Detecção e classificação de doenças de pele utilizando Redes Neurais Convolucionais

Lucas N Lanferdini
Universidade Federal de Ciências da Saúde de Porto Alegre
(UFCSPA)
Porto Alegre, Brasil
lucas.lanferdini@ufcspa.edu.br

I. Introdução

As redes neurais representam uma abordagem inovadora e poderosa na área de inteligência artificial, inspiradas pelo funcionamento do cérebro humano. Esses sistemas computacionais são compostos por unidades interconectadas, chamadas neurônios artificiais, que trabalham em conjunto para processar informações e realizar tarefas complexas de aprendizado e reconhecimento de padrões.

A estrutura de uma rede neural é organizada em camadas, incluindo uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída. Cada conexão entre os neurônios possui um peso, que é ajustado durante o treinamento da rede, permitindo a captura de relações complexas nos dados. Esse processo de ajuste de pesos é fundamental para o funcionamento eficaz da rede e para a capacidade de generalização em diferentes situações.

As aplicações de redes neurais são vastas e abrangem diversas áreas, como reconhecimento de padrões, processamento de linguagem natural, visão computacional, jogos e diagnóstico por imagem. Neste trabalho, ela será usada para fazer o diagnóstico das 10, seguintes, doenças de pele: Queratose Seborreica e Outros Tumores Benignos, Psoríase, Líquen Plano e Doenças Relacionadas, Dermatite Atópica, Fungo nas Unhas e Outras Doenças Ungueais, Queratose Actínica, Carcinoma Basocelular e Outras Lesões Malignas, Eczema, Herpes, HPV e Outras DSTs, Acne e Rosácea, Heras Venenosas e Outras Dermatites de Contato.

II. Materiais e Métodos

A. Dataset

O dataset disponível publicamente em repositório da plataforma Kaggle é dividido em dois diretórios principais: *Training* e *Test*, contendo os dados de treinamento e teste, respectivamente. Cada um destes diretórios possui 23 pastas, com os dados de cada possível classe ou categoria a ser determinada pelo modelo, para a realização do trabalho, foram selecionadas apenas as 10 doenças descritas anteriormente.

B. Ambiente de programação e Carregamento dos dados

O ambiente de programação escolhido foi a plataforma Google Colab Pro, utilizando a linguagem de programação Python e suas respectivas ferramentas e bibliotecas que serão utilizadas para a elaboração do modelo de aprendizado profundo. O Google Colab permite utilizar recursos computacionais da nuvem, além de permitir o desenvolvimento totalmente baseado na *web* a partir de um *browser*, facilitando o acesso, manutenção e compartilhamento do código-fonte. Para a elaboração do modelo, foi necessário o uso das bibliotecas de aprendizado de máquina TensorFlow e Keras.

Uma vez que o ambiente de programação é baseado na web, foi necessário carregar para este ambiente o dataset previamente obtido. Esse processo foi realizado utilizando o Google Drive.

C. Exploração e Visualização dos Dados

Inicialmente uma breve análise dos dados foi realizada. Esta etapa consistiu basicamente em (1) visualizar algumas imagens de exemplo para cada classe e (2) visualizar a distribuição dos dados. A análise exploratória é imprescindível para o conhecimento dos dados e do problema, melhorando o processo de decisões arquitetônicas posteriores.

D. Preparação dos dados

Antes de iniciar o treinamento da rede neural artificial, os dados passaram por algumas etapas de pré-processamento. O pré processamento dos dados visa garantir que os dados tenham qualidade e características adequadas para o modelo proposto. De tal forma, primeiramente as imagens que estavam na pasta de teste, foram adicionadas à pasta de treino, pois a divisão de treino e teste foi realizada no código, separando 20% para teste.

O segundo passo foi realizar a normalização das imagens, que, através de um conjunto de procedimentos que ajustam e padronizam as características visuais de uma imagem, torná-la mais adequada para processamento e análise. Por fim, uma vez que os algoritmos de aprendizado de máquina, em geral, não trabalham com dados categóricos, as classes (ou rótulos) foram transformados para representação binária, a partir de uma técnica chamada *One Hot Encoding*.

E. Arquitetura da rede e treinamento

A arquitetura da rede neural convolucional (CNN) foi definida utilizando a biblioteca Keras. A CNN possui três camadas convolucionais, cada uma seguida por uma camada de pooling para redução espacial, e uma camada densa com ativação ReLU, com 500 neurônios, e duas funções de dropout. Após as camadas convolucionais, a saída foi vetorizada usando uma camada de "Flatten" e uma camada de dropout foi incorporada para regularização.
A camada totalmente conectada consiste em duas camadas densas, a primeira com ativação ReLU e uma taxa de

dropout, e a segunda sendo a camada de saída com ativação softmax para classificação multiclasse. O modelo foi compilado utilizando o otimizador Adam, a função de perda categorical_crossentropy e métrica de acurácia.

Na etapa de treinamento propriamente dita, o tamanho de batch utilizado foi de 50 amostras e um total de 500 épocas.

Também foi usada a função ReduceLROnPlateau(), que é uma técnica usada em aprendizado de máquina para ajustar a taxa de aprendizado durante o treinamento de um modelo.

De forma comparativa, os resultados serão discutidos na seção IV.

III. Resultados

Os resultados obtidos perpassam desde a análise e preparação dos dados até o treinamento propriamente dito, seguido da fase de avaliação do modelo. Os resultados obtidos em cada etapa são discutidos nas sessões subsequentes.

A. Análise exploratória dos dados

A Figura 1 ilustra exemplos de imagens de treino. Já a Figura 2 apresenta as imagens de teste. As imagens foram levantadas dentro do próprio ambiente de programação.

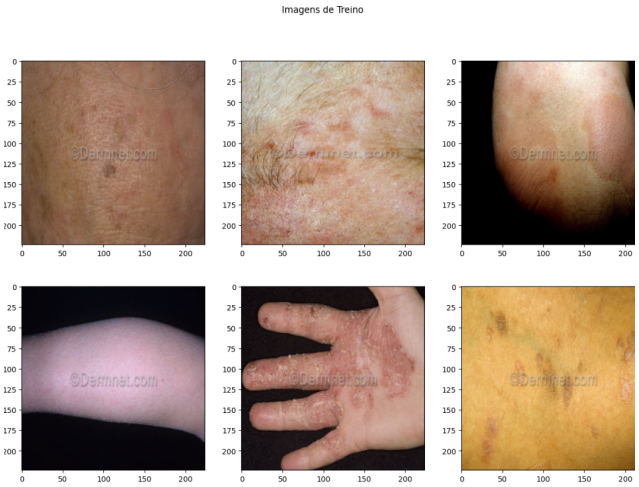

Figura 1. Exemplos de amostras de treino.

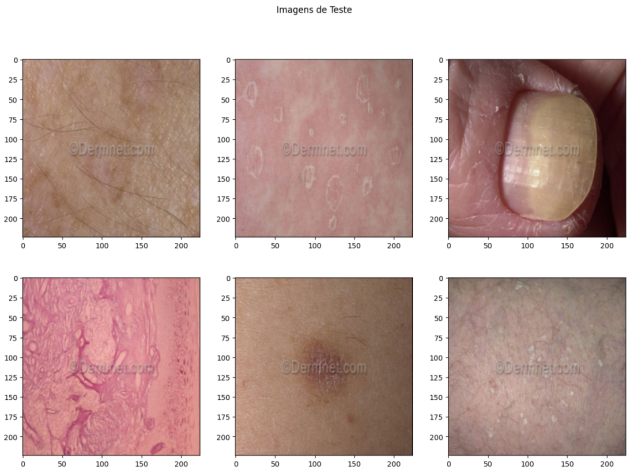

Figura 2. Exemplos de amostras de teste.

B. Avaliação dos modelos

Na Figuras 3 é possível observar a matriz de confusão do modelo. Já nas Figuras 4 é possível observar as curvas de aprendizado de cada modelo, tanto em termos de função de perda quanto acurácia.

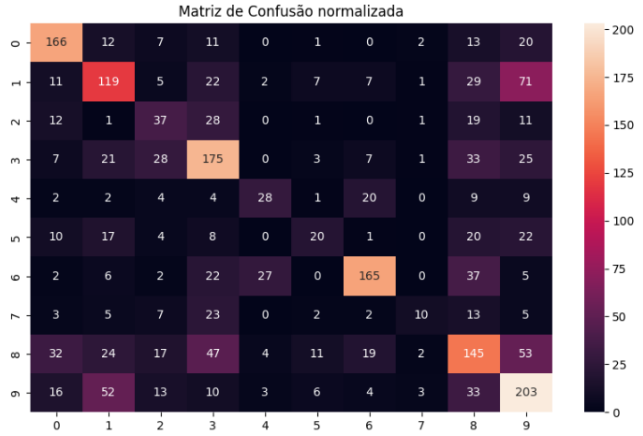

Figura 3. Matriz de confusão do modelo.

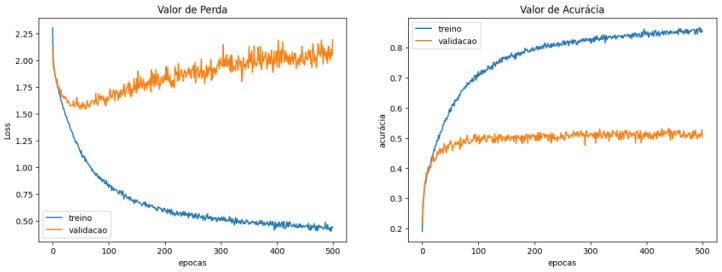

Figura 4. Avaliação do treinamento do modelo.

IV. Discussão

É possível perceber um resultado não muito positivo, apesar de algumas classes apresentarem bons resultados na matriz de confusão, o gráfico, que deve ser o mais analisado para saber se a rede está com um bom funcionamento, não tem um bom resultado. Ainda que a acurácia do treinamento esteja alta, com 0.8, é possível perceber um overfitting , quando o gráfico de validação é observado. Overfitting é

causado Overfitting é um comportamento indesejável de aprendizado de máquina que ocorre quando o modelo de aprendizado de máquina fornece previsões precisas para dados de treinamento, mas não para novos dados.
Este comportamento das curvas se deve a diversos fatores, especialmente ao tamanho do batch e a taxa de aprendizado. O problema foi parcialmente controlado devido ao uso da função ReduceLROnPlateau e com os dropout, que modifica a taxa de aprendizado durante o treinamento. Ainda assim, um aumento no tamanho do batch, junto com mais capacidade de processamento, ajudariam a obter melhores resultados.

Referências

"Deep Learning with Python": Chollet, F. (2021). Deep Learning with Python (2ª ed.). Simon and Schuster.

ALSHAZLY, Hammam et al. Explainable COVID-19 detection using chest CT scans and deep learning. Sensors, v. 21, n. 2, p. 455, 2021.

LECUN, Yann et al. LeNet-5, convolutional neural networks. URL: http://yann. lecun. com/exdb/lenet, v. 20, n. 5, p. 14, 2015.

Google Colaboratory. Google, 2023. Disponível em <https://colab.research.google.com/>

DeepwizAI. Why Random Shuffling improves Generalizability of Neural Nets. DeepwizAI, 29 de jan. de 2021. Disponível em: <https://www.deepwizai.com/simply-deep/why-random-shuffling-improves-generalizability-of-neural-nets>

C. Seger, "An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing", Dissertation, 2018.

Callbacks API. Keras API Reference, 2023. Disponível em: <https://keras.io/api/callbacks/>

Xue Ying. An Overview of Overfitting and its Solutions. *J. Phys.: Conf. Ser.* **1168** 022022, 2019.

Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. *J Big Data* **6**, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0