DOI: 10.48011/asba.v2i1.1537

APLICAÇÃO DO DEEP LEARNING PARA ANÁLISE DE FISSURAS EM TESTES DE QUEDAS DE PELOTAS

MARCONI J. H. MAGNANI[1], THYAGO R. SOUZA[1], JORGE J. F. FILHO[1], MARCO A. D. S. L. CUADROS[1]

1. *Programa de Pós-Graduação em Automação e Controle, Instituto Federal do Espírito Santo ES-010, Km-6,5 – Manguinhos, 29173-087, Serra-ES, Brasil.(e-mails: marconi@in9automacao.com.br,thyago@in9automacao.com.br,jorge@in9automacao.com.br,marcoantonio@ifes.edu.br)*

Abstract: Iron ore pellets are a prime input for iron production. Therefore there is a need for a rigorous control of the quality of the pellets to apply them in the industrial process. The pellets are degraded due to impacts caused by their handling or transport systems. As a result of these degradations many pellet shipments reach the customer with a proportion of cracks. Laboratory drop test trials are required on wet raw pellets to assess their resistance to the various drops they suffer in the industrial process. Currently the drop test is performed manually, where the whole test process, from pellet manipulation and data collection, depends on human action. The present work aims at the application of Deep Learning to carry out the analysis of pellet cracks, pellet segmentation is initially presented in this article.

Resumo: As pelotas de minério de ferro são um insumo nobre na produção de ferro. Por tanto tem uma necessidade de um controle rigoroso da qualidade das pelotas para aplicação das mesmas no processo industrial. As pelotas sofrem degradações devido aos impactos provocados pelos seus sistemas de manuseio ou transporte. Como resultado dessas degradações muitos carregamentos de pelotas chegam ao cliente com uma proporção de fissuras. São necessários ensaios de testes de queda laboratoriais realizados em pelotas cruas úmidas para avaliação de sua resistência às diversas quedas que as mesmas sofrem no processo industrial. Atualmente o ensaio do teste de queda é realizado de forma manual, onde todo o processo do teste, desde a manipulação das pelotas e obtenção dos dados, depende de uma ação humana. O presente trabalho tem como objetivo a aplicação de aprendizagem profunda para realizar a análise das fissuras das pelotas, sendo apresentado inicialmente neste artigo a segmentação da pelota.

Keywords: Pellets; Industrial Process; Drop Test; Deep Learning; Pellet Cracks;

Palavras-chaves: Pelotas; Processo Industrial; Teste de Queda; Aprendizagem Profunda; Fissuras das Pelotas.

1. INTRODUÇÃO

Atualmente carros, caminhões, bicicletas, aviões, eletrodomésticos e grande parte dos produtos que utilizamos em nosso dia a dia são feitos a partir do aço produzido nas siderúrgicas. Para a sua obtenção, uma importante matéria-prima utilizada é a pelota. A pelotização é um processo industrial responsável pela produção de pelotas.

A pelota de minério de ferro é um dos principais insumos na etapa de produção de ferro primário dentro da rota de produção do aço, pois apresenta características físicas, químicas e metalúrgicas mais favoráveis às operações de redução se comparada a outras matérias-primas como o minério de ferro granulado e o sínter. Dentre estas características podem-se citar a distribuição de tamanho mais estreita, a elevada porosidade, a baixa perda por ignição, o teor mais elevado de ferro, entre outras (Meyer, 1980).

É necessário um controle rigoroso quanto a qualidade das pelotas, um método de controle utilizado é o ensaio do teste de queda *(drop teste)*, realizado em pelotas cruas úmidas, este teste permite avaliar a resistência das pelotas cruas úmidas às diversas quedas que as mesmas sofrem desde o disco de pelotamento até o forno de grelha móvel.

Cada pelota é solta individualmente de uma altura de aproximadamente 45cm várias vezes até que a mesma apresente alguma trinca. Assim, o número de quedas que a pelota suportou será o valor da resiliência (capacidade de voltar ao estado natural, após uma situação fora do comum). Após vários testes é reportado o valor médio como resultado.

Atualmente o ensaio do teste de queda é realizado de forma manual, onde todo o processo do teste, desde a manipulação das pelotas e obtenção dos dados, depende de uma ação humana.

As redes neurais e as técnicas de aprendizado profundo estão sendo muito utilizadas nas mais diversas áreas e seguimentos

como, por exemplo na medicina onde são utilizadas para determinação de doenças nos rins analisando as mais diversas características desse órgão (Raju, Rao, & Rao, 2018).

Nos últimos anos, o *Deep Learning* revolucionou o campo do aprendizado de máquina, para visão computacional em especial. Nessa abordagem, uma rede neural artificial (RNA) profunda (multicamada) é treinada, geralmente numa maneira supervisionada usando retropropagação. São necessárias grandes quantidades de exemplos de treinamento rotulados, mas a precisão da classificação resultante é realmente impressionante, às vezes superando os humanos(Tavanaei,Ghodrati,Kheradpisheh, Masquelier, & Maida, 2019). Inúmeros trabalhos relacionados com a detecção de utilizando o Deep Learning podem ser encontradas na literatura, em (Duan, Liu, Wu, & Mao, 2019) foi projetado uma rede leve de aprendizado profundo da rede U-net para detectar automaticamente pelotas de imagens e obter os mapas de probabilidade de contornos de pelotas, foram utilizadas técnicas de aprendizado profundo na medicina para determinação de doenças nos rins analisando as mais diversas características desse órgão (Raju et al., 2018).

Neste artigo se propõe a utilização de técnicas de Inteligência Artificial para um projeto de inovação tecnológica, onde será aplicada uma rede convolucional profunda para análise e segmentação das pelotas.

O presente artigo tem como objetivo principal a aplicação do Deep Learning em um protótipo autônomo para o teste de queda, para análise das fissuras das pelotas.

Este artigo está dividido em 5 capítulos. Este, o capítulo 1, que contextualiza o problema a ser resolvido, expõe a justificativa e importância do trabalho. O capítulo 2 referente ao processo de pelotização das pelotas e o teste de qualidade chamado de *(drop teste)*, um teste realizado manualmente com objetivo de verificar o número de quedas que a pelota suporta. O capítulo 3 que apresenta o referencial teórico que embasa essa pesquisa. O capítulo 4, que explica o desenvolvimento do trabalho e finaliza com capítulo 5 com a conclusão do trabalho.

2. PELOTIZAÇÃO

Pontes, carros, aviões, bicicletas, eletrodomésticos e grande parte dos produtos que utilizamos em nosso dia a dia são feitos a partir do aço produzido nas siderúrgicas. Para a sua obtenção, uma importante matéria-prima utilizada é a pelota, cuja transformação, a partir de frações ultrafinas de minério de ferro, se dá em nossas usinas de pelotização (VALE, 2014).

O sistema de produção destas pequenas esferas começa com a extração de minério de ferro em Minas Gerais. O fino do minério de ferro, denominado pellet-feed, chega aos pátios das unidades produtoras, vindo das minas. Nos pátios são formadas pilhas que, posteriormente, são recuperadas e transportadas em correias para o processo da moagem. Paralelamente, os pátios recebem insumos, como o calcário, que será adicionado ao minério. Na moagem, o minério é moído com água, formando uma polpa classificada por hidrociclones (equipamento para separação de sólido e líquido) e enviada para o espessador, onde é sedimentada e, em seguida, encaminhada para tanques homogeneizadores (VALE, 2014), conforme Figura 1.

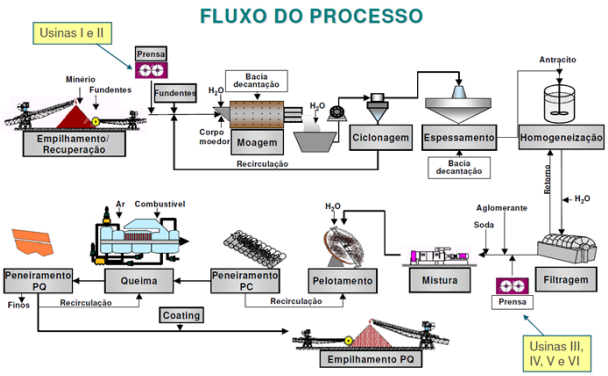

Fig. 1 Fluxo do Processo de Pelotização.

A qualidade das pelotas é verificada através de diversos testes, e um deles é o teste de queda *(drop teste)*, realizado em pelotas cruas úmidas, este teste permite avaliar a resistência das pelotas cruas úmidas ás diversas quedas que as mesmas sofrem do disco de pelotamento a grelha. Cada pelota é solta individualmente de uma altura de aproximadamente 45cm várias vezes.

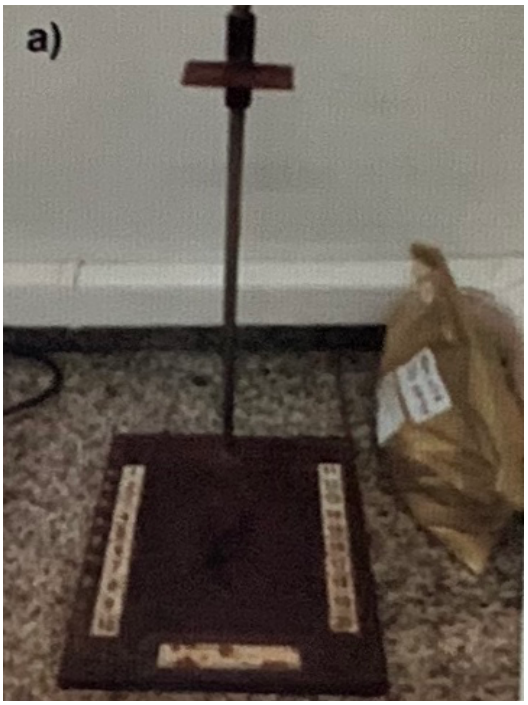

Fig. 2 Teste de Queda.

Na Figura 2 pode-se observar o equipamento usado para o teste, a altura correta é indicada numa haste, desde onde o operador solta a pelota, que caí numa base de metal, posteriormente o operador pega a pelota com as mão e faz uma verificação visual, caso não exista nenhuma trinca, volta a soltar a pelota, este procedimento é realizado da mesma forma até que a pelota apresente alguma trinca. Assim, o número de quedas que a pelota suportou será o valor da resiliência

(capacidade de voltar ao estado natural, após uma situação fora do comum). Após vários testes é reportado o valor médio como resultado.

3. SISTEMA INTELIGENTE

3.1 Redes Neurais

As Redes Neurais Artificiais são sistemas simultâneos compostos por unidades de processamento simples, neurônios, que calculam determinadas funções matemáticas, lineares ou não-lineares (Braga, Ludemir, & Carvalho, 2000).

Enquanto os outros métodos utilizam probabilidade e estatística, os algoritmos de redes neurais buscam imitar as estruturas do cérebro humano (SUGOMORI et al., 2017).

3.2 Rede Neural Convolucional

No contexto de redes neurais profundas, as redes convolucionais vem se apresentando muito eficiente nas aplicações de processamento de imagens. Segundo (Ponti, 2017), os métodos de aprendizagem profundo são hoje o estado da arte em muitos problemas de aprendizagem de máquina, em particular nos problemas de classificação.

Este método de aprendizagem profunda tem ganhando muito espaço em função da grande disponibilidade de dados a custos mais baixos, e também devido à capacidade computacional de processamento destas informações em menor tempo.

Ainda segundo (Ponti, 2017), as CNNs (Redes Neurais Convolucionais, em inglês), são os modelos de redes Deep Learning mais conhecidos e utilizados atualmente. O que caracteriza este tipo de rede é que são colocadas camadas convolucionais, as quais processam as entradas considerando campos locais, dentre outros artifícios, como o pooling, que reduz a dimensão espacial das entradas. A Figura 3 mostra a aplicação de convolução em regiões de uma imagem de entrada.

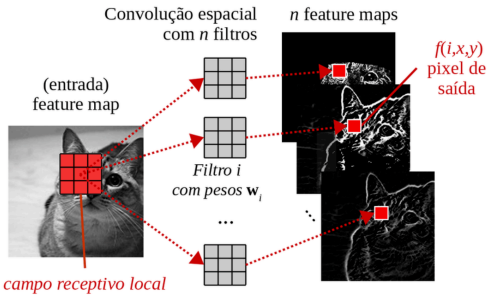

Fig. 3 Convolução Espacial.

Na camada convolucional cada neurônio é um filtro aplicado a uma imagem de entrada e cada filtro é uma matriz de pesos.

Já na Figura 4, é possível observar a aplicação de duas camadas de convolução em uma imagem RGB, ou seja, imagem com três dimensões. Na primeira camada são aplicados 4 filtros 5x5x3, produzindo 4 mapas de características, e em seguida, uma outra camada convolucional com 5 filtros 3x3x4, geram novos mapas de características, conforme detalhado por (Ponti, 2017).

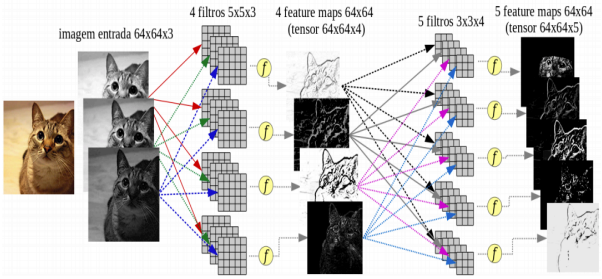

Fig. 4 CNN com duas camadas convolucionais.

É possível observar também as funções de ativação, em círculos, entre as camadas. É importante citar as limitações nas redes profundas, e uma delas citadas por (Ponti, 2017), se trata da necessidade de grande quantidade de dados rotulados para aprender conceitos simples. Em contrapartida, seres humanos são capazes de compreender um conceito a partir de pequena quantidade de exemplos.

4. DESENVOLVIMENTO

Para desenvolvimento do sistema foi criado um data set do teste de queda de pelotas para o treinamento da rede de aprendizagem profunda, ou seja, foram obtidas imagens das pelotas no andamento dos testes para criar o data set, inicialmente foram utilizadas aproximadamente 50 figuras do teste de queda, algumas tinham fraturas e outras não, foram utilizadas técnicas para tratamento das imagens conforme Figura 5 para treinamento da rede.

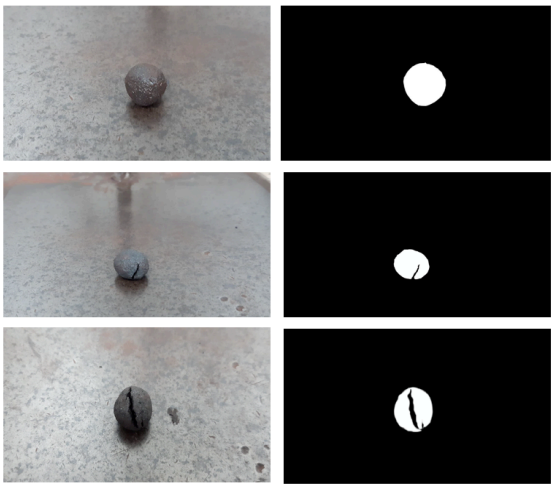

Fig. 5 Tratamento das imagens.

Foi projetada uma rede leve de aprendizado profundo conforme Figura 6.

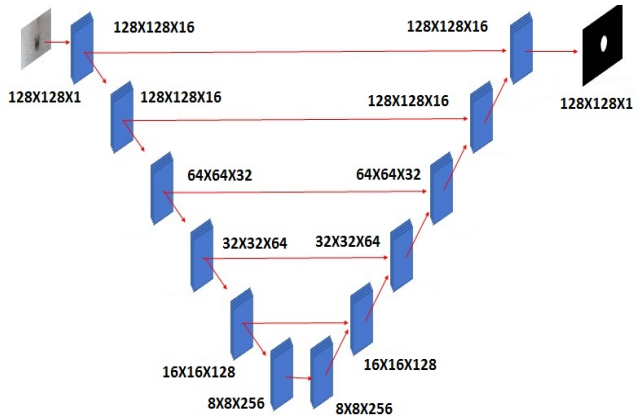

Fig. 6 Estrutura da rede utilizada.

Para classificação das pelotas foi utilizado uma rede leve de aprendizado profundo da rede U-net para detectar automaticamente pelotas de imagens e obter os mapas de probabilidade de contornos das pelotas conforme Figura 7, Figura 8 e Figura 9.

Artigo - APLICAÇÃO DO DEEP LEARNING PARA ANÁLISE DE FISSURAS EM TESTES DE QUEDAS DE PELOTAS

Mestrado em Engenharia de Controle e Automação

Aluno: Marconi Junio Henriques Magnani

```
[ ]  1 from google.colab import drive
     2 drive.mount('/content/drive')

Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8q

Enter your authorization code:
·····
Mounted at /content/drive
```

```
[ ]  1 import os
     2 import cv2
     3 import numpy as np
     4 import matplotlib.pyplot as plt
     5 from numpy import expand_dims
     6 import random
     7 import pandas as pd
     8 import numpy as np
     9 import matplotlib.pyplot as plt
    10 plt.style.use("ggplot")
    11 %matplotlib inline
    12
    13 from tqdm import tqdm_notebook, tnrange
    14 from itertools import chain
    15 from skimage.io import imread, imshow, concatenate_images
```

Fig. 7 Código do programa.

A Figura 7 representa o código do programa, a ferramenta utilizada para desenvolvimento do programa foi o *Google Colaboratory,* conhecida comercialmente como *Google* Colab, que é um ambiente de programação executado na nuvem, essa ferramenta permite escrita e execução de códigos em Python na versão 3.7 gratuitamente diretamente do seu navegador de internet. O programa foi feito em blocos o quais são possíveis testar unitariamente e segmentar cada etapa do programa. Algumas bibliotecas foram usadas tais como keras, pandas, numpy, matplotlib, open cv, itertools, skimage e a biblioteca os que é a biblioteca que permite acessar os arquivos alocados no sistema operacional. Os arquivos de imagens de pelotas foram arquivados na nuvem, utilizando o Google Drive, assim como as máscaras que foram criadas e usadas no processo de treinamento da rede.

```
[ ]  1 def get_unet(input_img, n_filters=16, dropout=0.5, batchnorm=True):
     2     # contracting path
     3     c1 = conv2d_block(input_img, n_filters=n_filters*1, kernel_size=3, batchnorm=batchnorm)
     4     p1 = MaxPooling2D((2, 2)) (c1)
     5     p1 = Dropout(dropout*0.5)(p1)
     6
     7     c2 = conv2d_block(p1, n_filters=n_filters*2, kernel_size=3, batchnorm=batchnorm)
     8     p2 = MaxPooling2D((2, 2)) (c2)
     9     p2 = Dropout(dropout)(p2)
    10
    11     c3 = conv2d_block(p2, n_filters=n_filters*4, kernel_size=3, batchnorm=batchnorm)
    12     p3 = MaxPooling2D((2, 2)) (c3)
    13     p3 = Dropout(dropout)(p3)
    14
    15     c4 = conv2d_block(p3, n_filters=n_filters*8, kernel_size=3, batchnorm=batchnorm)
    16     p4 = MaxPooling2D(pool_size=(2, 2)) (c4)
    17     p4 = Dropout(dropout)(p4)
    18
    19     c5 = conv2d_block(p4, n_filters=n_filters*16, kernel_size=3, batchnorm=batchnorm)
    20
    21     # expansive path
    22     u6 = Conv2DTranspose(n_filters*8, (3, 3), strides=(2, 2), padding='same') (c5)
    23     u6 = concatenate([u6, c4])
    24     u6 = Dropout(dropout)(u6)
    25     c6 = conv2d_block(u6, n_filters=n_filters*8, kernel_size=3, batchnorm=batchnorm)
    26
    27     u7 = Conv2DTranspose(n_filters*4, (3, 3), strides=(2, 2), padding='same') (c6)
    28     u7 = concatenate([u7, c3])
    29     u7 = Dropout(dropout)(u7)
    30     c7 = conv2d_block(u7, n_filters=n_filters*4, kernel_size=3, batchnorm=batchnorm)
    31
    32     u8 = Conv2DTranspose(n_filters*2, (3, 3), strides=(2, 2), padding='same') (c7)
```

Fig. 8 Rede de aprendizado profundo U-Net projetada.

A Figura 8 representa a configuração da rede U-Net, onde foi definido todos os parâmetros da rede.

Model: "model_2"

Layer (type)	Output Shape	Param #	Connected to
img (InputLayer)	(None, 128, 128, 1)	0	
conv2d_20 (Conv2D)	(None, 128, 128, 16)	160	img[0][0]
batch_normalization_19 (BatchNo	(None, 128, 128, 16)	64	conv2d_20[0][0]
activation_19 (Activation)	(None, 128, 128, 16)	0	batch_normalization_19[0][0]
conv2d_21 (Conv2D)	(None, 128, 128, 16)	2320	activation_19[0][0]
batch_normalization_20 (BatchNo	(None, 128, 128, 16)	64	conv2d_21[0][0]
activation_20 (Activation)	(None, 128, 128, 16)	0	batch_normalization_20[0][0]
max_pooling2d_5 (MaxPooling2D)	(None, 64, 64, 16)	0	activation_20[0][0]
dropout_9 (Dropout)	(None, 64, 64, 16)	0	max_pooling2d_5[0][0]
conv2d_22 (Conv2D)	(None, 64, 64, 32)	4640	dropout_9[0][0]
batch_normalization_21 (BatchNo	(None, 64, 64, 32)	128	conv2d_22[0][0]
activation_21 (Activation)	(None, 64, 64, 32)	0	batch_normalization_21[0][0]
conv2d_23 (Conv2D)	(None, 64, 64, 32)	9248	activation_21[0][0]
batch_normalization_22 (BatchNo	(None, 64, 64, 32)	128	conv2d_23[0][0]
activation_22 (Activation)	(None, 64, 64, 32)	0	batch_normalization_22[0][0]
max_pooling2d_6 (MaxPooling2D)	(None, 32, 32, 32)	0	activation_22[0][0]

Fig. 9 Parâmetros da rede de aprendizado profundo U-Net projetada.

Foi realizado um treinamento da rede utilizando o data set criado, nessa etapa foi definido a quantidade de épocas, e parâmetros da rede com objetivo de se obter a maior quantidade de acerto possível, sempre observado a acuracidade do resultado dos testes. Após o treinamento o resultado da acurácia como meta será acima de 87% conforme Figura 10.

```
Epoch 00020: val_loss did not improve from 0.29790
Epoch 21/200
432/432 [==============================] - 2s 5ms/step - loss: 0.2883 - acc: 0.9537 - val_loss: 0.3294 - val_acc: 0.9495

Epoch 00021: val_loss did not improve from 0.29790
Epoch 22/200
432/432 [==============================] - 2s 5ms/step - loss: 0.2858 - acc: 0.9540 - val_loss: 0.3249 - val_acc: 0.9500

Epoch 00022: ReduceLROnPlateau reducing learning rate to 1e-05.

Epoch 00022: val_loss did not improve from 0.29790
Epoch 23/200
432/432 [==============================] - 2s 5ms/step - loss: 0.2876 - acc: 0.9537 - val_loss: 0.3210 - val_acc: 0.9504

Epoch 00023: val_loss did not improve from 0.29790
Epoch 24/200
432/432 [==============================] - 2s 4ms/step - loss: 0.2854 - acc: 0.9541 - val_loss: 0.3170 - val_acc: 0.9508

Epoch 00024: val_loss did not improve from 0.29790
Epoch 25/200
432/432 [==============================] - 2s 5ms/step - loss: 0.2871 - acc: 0.9539 - val_loss: 0.3137 - val_acc: 0.9512

Epoch 00025: val_loss did not improve from 0.29790
Epoch 26/200
432/432 [==============================] - 2s 5ms/step - loss: 0.2859 - acc: 0.9540 - val_loss: 0.3106 - val_acc: 0.9515

Epoch 00026: val_loss did not improve from 0.29790
Epoch 00026: early stopping
```

Fig. 10 Treinamento da Rede.

A Figura 11 representa a curva de aprendizado da rede após o treinamento da rede, para realizar o treinamento da rede foi

considerado o uso de algumas ferramentas de otimização do treinamento tais como *early stop* e a redução da taxa de aprendizado da rede. O método *early stop* é um método que retém o melhor resultado de treinamento dentro de uma sequência de resultados posteriores, ele é usado para evitar que haja *overfit* de treinamento, nesse caso foi utilizado um *early stop* de 12 épocas o que significa que depois do melhor resultado irá ocorrer 12 épocas de resultados piores e encerrará o treinamento. A redução da taxa de aprendizado da rede foi utilizada para que caso ocorra tendência de piora no treinamento a taxa de aprendizado seja ajustada para uma tentativa de convergência do treinamento, nesse caso foi utilizada um ajuste da taxa a cada duas épocas sequenciais de resultados piores que o melhor resultado de aprendizado. Ao iniciar o treinamento a perda (*loss*) e a perda da validação (*validation loss*) iniciaram alta como o esperado, a curva de *loss* foi se caracterizando como uma assintótica e se estabilizando, a *validation loss* teve um comportamento variável sendo que nas épocas 7, 19 e 22 a taxa foi ajustada em função da piora da resposta ao treinamento, o melhor resultado foi obtido na época 2000 não foi necessário atuar o *stop early*, conforme demonstrado na Figura 11.

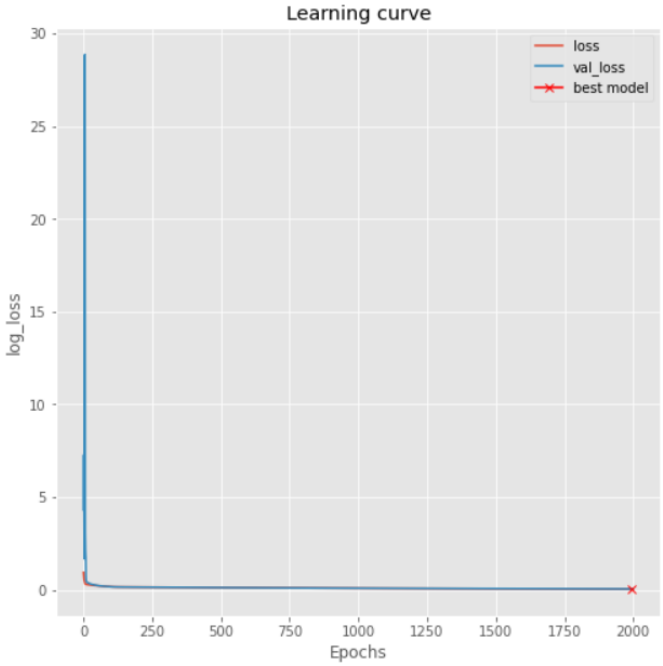

Fig. 11 Curva de aprendizado da rede.

Após treinamento da rede foi realizado a predição das imagens com o objetivo de verificar a qualidade do treinamento da mesma, a Figura 12 representa o resultado de duas predições de imagens de entrada na rede U-Net treinada. A primeira imagem é a imagem de entrada da rede, a segunda imagem é a imagem da máscara da imagem de entrada, ou seja, é a imagem com a qual iremos comparar com a imagem de saída da rede. A terceira imagem é a saída da predição da rede e a quarta imagem é a saída de predição com os valores de pixels binarizados (valores 0 ou 1).

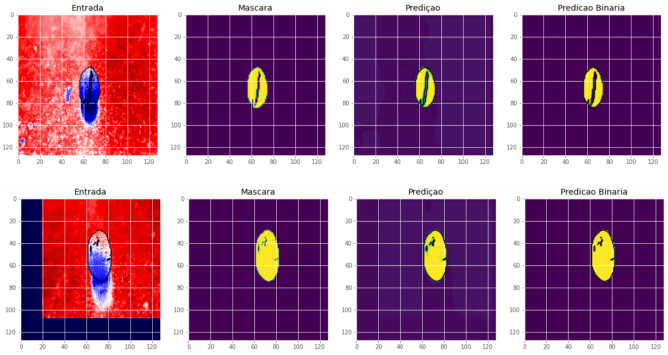

Fig. 12 Predição do teste.

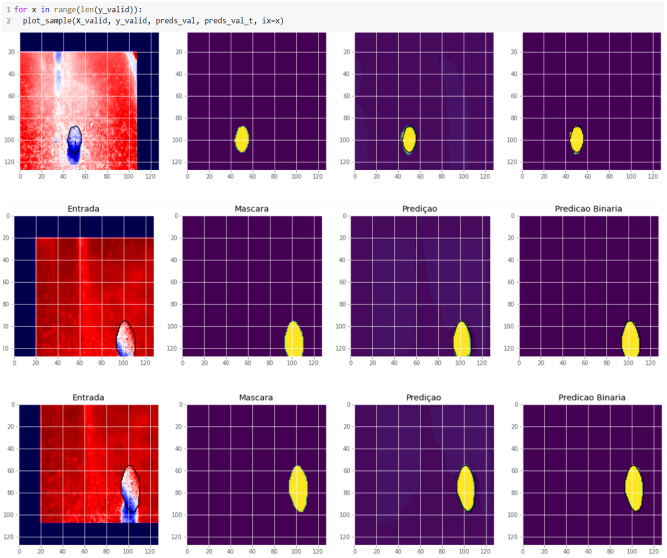

Fig. 13 Validação do teste.

A Figura 13 representa a validação do treinamento da rede, onde foram utilizadas 48 imagens que não fazem parte dos dados do treinamento da rede para validação da rede, e os resultados podem ser observados em três imagens conforme Figura 13, entrada, mascara, predição e predição binária.

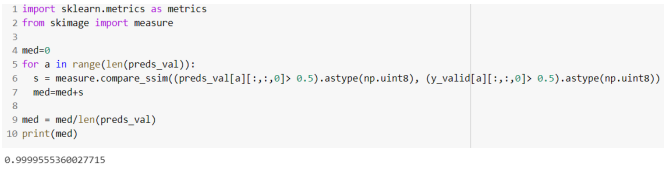

Fig. 14 Verificação do modelo.

Para confrontar as máscaras geradas pelas máscaras mapeadas foi usado o método de comparação por similaridade estrutural, o mesmo método foi aplicado e comparado a imagem oriunda da predição com a máscara criada, como resultado obteve-se o resultado de 99,9955%, isso significa que ambas imagens são aproximadamente 100% idênticas validando a qualidade do treinamento da rede.

5. CONCLUSÕES

A rede projetada atendeu a expectativa, utilizamos uma meta da acurácia acima de 87% e a rede treinada apresentou uma média de 97% de acurácia durante o treinamento e maior que 99% para a validação. Ao analisar o resultado para uma predição de uma imagem não utilizada no treinamento foi verificado que a qualidade da predição supera a expectativa e metas criadas no início do projeto.

Para trabalhos futuros será aplicada uma rede densa na saída da U-net para classificação de pelotas boas ou ruins onde serão coletadas imagens em tempo real que serão utilizadas como entrada na rede, para validação e aplicação no protótipo autônomo.

Essa rede será aplicada no protótipo autônomo para o teste de queda, uma inovação tecnológica que está sendo desenvolvido pelo Grupo de Pesquisa e Automação (GAIn), do Instituto Federal do Espírito Santo, situado no Município da Serra, para análise das fissuras das pelotas.

AGRADECIMENTOS

Agradecemos primeiramente a Deus, pois sem ele, nada seria possível, agradecemos também nossas famílias e amigos.

REFERÊNCIAS

BRAGA, A. P.; CARVALHO, A. C. P. L. F.; LUDEMIR, T. B. Redes Neurais Artificiais: teoria e aplicações. 1 ed. Rio de Janeiro: LTC, 2000.

Duan, J., Liu, X., Wu, X., & Mao, C. (2019). Detection and segmentation of iron ore green pellets in images using lightweight U-net deep learning network. *Neural Computing and Applications*, *8*. https://doi.org/10.1007/s00521-019-04045-8

Meyer, K., 1980, Pelletizing of Iron Ores, Springer-Verlag Berlin, Heidelberg, Germany.

Ponti MA, Costa GBP da. Como funciona o deep learning [Internet]. In: Tópicos em gerenciamento de dados e informações 2017. Uberlândia: SBC; 2017. Available from: http://sbbd.org.br/2017/wp-content/uploads/sites/3/2017/10/topicos-em-gerenciamento-de-dados-e-informacoes-2017.pdf

Raju, P., Rao, V. M., & Rao, B. P. (2018). Grey Wolf Optimization-Based Artificial Neural Network for Classification of Kidney Images. *Journal of Circuits, Systems and Computers*, *27*(14), 1–21. https://doi.org/10.1142/S0218126618502316

SUGOMORI, Y. et al. Deep Learning: Practical Neural Networks with Java. [S.l.]: Packt Publishing Ltd, 2017.

Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T., & Maida, A. (2019). Deep learning in spiking neural networks. *Neural Networks*, *111*, 47–63. https://doi.org/10.1016/j.neunet.2018.12.002

VALE. (2014). Entenda como funciona o processo de pelotização em nossas usinas. Retrieved from http://www.vale.com/brasil/PT/aboutvale/news/Paginas/entenda-funciona-processo-pelotizacao-usinas.aspx