Análise de Redes Neurais Convolucionais para a classificação de doenças em folhas de soja

Analysis of Convolutional Neural Networks for diseases classification in soybean leaves

DOI:10.34117/bjdv8n6-339

Recebimento dos originais: 21/04/2022
Aceitação para publicação: 31/05/2022

Rafael Zeferino Rossi
Instituição: Instituto de Biotecnologia da Universidade Federal de Catalão (UFCat)
Endereço: Av. Dr. Lamartine Pinto de Avelar, 1120, Setor Universitário,
CEP: 75704-020, Goiás, Brasil
E-mail: rafaelzrossibr@gmail.com

Matheus Matos Machado
Instituição: Instituto de Biotecnologia da Universidade Federal de Catalão (UFCat)
Endereço: Av. Dr. Lamartine Pinto de Avelar, 1120, Setor Universitário,
CEP: 75704-020, Goiás, Brasil
E-mail: eu.lucasavila@gmail.com

Lucas Ávila Oliveira
Instituição: Instituto de Biotecnologia da Universidade Federal de Catalão (UFCat)
Endereço: Av. Dr. Lamartine Pinto de Avelar, 1120, Setor Universitário,
CEP: 75704-020, Goiás, Brasil
E-mail: m.matos1012.m@gmail.com

Gustavo Evangelista Araújo
Instituição: Instituto de Biotecnologia da Universidade Federal de Catalão (UFCat)
Endereço: Av. Dr. Lamartine Pinto de Avelar, 1120, Setor Universitário,
CEP: 75704-020, Goiás, Brasil
E-mail: gustavoevangelistaaraujo@gmail.com

Tércio Alberto dos Santos Filho
Instituição: Instituto de Biotecnologia da Universidade Federal de Catalão (UFCat)
Endereço: Av. Dr. Lamartine Pinto de Avelar, 1120, Setor Universitário,
CEP: 75704-020, Goiás, Brasil
E-mail: tercioas@ufcat.edu.br

Sérgio Francisco da Silva
Instituição: Instituto de Biotecnologia da Universidade Federal de Catalão (UFCat)
Endereço: Av. Dr. Lamartine Pinto de Avelar, 1120, Setor Universitário,
CEP: 75704-020, Goiás, Brasil
E-mail: sergio@ufcat.edu.br

RESUMO
A soja se tornou a principal cultura do agronegócio brasileiro, provocando impactos sociais e econômicos significativos em diversas regiões do país, apresentando uma evolução de 75,3 milhões de toneladas produzidas na safra 2010/2011 para 124,8 milhões de toneladas na safra de 2019/2020. Com base em estudos dos últimos anos, diversas tecnologias foram aplicadas na tentativa´ de realizar a detecção de doenças nessa cultura utilizando como base imagens˜ de suas folhas. Dentre as técnicas utilizadas pode-se citar o uso dos classificadores Support Vector Machines (SVMs) e K-Nearest Neighbours em conjunto com técnicas tradicionais de extração de características de imagens. Esse estudo aborda esse desafio utilizando Redes Neurais Convolucionais, uma técnica que tem como base o aprendizado profundo. Com este foco, foram experimentadas cinco arquiteturas de redes convolucionais (VGG16, ResNet50, DenseNet121, InceptionV3 e EfficientNetB7) e quatro diferentes otimizadores (SGD, Adam, RMSProp e Adadelta). Essas combinações foram treinadas usando transferência de aprendizado e ajuste fino de toda a arquitetura em questão. Ao final dos experimentos foi poss´ıvel analisar que arquitetura ResNet50 se destacou entre as demais, apresentando resultados significativos, além de apresentar a maior acurácia, dentre todos os modelos, atingindo o valor de 98.09%.

Palavras-chave: Redes Neurais Convolucionais, aprendizado profundo, doenças de soja.

ABSTRACT
Soybean became the main crop in the Brazilian agribusiness, causing significant social and economic impacts in different parts of the country. Its production has grown from 75.3 million tons in the 2010/2011 harvest to 124.8 million tons in 2019/2020. Based on recent researches, various technologies have been applied to detect and classify illnesses in this crop, using leaf images. Among the techniques used are Support Vector Machines (SVMs) classifiers, K-Nearest Neighbors algorithm, and traditional feature extraction image processing techniques. This paper proposes to apply Convolutional Neural Networks, a deep learning-based technique, to solve this task. Experiments, to evaluate its ability, were performed using five different architectures (VGG16, ResNet50, DenseNet121, InceptionV3 e EfficientNetB7) and four different optimizers (SGD, Adam, RMSProp e Adadelta). The architectures and optimizers were combined to train the models using Transfer Learning and Fine Tuning. Among the results obtained from the experiments, the architecture ResNet50 stood out, showing relevant results and an accuracy of 98.09%, the best result among all trained models.

Keywords: Convolutional Neural Networks, deep learning, soybean diseases.

1 INTRODUÇÃO

A soja se tornou a principal cultura do agronegócio brasileiro, provocando impactos sociais e econômicos significativos em diversas regiões do país. No período de 2011 à 2020 o cultivo de soja expandiu significativamente, passando de uma área de cultivo de 24,2 milhões de hectares com produção de 75,3 milhões de toneladas na safra

2010/2011 para 36,9 milhões de hectares com produção de 124,8 milhões de toneladas na safra de 2019/2020 [Hirakuri 2021].

Os prognósticos do Ministério da Agricultura, Pecuária e Abastecimento (MAPA) indicam que a produção de soja continuar a se expandir em todo o país, com possibilidade de alcançar uma área de produção de 46,5 milhões de hectares na safra 2029/2030 [Hirakuri 2021]. Devido a sua importância nacional, o cultivo de soja tem apoio de diversos programas de Pesquisa, Desenvolvimento e Inovação (PD&I) e transferência tecnológica, que permitem a geração e difusão de um conjunto de tecnologias e conhecimentos para aumentar o potencial produtivo das lavouras [Hirakuri 2021]. Nesse setor, uma das preocupações dos pesquisadores é aumentar a produção sem aumentar a área de cultivo, pois a expansão da área de cultivo provoca a destruição de ecossistemas naturais como o Cerrado, o Pantanal e a Floresta Amazônica. Um dos focos principais visando aumentar a produtividade sem aumentar a área de cultivo e o investimento em agricultura de precisão [Stafford 2000], onde um dos pilares e a identificação precoce de doenças e seu controle.

Em razão de variações nas condições edafoclimáticas (clima, solo, altitude, entre outras), as plantações de soja podem sofrer de dois tipos de doenças: as bióticas e as abióticas [Hirakuri 2021, de Melo Reis e Casa 2012]. As doenças bióticas, conhecidas também como infecciosas, são causadas por fatores bióticos ou microrganismos, como bactérias, fungos, nematoides e vírus. Já as doenças abióticas são causadas por fatores abióticos ou ambientais como carência ou excesso de nutrientes essenciais, e pelo uso indevido, ou inadequado, de produtos químicos como fungicidas, herbicidas e inseticidas [FERREIRA et al. 1979]. Ambos os casos de doenças interferem nas funções fisiológicas normais, causando alterações na aparência das plantas e/ou perda em produção, quando comparadas com cultivos sadios. Este trabalho foca em doenças bióticas.

O processo de identificação de doenças em plantações de soja e usualmente realizado por profissionais em campo, estando sujeito ao risco de uma interpretação subjetiva e distorções no julgamento, uma vez que esse processo é realizado visualmente. Em vários casos, os sintomas das doenças são pouco perceptíveis visualmente nas plantas, e assim podem não ser detectadas e tratadas, contudo elas provavam grandes perdas na produção, como é o caso da mancha parda e o crestamento bacteriano, que podem causar perdas de até 15% [FERREIRA et al. 1979]. Além do risco de falsos negativos (existe a doença, mas ela não é detectada), e de interpretações errôneas, pequenos e médios agricultores muitas vezes enfrentam dificuldades em

contratar os profissionais que realizam diagnósticos, o que faz com que a detecção ocorra tardiamente dificultando os procedimentos preventivos, acarretando em perdas na produção [Araujo e Peixoto 2019]. Nos últimos anos, motivados pelos avanços da visão computacional e do aprendizado de máquina, surgiram diversas pesquisas que visam automatizar o processo de identificação de doenças, para que o processo manual, que e caro e demorado, possa seja automatizado [Kaur et al. 2018]. Diversos pesquisadores tem proposto modelos de identificação de doenças na cultura de soja e apresentado resultados significativos.

Outro fator de extrema importância e o fato de a cultura de soja estar sujeita à ataques bióticos de patógenos e pragas agrícolas desde a sua semeadura até sua colheita, o que exige um acompanhamento contínuo. Além disso, a identificação de doenças no estágio inicial favorece o seu controle, diminui custos e minimiza as perdas de produtividade [Tetila 2019]. Estes aspectos motivam o desenvolvimento de métodos automatizados de detecção de doenças de soja através de técnicas de visão computacional, uma vez que o produtor não precisaria contratar ou esperar para uma visita de um profissional, bastando tirar algumas fotos das folhas a alimentá-las no sistema computacional de detecção de doenças.

Este trabalho efetua um estudo pratico de arquiteturas de redes neurais convolucionais e algoritmos de aprendizado, para a detecção de doenças em pedaços (recortes) de imagens de folhas de soja. Devido a pequena quantidade de imagens disponíveis para treinar as arquiteturas são experimentados múltiplos níveis de aumentos de dados, que são baseados em técnicas geométricas, de alteração de cores, entre outras, além de ser usado transferência de aprendizado a partir da base de dados ImageNet em todas as arquiteturas. No geral, os melhores resultados foram obtidos com o maior nível de aumento de dados, evidenciando as arquiteturas de CNNs experimentadas, precisam de um conjunto de treinamento extenso para obter resultados de alta acurácia.

As seções seguintes deste artigo são organizadas da seguinte forma: na Seção 2 são apresentados os conceitos de redes neurais convolucionais; na Seção 3 são sumarizados os principais trabalhos da literatura voltados para a detecção de doenças de soja a partir de imagens das folhas; na Seção 4 e descrito a base de imagens de doenças de folhas de soja original, os níveis (grupos) de aumento de dados, as arquiteturas de redes neurais e de otimizadores experimentados; na Seção 5 são apresentados e discutidos os resultados obtidos. Por fim, na Seção 6 são apresentadas as principais conclusões acerca do trabalho.

2 REDES NEURAIS CONVOLUCIONAIS

Redes neurais convolucionais (CNN) são um tipo de rede neural que foram desenvolvidas com o propósito de lidar com imagens. Esses modelos de redes neurais recebem esse nome devido a um tipo de camada presente em suas arquiteturas: as camadas convolucionais [Brownlee 2019]. Além das camadas convolucionais são normalmente utilizadas nesses modelos dois outros tipos de camadas: as camadas de *pooling* e as camadas completamente conectadas [Stanford 2021].

As camadas convolucionais são responsáveis por realizarem um tipo de operação chamada de convolação. Uma convolução consiste na multiplicação de uma porção de pesos em uma porção de dados, semelhante ao que ocorre nos outros modelos de redes neurais. Porém, devido aos dados de entrada serem bidimensionais, essas operações são realizadas utilizando matrizes bidimensionais de pesos, que recebem o nome de filtros ou *kernels*. Filtros, historicamente, eram construídos para tarefas específicas, porém a capacidade de aprendizado das redes neurais, permite que filtros inicializados com valores aleatórios evoluam durante o processo de treinamento, e aprendam sozinhos a extraírem características relevantes para a tarefa proposta [Brownlee 2019].

Os filtros utilizados sempre devem ser de dimensão menor do que os dados de entrada, sendo assim, a operação que o filtro realiza no dado de entrada e aplicada somente a uma parte local dos dados de entrada. Um filtro utilizado em imagem recebe o nome de Filtro Receptivo (*Receptive Field*). E importante destacar que a multiplicação aplicada entre o filtro e essa parcela local dos dados de entrada e realizada elemento a elemento, e no fim os valores resultantes são somados. Essa operação e chamada de Produto Escalar. Devido a diferença do tamanho do filtro e dos dados de entrada, é possível realizar a aplicação do filtro diversas vezes na imagem, em pontos diferentes. Essa possibilidade permite que um filtro especializado em detectar uma dada característica percorra toda a imagem e descubra os locais onde ela está presente. A aplicação do filtro ocorre no sentido esquerda para direita e de cima para baixo, e essa capacidade do filtro identificar uma dada característica, independentemente de sua posição, e chamada de Invariância a Translação [Brownlee 2019].

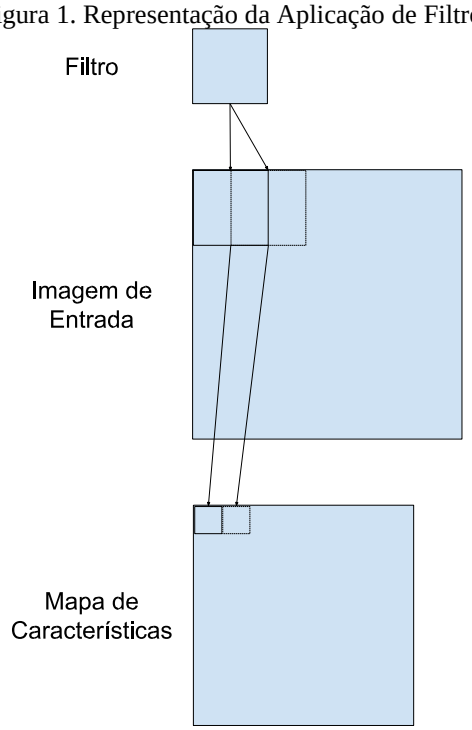
Ao aplicar um filtro em uma imagem, uma única vez, temos como saída um escalar. Porém ao aplica-lo na imagem toda, obtém-se como saída uma matriz bidimensional. Essa saída, que consiste na aplicação do filtro sobre todo o dado de entrada, e chamada de mapa de características (*feature map*). Por fim, após a criação do mapa de características, e possível aplicar uma função de ativação para cada valor ali

presente. É importante destacar que as CNNs aprendem em torno de 32 a 512 filtros diferentes para um dado conjunto de dados. Todos esses filtros durante o treinamento aprendem a ‘enxergar’ diferentes características das imagens apresentadas, e após essa fase e esperado que a rede consiga compreender também imagens similares as que foram apresentadas. Todo o processo de aplicação de um filtro sobre uma imagem de entrada e o processo de geração de um mapa de características e ilustrado na Figura 1 [Brownlee 2019].

É importante ressaltar que filtros não são aplicados somente aos dados de entrada. Eles também são aplicados nas saídas das camadas intermediarias. Essa capacidade de´ empilhar camadas convolucionais permite a extração de características complexas das imagens. Enquanto alguns filtros, de forma isolada, são capazes de extrair informações simples de imagens, como por exemplo linhas, filtros que operam na saída de outras camadas são capazes de utilizar as informações geradas por esses filtros simples para extrair informações complexas, como por exemplo um conjunto de linhas que determinam uma forma geométrica específica. Se esse processo de aplicar filtros nas saídas de outras camadas for perpetuado por mais camadas, e possível a criação de filtros que consigam extrair informações mais complexas, como faces, animais, casas, e assim por diante [Brownlee 2019].

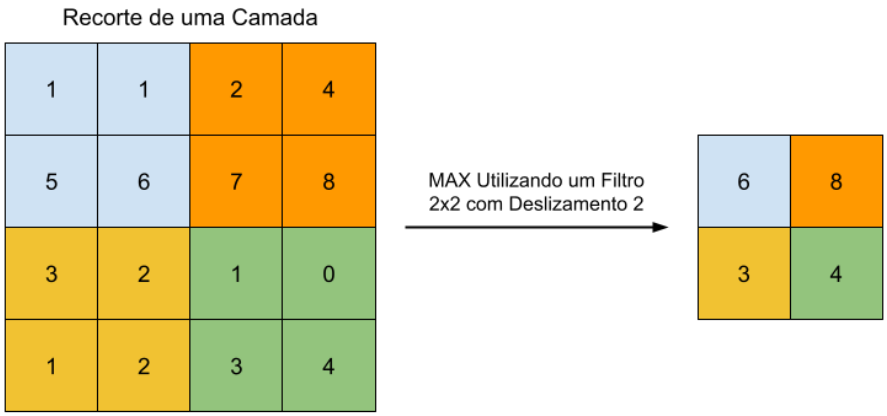
As camadas de *pooling* são inseridas entre as camadas convolucionais, e tem como objetivo reduzir o tamanho das matrizes que representam a informação que está sendo computada. Essa redução no tamanho da representação do dado tem como objetivo reduzir o número de parâmetros que serão computados na rede daquele ponto em diante, além de auxiliar a prevenir o *overfitting*. Para realizar essa redução nos tamanhos dos dados, a camada de *pooling* opera em cada camada do volume de saída da camada convolucional, realizando uma operação de sumarização, sendo a função máximo (Max) a mais tradicional. Essa operação e realizada com auxílio de um filtro, que assim como na camada convolucional irá percorrer todo o dado de entrada. A Figura 2 ilustra a aplicação de um filtro 2x2 com deslizamento 2 em uma camada de *pooling*. A aplicação desse filtro reduz a imagem em 75%, eliminando todos os pixels, menos o com o maior valor [Stanford 2021].

Figura 1. Representação da Aplicação de Filtros



Por fim, o último tipo de camada presente nas redes neurais: a camada completamente conectada. Uma camada completamente conectada opera da mesma forma que em outros modelos de redes neurais. Uma camada completamente conectada liga todas as saídas da camada anterior com todos os seus neurônios [Stanford 2021]. Nas redes neurais convolucionais pode se ter uma ou mais camadas completamente conectadas.

Figura 2. Representação da Aplicação de *Pooling*



3 TRABALHOS CORRELATOS

Nesta seção são descritas as principais pesquisas existentes na literatura que lidam com a classificação de doenças em soja através de análise de imagens das folhas. Essas pesquisas são sumarizadas pela Tabela 1, sendo que as doenças que cada trabalho considera, especificadas na coluna “ID das Doenças Analisadas” na Tabela 2.

Os trabalhos [Shrivastava e Hooda 2014], [Dandawate e Kokare 2015], [Pires et al. 2016], [Kaur et al. 2018], [Araujo e Peixoto 2019] e [Jadhav et al. 2020] usam métodos tradicionais de análise de imagens, baseados em métodos de extração de características pré-projetados e classificadores para a classificação de doenças de soja. Todos estes trabalhos extraem características a partir das imagens coloridas representadas pelo sistema de cores RGB, porem alguns trabalhos consideram a extração características representando a imagem em outros sistemas de cores, como os trabalhos [Dandawate e Kokare 2015], [Pires et al. 2016], [Kaur et al. 2018] que extraem também características considerando o sistema de cores *hue, saturation, value* (HSV). Para extração de características tem sido aplicado diversos métodos, contudo, há uma maior prevalência do uso extratores invariantes a aspectos de forma, como o *scale invariante feature transform* (SIFT), usado em [Dandawate e Kokare 2015] e [Pires et al. 2016], e de características de textura, principalmente as extraídas a partir de matrizes de coocorrencia, usadas em [Kaur et al. 2018] e [Jadhav et al. 2020]. Para a classificação o método mais empregado são as *support vector machines* (SVMs), seguidas pelo classificador *k-nearest Neighbours* (kNN). Os tamanhos dos *datasets*, o modelo de particionamento e treinamento, validação e teste, assim como o número de classes de doenças, variam de trabalho para trabalho, conforme mostrado na Tabela 1. Dentre os trabalhos correlatos que usam extratores de características pré-projetados, sumarizados na Tabela 1, o que produziu numericamente a maior acurácia, sendo esta de 99.83%, e [Pires et al. 2016], que explora informações das imagens representadas por múltiplos sistemas de cores, extraí características com múltiplos extratores, sendo dois deles derivados do método SIFT, e usa SVM como classificador.

[Sahu et al. 2021] e [Zhang et al. 2021] usam métodos que fazem o aprendizado de características para a classificação de doenças de soja, sendo estes baseados em redes neurais convolucionais. [Sahu et al. 2021] investiga o desempenho de múltiplas arquiteturas de CNNs (ResNet50, Alexnet, ResNet18, VGG16, VGG19, GoogLeNet), o uso de métodos de aumento de dados com *flipping*, modificação da escala da imagem, aplicação de rotações, translações e inserção de ruídos gaussianos, além do uso de

transferência de aprendizado. Os resultados do estudo mostram que o processo de aumento de dados permite que as redes neurais pré-treinadas consigam uma melhor generalização, obtendo maior acurácia. Já [Zhang et al. 2021] utiliza *regional CNNs*, que são redes neurais convolucionais que além de classificar as doenças, identificam as regiões na folha que as contêm. A principal contribuição do trabalho [Zhang et al. 2021] e o desenvolvimento de uma R-CNN baseada na Resnet50, como uma evolução da rede Faster R-CNN original que é baseada na CNN VGG-16. O modelo desenvolvido obteve uma acurácia superior a Faster R-CNN original em 18,17%.

[Tetila 2019] faz uma comparação entre as abordagens de métodos usados para a classificação de doenças em folhas de soja, ou seja, de métodos tradicionais que são baseados em extratores de características pré-projetados e classificadores clássicos, com métodos mais recentes baseados na aprendizagem de características através de CNNs. Dentre os classificadores clássicos experimentados (SVM, kNN , Naive Bayes, AdaBoost e Random Forest), o SVM foi a que se destacou atingindo uma acurácia de 98.34%. Já dentre as arquiteturas de redes convolucionais (Inception-V3, ResNet50, VGG16, VGG19, Xception, Inception-ResNet-v2 e DenseNet201), a que se destacou foi o modelo Inception-v3, que atingiu uma acurácia de 99.04%. Salienta-se que este resultado não poder ser comparado diretamente com o de [Pires et al. 2016], contudo ele e bastante expressivo pois em [Tetila 2019] foi utilizado um base de dados com seis classes, enquanto que em [Pires et al. 2016] havia somente três classes.

Tabela 1. Trabalhos Correlatos

Autor(es)	Técnica Base	Tipo de Imagem	ID das Doenças Analisadas (Vide Tabela 2)	Tamanho do Dataset	Distribuição do Dataset	Medida de Desempenho
[Shrivastava e Hooda 2014]	KNN	RGB	12, 13	100 / -	-	Acurácia - 75%
[Dandawate e Kokare 2015]	SVM	RGB & HSV	1, 2	120 / -	75 / - / 25	Acurácia - 98,42%
[Pires et al. 2016]	SVM	RGB & Grays-cale HSV	1, 6, 11	1200 / -		Correct Classification Rate (CCR) - 99.83%
[Kaur et al. 2018]	K-Means & SVM	RGB & HSV & L*a*b	1, 10, 13, 14	4775 / -	50 / - / 50 (Modelo 01) 60 / - / 40 (Modelo 02) 70 / - / 30 (Modelo 03)	Acurácia - 85.65% (Modelo 03)
[Araujo e Peixoto 2019]	Técnicas de Processamento de Imagens & SVM	L*a*b & HSL & RGB	5, 6, 7, 8, 9, 10, 11, 12	354 / 2832	70 / - / 30	Acurácia (73,1% - 77,5%)
[Tetila 2019]	SVM & KNN & Naive Bayes & J48 & AdaBoost & Random Forest & CNN	RGB	1, 4, 6, 9, 10, 11	3000 / -		Acurácia - 99,04%

[Jadhav et al. 2020]	SVM	RGB	1, 5, 12, 13	350 / -	77 / - / 23	Acurácia - 90,20%
[Sahu et al. 2021]	CNN	RGB	5, 6, 7, 8, 9, 10, 11, 12	321 / 2247	80/10/10 70/20/10 (Melhores Resultados) 60/20/20	Acurácia - 93% (ResNet50)
[Zhang et al. 2021]	MF^3 R-CNN	RGB	3, 5, 13	230 / 2230	70 / - / 30	Mean Average Precision - 83,64

Tabela 2. Trabalhos Correlatos - Doenças encontradas

Índice	Nome da doença
1	Saudável (Healthy)
2	Infectada
3	Viral Disease
4	Solo Exposto e Palha
4	Bacterial Blight
6	Ferrugem Asiática(Soybean Rust)
7	Copper Phytotoxicity
8	Soybean Mosaic
9	Mancha-Alvo (Target Spot)
10	Mıldio (Downy Mildew)
11	Oídio (Powdery Mildew)
12	Solo exposto e palha
13	Frog Eye
14	Septoria Leaf Blight

4 MATERIAL E MÉTODOS

Nessa seção será discutido a metodologia utilizada neste trabalho, desde a captação de imagens de folhas de soja doentes, os procedimentos de pré-processamento, as arquiteturas de CNNs utilizadas até os processos de treinamento.

4.1 BASE DE DADOS DE FOLHAS DE SOJA

Para a realização desse trabalho foram utilizadas imagens coloridas de folhas de soja com algum grau de uma das seguintes doenças: Crestamento Bacteriano, Míldio, Oídio e Murcha de Sclerotium. As imagens foram adquiridas da base da dados Digitaphos, disponibilizada publicamente pela Empresa Brasileira de Pesquisa Agropecuária (Embrapa), e tem como dimensão 4128x3096 pixels. Ao todo foram coletadas 220 imagens, distribuídas da seguinte forma: 51 imagens - Crestamento Bacteriano; 33 imagens - Míldio; 74 imagens - Oídio; 62 imagens - Murcha de Sclerotium.

4.2 AUMENTO DE DADOS

Devido a quantidade de dados não ser suficiente para o treinamento de redes neurais profundas, foi realizado um aumento na base de dados. Ao todo três grupos de aumento de dados foram gerados: Grupo 1, Grupo 2 e Grupo 3.

Para o Grupo 1, com auxílio de um *script* desenvolvido em Python, foram realizados recortes em diversas secções das imagens da base dados de forma manual, resultando em imagens com dimensão igual a 128x128 pixels. Ao final dessa fase foram obtidas 5947 imagens, distribuídas da seguinte forma: 1363 imagens - Crestamento Bacteriano; 1426 imagens - Míldio; 1600 imagens - Oídio; 1558 imagens - Murcha de Sclerotium. As imagens de cada classe foram divididas e agrupadas em três conjuntos: Teste (30%), Treinamento (50%) e Validação (20%). O processo de distribuição foi realizado de forma aleatória e a quantia de imagens, para cada classe, após a distribuição esta detalhada na Tabela 3.

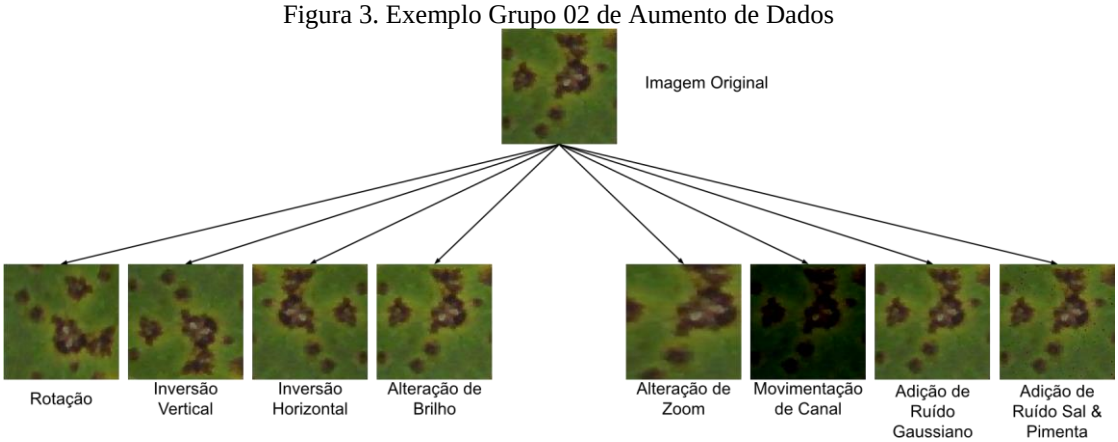
Tabela 3. Distribuição das Classes

	Crestamento Bacteriano	Mıldio	Oídio	Murcha de Sclerotium
Teste	409	428	480	467
Treinamento	682	714	800	780
Validação	272	284	320	311
Total	1363	1426	1600	1558

Para o Grupo 2, foi realizada uma cópia das imagens do Grupo 1, e o aumento de dados foi realizado somente para as imagens de treinamento. As operações realizadas nessa fase foram: Rotação – podendo rotacionar a imagens em qualquer angulo, de forma aleatória; Inversão Vertical – a imagem e invertida no sentido vertical; Inversão Horizontal – a imagem e invertida no sentido horizontal; Alteração no Brilho – o brilho da imagem e modificado, podendo resultado em qualquer valor no intervalo [0.2,0.9] Alteração no Zoom – a imagem e ampliada entre 50% e 90%; Movimentação de canal de cor – aleatoriamente um dos canais de cor da imagem e escolhido, e todos os seus pixels podem ter seu valor de intensidade modificado em até 75 unidades; Adição de Ruído Gaussiano e Adição de Ruído Sal Pimenta.

Todas essas modificações foram aplicadas para cada imagem original do Grupo 1 da partição de treinamento, ou seja, para cada imagem, 8 novas imagens eram criadas, assim aumentando o conjunto de treinamento em 8 vezes. A Figura 3 ilustra o resultado da aplicação desse aumento de dados em uma das imagens do conjunto de treinamento.

Figura 3. Exemplo Grupo 02 de Aumento de Dados



Por fim, O Grupo 3 de aumento de dados e gerado de maneira similar ao Grupo 2. O Grupo 1 e inicialmente copiado e o aumento também é realizado somente para as imagens de treinamento. No processo de aumento desse grupo são criados dois conjuntos de modificações: **A** = {Rotação, Inversão Vertical, Inversão Horizontal, Modificação no

Zoom} e **B** = {Alteração de Brilho, Movimentação de Canal, Ruído Gaussiano e Ruído Sal Pimenta}. Dado esses dois grupos, foi realizada a operação do produto cartesiano ($A \times B$), para combinar essas duas modificações, gerando ao todo 16 pares de modificações. Esses 16 pares foram aplicados para cada imagem, e no final todos os dados gerados do Grupo 2 foram adicionados nesse grupo também, gerando um aumento total de 24 vezes.

4.3 ARQUITETURAS DE REDES NEURAIS

Para avaliar o desempenho das CNNs para a tarefa de classificar as doenças de soja com base nas imagens de suas folhas foram escolhidas cinco arquiteturas diferentes: VGG16, ResNet50, DenseNet121, InceptionV3 e EfficientNetB7. Além das arquiteturas foram escolhidos também quatro otimizadores para serem avaliados durante os experimentos, sendo eles: SGD, Adam, RMSProp e Adadelta. Todos os otimizadores tiveram suas configurações *default*, conforme o pacote Keras do Tensorflow/Python, mantidas, menos a taxa de aprendizado que foi definida como 0.01 para todos os otimizadores. Para realizar o treinamento das redes foram feitas a transferência dos pesos da arquitetura pré-treinada na base dados ImageNet e em seguida rede toda a rede e retreinada na base de treinamento de folhas de soja. Dada essa combinação de Arquiteturas (5), Otimizadores (4), conjuntos de dados obtidos conforme o grupo de aumento de dados aplicado (3), e realizado um total de 60 experimentos.

Para a realização dos experimentos foi utilizado o ambiente do Google Colabs. De acordo com Google-Colab, o Google Colabs e um produto que permite a escrita e execução de código Python por meio do navegador, e é um ambiente adequado para aprendizado de máquina, análise de dados e educação. Devido a versão gratuita do ambiente ter algumas limitações como tempo máximo de uso de GPU e capacidade dos componentes disponibilizados, optou-se por utilizar uma das versões pagas do produto: o Google Colab Pro, que custa R$ 58,00 mensais, e disponibiliza CPUs e GPUs mais potentes, uma maior quantia de memória RAM, além de não limitar o uso diário de GPUs.

Devido a assinatura do produto, as máquinas utilizadas para o treinamento tinham a seguintes configurações: 25GB de memória RAM, aproximadamente 160GB de armazenamento e durante as sessões de treinamento as placas de vídeo utilizadas foram a Tesla P100 e Tesla T4. As configurações podem variar um pouco entre sessões, podendo haver até alteração nas placas de vídeo fornecidas, como ocorreu durante os treinamentos.

Durante o processo de treinamento e coleta dos resultados, foram utilizadas as seguintes bibliotecas no ambiente: Drive, os, Glob, keras, cv2, pickle, sklearn, csv,

pandas, io, matplotlib, numpy e tensorflow. As imagens utilizadas pelas redes neurais foram armazenadas no Google Drive, comprimidas em um arquivo .zip. Ao executar o código do ambiente, as imagens eram importadas para a máquina do Google Colabs e eram então descomprimidas.

Antes de iniciar o treinamento eram definidos os otimizadores com as configurações citadas anteriormente, e as arquiteturas das redes eram definidas, ajustando sua entrada para o mesmo formato das imagens utilizadas (128x128x3 – isto e, imagens coloridas com 128x128 pixels) e definindo os seus pesos iniciais como os pesos da rede em questão pré-treinada na base de dados Imagenet. Devido a modificação do formato dos dados de entrada, a parte final de todas as arquiteturas tiveram que ser refeitas e adequadas para o número total de possíveis resultados que seriam produzidos pela rede, ou seja, o número total de classes. Por fim, duas heurísticas foram escolhidas para serem utilizadas durante todos os experimentos: *Reduce Learning Rate* e *Early Stopping*, sendo elas aplicadas em conjunto para cada rede. A primeira heurística visa reduzir a taxa de aprendizado caso a rede não produza os resultados esperados dentro de um prazo mínimo. Na configuração escolhida, caso a rede não tenha uma redução nos resultados da sua função de perda de pelo menos 0.01% ao longo de 4 épocas, sua taxa de aprendizado e reduzida em 5 vezes. Já a segunda heurística, visa interromper o treinamento caso a rede não evolua conforme o esperado. Na configuração escolhida, caso a rede não tenha uma redução nos resultados da sua função de perda de pelo menos 0.05% ao longo de 12 épocas, o treinamento é encerrado.

O processo de treinamento de todos os experimentos foi configurado para durar no máximo 100 épocas. Devido ao grande volume de dados, as imagens foram carregadas em memória e utilizadas pela rede em lotes de 32 imagens. Durante esse processo de carga, as imagens são normalizadas no intervalo [0,1].

5 RESULTADOS

Nessa pesquisa foram realizados um total de 60 experimentos dados por treinar todas as arquiteturas propostas para serem avaliadas (VGG16, ResNet50, DenseNet121, InceptionV3 e EfficientNetB7) utilizando os diferentes otimizadores (SGD, Adam, RMSProp e Adadelta) em três conjuntos de dados, oriundos dos três grupos de aumento de dados. A forma treinamento se dá por transferir os pesos da arquitetura treinada na base de imagens ImageNet e retreinar a rede toda. Os resultados são avaliados através da medida de acurácia da classificação. Os resultados obtidos foram organizados em três

diferentes Tabelas 4, 5 e 6), onde cada tabela corresponde ao uso de um *dataset* oriundo de um grupo de aumento de dados. Todas as medidas de acurácia ilustradas nas tabelas supracitadas são calculadas sob o conjunto de teste, após o treinamento dos modelos

Tabela 4. Resultados (Acurácia) - Dataset 01

	VGG16	ResNet50	DenseNet121	InceptionV3	EfficientNetB7
RMSProp	0.2691	0.1973	0.9557	0.8733	0.9608
Adam	0.2691	0.4849	0.9529	0.9669	0.9664
SGD	0.2691	0.2522	0.8946	0.6491	0.5078
Adadelta	0.9596	0.2932	0.6934	0.5286	0.2225

Tabela 5. Resultados (Acurácia) - Dataset 02

	VGG16	ResNet50	DenseNet121	InceptionV3	EfficientNetB7
RMSProp	0.9675	0.2668	0.9439	0.8453	0.4787
Adam	0.2691	0.6396	0.9496	0.8627	0.3520
SGD	0.9630	0.2220	0.9535	0.8571	0.9714
Adadelta	0.9434	0.2836	0.9294	0.9260	0.9630

Tabela 6. Resultados (Acurácia) - Dataset 03

	VGG16	ResNet50	DenseNet121	InceptionV3	EfficientNetB7
RMSProp	0.9613	0.9333	0.9686	0.9400	0.9759
Adam	0.9703	0.9613	0.9686	0.9445	0.9776
SGD	0.6805	0.9809	0.9787	0.9725	0.9742
Adadelta	0.8683	0.9664	0.9552	0.8789	0.9709

Ao analisar os resultados de forma geral, e possível visualizar que a arquitetura VGG16, apresentou baixa acurácia para o primeiro dataset, com exceção do otimizador Adadelta onde atingiu uma acurácia de aproximadamente 96%. Já no segundo dataset, com exceção do otimizador Adam, a rede foi capaz de produzir resultados superiores à 94% de acurácia. Por fim, para o terceiro dataset, os otimizadores RMSProp e Adam foram os destaques, apresentando uma acurácia superior a 96% em seus resultados, enquanto o modelo com o otimizador Adadelta apresentou uma acurácia de 86% e o otimizador SGD fez com que o modelo atingisse uma acurácia inferior a 70%.

Já a arquitetura ResNet50, apresentou uma acurácia inferior a 70% para todos os modelos treinados utilizando os dois primeiros datasets. Já para o terceiro dataset, todos os resultados tiverem uma acurácia superior a 93%, e conseguiu gerar o modelo com

maior acurácia dentre todos os experimentos realizando, atingindo uma acurácia de aproximadamente 98.1% ao ser treinado utilizando o otimizador SGD.

A arquitetura DenseNet121, apresentou excelentes resultados, possibilitando que a maioria dos modelos treinados atingissem uma acurácia superior a 90%, com exceção de dois modelos treinados com o primeiro conjunto de dados que apresentaram acurácias iguais a 89% e 69% aproximadamente. O melhor resultado encontrado para essa arquitetura, foi o modelo treinado com utilizando o terceiro dataset e o otimizador SGD, que permitiram atingir 97.87% de acurácia.

A arquitetura InceptionV3, apresentou uma disparidade nos resultados para o primeiro dataset. Para os otimizadores RMSProp e Adam, os resultados foram positivos e apresentaram acurácia igual a 87.33% e 96.69%, respectivamente. Enquanto os otimizadores SGD e Adadelta apresentaram os valores 64.91% e 52.86%, respectivamente. Já para o segundo dataset, todos os modelos apresentaram acurácia superior a 84%, tendo como destaque o modelo treinado com o otimizador Adadelta, que apresenta acurácia igual à 92.6%. Por fim, para o terceiro dataset, resultados com acurácia superior a 94% prevaleceram, tendo somente o modelo treinado com o otimizador Adadelta como exceção, apresentando acurácia igual a 87.89%.

A arquitetura EfficientNetB7, apresentou, assim como a arquitetura InceptionV3, uma disparidade nos resultados para o primeiro dataset. Para os otimizadores RMSProp e Adam, os resultados foram positivos, também, e apresentaram acurácia igual a 96.08% e 96.64%, respectivamente. Enquanto os otimizadores SGD e Adadelta apresentaram os valores 50.78% e 22.25%, respectivamente. Já para o segundo dataset, essa disparidade foi invertida. Sendo assim, os otimizadores SGD e Adadelta se tornaram os destaques, apresentando acurácia igual a 97.14% e 96.3%, respectivamente, enquanto que os otimizadores RMSProp e Adam apresentaram acurácia igual a 47.87% e 35.20%, respectivamente. Por fim, o terceiro dataset apresentou resultados equilibrados, tendo acurácia próxima a 97% para todos os modelos.

6 CONCLUSÃO

Dados os resultados apresentados anteriormente, e possível observar que a arquitetura EfficientNetB7 apresentou resultados muito próximos durante o treinamento utilizando o terceiro dataset, e apesar dos resultados apresentarem resultados expressivos, nenhum deles foi o destaque dentre todos. O modelo que se destacou dentre todos foi o modelo originado pela arquitetura ResNet50, utilizando o terceiro dataset e o otimizador

SGD. A arquitetura DenseNet121, também apresentou resultados equilibrados, quando treinada utilizando o terceiro dataset. Já as demais arquiteturas, conseguiram apresentar bons resultados, porem a escolha do otimizador impactava muito na acurácia final, o que se torna visível dada as oscilações entre as acurácias dentro de um mesmo dataset.

Em estudos futuros pode-se utilizar outras formas de transferência de aprendizado, como por exemplo, retreinar somente as camadas completamente conectadas, e fazer o retreinamento em dois passos onde inicialmente retreina somente as camadas completamente conectadas e depois faz um ajuste fino em todos os pesos da rede. Além disso, pode-se analisar outras medias de desempenho como por exemplo *Precision*, *Recall*, *F1Score* e as matrizes de confusão.

REFERÊNCIAS

Araujo, J. M. M. e Peixoto, Z. M. A. (2019). A new proposal for automatic identification of multiple soybean diseases. Computers and Electronics in Agriculture, 167:105060.

Brownlee, J. (2019). Deep learning for computer vision: image classification, object detection, and face recognition in python. Machine Learning Mastery.

Dandawate, Y. and Kokare, R. (2015). An automated approach for classification of plant diseases towards development of futuristic decision support system in indian perspective. In 2015 International conference on advances in computing, communications and informatics (ICACCI), pages 794–799. IEEE. de Melo Reis, E. and Casa, R. T. (2012). Doenças da Soja. Berthier.

FERREIRA, L. P., LEHMAN, P. S., e ALMEIDA, A. M. R. (1979). Doenças da soja no brasil.

Hirakuri, M. H. (2021). Perdas econômicas geradas por estresses bióticos e abióticos na produção brasileira de soja no período 2016-2020.

Jadhav, S., Udupi, V., and Patil, S. (2020). Efficient framework for identification of soybean disease using machine learning algorithms. In International Conference on Image Processing and Capsule Networks, pages 718–729. Springer.

Kaur, S., Pandey, S., and Goel, S. (2018). A semi-automatic leaf disease detection and classification system for soybean culture. IET Image Processing, 12.

Pires, R. D. L., Goncalves, D. N., Orue, J. P. M., Kanashiro, W. E. S., Rodrigues Jr, J. F., Machado, B. B., and Gonçalves, W. N. (2016). Local descriptors for soybean disease recognition. Computers and Electronics in Agriculture, 125:48–55.

Sahu, S. K., Pandey, M., and Geete, K. (2021). Classification of soybean leaf disease from environment effect using fine tuning transfer learning. Annals of the Romanian Society for Cell Biology, pages 2188–2201.

Shrivastava, S. and Hooda, D. S. (2014). Automatic brown spot and frog eye detection from the image captured in the field. Am. J. Intell. Syst, 4(4):131–134.

Stafford, J. V. (2000). Implementing precision agriculture in the 21st century. Journal of Agricultural Engineering Research, 76(3):267–275.

Stanford (2021). Convolutional neural networks for visual recognition. https:// cs231n.github.io/convolutional-networks/#conv. Acesso em: 17 de Nov. de 2021.

Tetila, E. C. (2019). Detecção e classificação de doenças e pragas da soja usando imagens˜ de veículos aéreos não tripulados e técnicas de visão computacional.

Zhang, K., Wu, Q., and Chen, Y. (2021). Detecting soybean leaf disease from synthetic image using multi-feature fusion faster r-cnn. Computers and Electronics in Agriculture, 183:106064.