual mínimo de detecções aceitáveis, foi possível obter alguns resultados uma vez que a imagem da esquerda foi configurada com um valor alto de no mínimo 90% de detecções aceitáveis, a imagem central com um valor baixo de 50% e finalmente a imagem a direita com um valor moderado de 70%.

Em relação aos quatro treinamentos utilizando a arquitetura U-Net, foi possível observar que quanto maior as dimensões da imagem e menor o número de imagens simultâneas enviadas para serem processadas pela GPU, melhor é o resultado obtido. Na Figura 15 é possível comparar o resultado do pior algoritmo obtido (com imagens de dimensões 512x512 sendo enviadas de duas em duas para a GPU) com o resultado do melhor algoritmo obtido (imagens de dimensões 1024x1024 sendo enviadas uma por uma para a GPU):

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

ALEX DA SILVA ALVES
MICHAEL SAMPAIO DOS SANTOS
GEDEANE KENSHIMA

Figura 15 – Resultados obtidos com arquitetura U-Net



Fonte: Autoria própria (2023).

Ao analisar a Figura 15 é possível observar que a imagem inferior apresentou uma predição muito próxima de sua Máscara real, ao contrário da imagem superior que foi configurada com imagens menores e com uma menor utilização da GPU.

Ambos os modelos foram capazes de obter resultados, no entanto, existem diferenças significativas entre eles. Enquanto o modelo utilizando a arquitetura Mask R-CNN é capaz de segmentar a falha utilizando um *bounding box* e classificá-la com um valor de probabilidade, o

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

modelo utilizando a arquitetura U-Net é mais simples ao exibir apenas a falha binarizada. Apesar da simplicidade, o modelo U-Net se mostrou mais preciso, pois ao segmentar a falha chegou muito mais próximo do valor esperado. Esse resultado foi obtido em função de um maior número de imagens nos conjuntos de treinamento e validação e das dimensões das imagens, uma vez que foi possível utilizar imagens de dimensões 1024x1024. Cabe ressaltar que o treinamento dentro da Mask R-CNN foi realizado com 30 épocas e o da arquitetura U-Net com 20 épocas.

Além disso, o ideal é executá-los em um ambiente de desenvolvimento robusto, com hardware capaz de atender todas as demandas inerentes ao processamento de imagens, pois, apesar de o ambiente gratuito do Google Colaboratory oferecer uma unidade de processamento gráfico gratuita, muitas vezes o treinamento do algoritmo foi interrompido em função da quantidade de épocas desejadas e das dimensões das imagens

4 CONCLUSÃO

Este projeto possibilitou a obtenção de dois algoritmos desenvolvidos com técnicas de segmentação clássica e recursos de Inteligência Artificial capazes de segmentar e classificar falhas de qualidade de pneus radiais metálicos presentes em imagens de Raio-X.

Através dos resultados obtidos entre as duas arquiteturas, pode-se concluir que para uma aplicação real utilizando a arquitetura Mask R-CNN, alguns ajustes precisam ser realizados, pois no Mask R-CNN, para aumentar a acurácia, os algoritmos precisam de um número maior de imagens no conjunto de treinamento e consequentemente no conjunto de validação. Já a arquitetura U-Net mostrou-se mais precisa, pois ao segmentar a falha, tratando-a em forma binarizada, chegou muito mais próximo do valor esperado. Esse resultado foi obtido na arquitetura U-Net em função de um maior número de imagens nos conjuntos de treinamento e validação e das dimensões das imagens, uma vez que foi possível utilizar imagens de dimensões 1024x1024.

Para trabalhos futuros, sugere-se explorar a arquitetura U-Net, pois a falha não possui como característica um padrão bem definido para ser explorado, de modo que o processo de binarização seja mais relevante do que o processo de anotação. O mesmo método também pode ser aplicado e avaliado em outros processos de identificação de defeitos visuais na indústria.

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

REFERÊNCIAS

BARELLI, F. **Introdução à Visão Computacional: Uma abordagem prática com Python e OpenCV**. São Paulo: Casa do Código, 2018.

DU, Getao; CAO, Xu; LIANG, Jimin; CHEN, Xueli; ZHAN, Yonghua. **Medical Image Segmentation based on U-Net: A Review**. Journal of Imaging Science and Technology. 2020. p. 1-12.

GENT, A. N; WALTER, J. D. **The Pneumatic Tire**. Mechanical Engineering Faculty Research: Washington D.C., 2005..

HE, Kaiming; GKIOXARI, Georgia; DOLLAR, Piotr; GIRSHICK, Ross. Mask R-CNN. In: **Proceedings of the IEEE international conference on computer vision**. arXiv. 2017. p. 2961-2969.

HU, Jingwei; HE, Jing; and GUO, Chengjun. **End-to-End Powerline Detection Based on Images from UAVs**. Remote Sensing. 2023. p. 01-18.

KAUSHIK, S; RAMAN, Abhishek; RAO, Rajeswara K.V.S. **Leveraging Computer Vision For Emergency Vehicle Detection1Implementation And Analysis**. In: 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT). 2020. p. 1-6.

LI, Pengcheng; DONG, Zihao; SHI, Jianjie; PANG, Zengzhi; LI, Jinping. **Detection of Small Size Defects in Belt Layer of Radial Tire Based on Improved Faster R-CNN**. In: 11th International Conference on Information Science and Technology (ICIST). 2021. p. 531-538.

LIN, Tsung-Yi; MAIRE, Michael; BELONGIE, Serge; BOURDEV, Lubomir; GIRSHICK, Ross; HAYS, James; PERONA, Pietro; RAMANAN, Deva; ZITNICK, C; LAWRENCE, DOLLÁR Piotr. **Microsoft COCO: Common objects in context.** In: 13th European Conference. 2015. p. 1-15.

MALEKI, Farhad; MUTHUKRISHNAN, Nikesh; OVENS, Katie; REINHOLD, Caroline; FORGHANI, Reza. **Machine Learning Algorithm Validation From Essencials to Advanced Applications and Implications for Regulatory Certification and Deployment**. Neuroimaging Clinics of North America, v.30, 2020. p. 433–445

MATTERPORT, INC. **GitHub - matterport/Masck_RCNN: Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow.** Disponível em: <https://github.com/matterport/Mask_RCNN#readme>
Acesso em 30-mai-2023.

OpenCV. **Open Source Computer Vision.** Disponível em: <https://docs.opencv.org/>

ISSN 1983-1838
(DOI): 10.18624/etech.v17i1.1262

Acesso em 30-Maio-2023

PAHINKAR, Ajinkya; MOHAN, Prajval; MANDAL, Ankita; KRISHNAMOORTHY, A. **Faster Region Based Convolutional Neural Network and VGG 16 for Multi-Class Tyre Defect Detection**. In: 2th International Conference on Computing Communication and Networking Technologies (ICCCNT). 2021. p. 01-07.

RONNEBERGER, Olaf; FISCHER, Philipp; BROX, Thomas. **U-Net: Convolutional networks for biomedical image segmentation**. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings:. Springer International Publishing, 2015. p. 234-241.

SALEH, Radhwan A. A.; KONYAR, Mehmet Zeki; KAPLAN, Kaplan; ERTUNC, H. Metin. **Tire Defect Detection Model Using Machine Learning**. In: 2nd International Conference on Emerging Smart Technologies and Applications (eSmarTA). 2022. p. 01-05.

VGG Image Annotator (VIA). **VGC Image Annotator.** Disponível em: <https://www.robots.ox.ac.uk/~vgg/software/via/via.html>
Acesso em 30-Maio-2023.

ZHENG, Xiunan; DING, Jianpei; PANG, Zengzhi; LI, Jinping. **Detection of Impurity and Bubble Defects in Tire X-Ray Image Based on Improved Extremum Filter and Locally Adaptive-threshold Binaryzation**. In: 2018 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC). 2018. p. 360-365.