Identificação de Doenças na Folha da Soja a Partir da Análise Computacional de Bancos de Imagens de Folhas de Soja Variados

Fundamentos de Sistemas Inteligentes
Projeto 2 - Turma 01
Departamento de Ciência da Computação

Filipe Aziz Batista
UnB
16/0080118
160080118@aluno.unb.br

Gustavo Almeida Valentim
UnB
20/2014468
202014468@aluno.unb.br

Henrique Rodrigues Rocha
UnB
21/1036061
211036061@aluno.unb.br

Joao Vitor Dickmann
UnB
21/1042757
211042757@aluno.unb.br

Abstract

Com o crescimento das plantações de soja no Brasil, formas de cortar custos na produção vem recebendo grande atenção e investimentos. Como uma maneira de reduzir o uso de pesticidas, ampliamos uma rede neuronal convolucional já existente para detectar a passagem de lagartas em folhas de soja. Treinamos nosso modelo com mais de 9,500 imagens de plantas de soja e obtivemos resultados satisfatórios, porém ainda há espaços para melhora.

1. Introdução

O Brasil é o segundo maior produtor mundial de soja, atrás apenas dos EUA [4]. É comum, no entanto, que plantações sejam infectadas por uma série de doenças, o que pode afetar a economia brasileira. Por isso, técnicas de aprendizado de máquina, em especial as redes neuronais convolucionais (CNNs), podem ser de grande utilidade para detectar essas doenças em uma plantação, a fim de minimizar os estragos e reduzir o prejuízo causado pelas infecções.

Uma rede neuronal convolucional ou *convolutional neural network* é um tipo de rede neuronal artificial que utiliza uma sequência de camadas de convolução e compressão para extrair automaticamente recursos de dados brutos para então realizar uma classificação ou uma regressão. Por isso, as CNNs são frequentemente usadas para tarefas de visão por computador, pois com elas é possível realizar a classificação utilizando as próprias imagens como entradas.

A solução do artigo *Soybean disease identification using original field images and transfer learning with convolutional neural networks* [1] utiliza a CNN *DenseNet201* para classificar fotos de folhas de soja em plantações em 8 classes: saudável (*healthy*), crestamento bacteriano (*bacterial blight*), crestamento de cercospora (*cercospora leaf blight*), míldio (*downy mildew*), mancha-olho-de-rã (*frogeye*), deficiência de potássio (*potassium deficiency*), ferrugem asiática (*soybean rust*), e mancha-alvo (*target spot*).

Dessa forma, a partir dela, adicionaremos a passagem da lagarta (*caterpillar*), muito comum no Brasil e que ataca além da soja outras culturas como feijão, algodão, ervilha e amendoim [3], porém nesse estudo focaremos apenas na passagem dela na soja, com o objetivo de expandir o uso dessa tecnologia para uma maior variedade de aplicações.

Figura 1. Folhas de soja após passagem de lagarta

2. Metodologias e Materiais

2.1. Preparação e Obtenção dos Dados

O código foi feito na linguagem *Python* 3 por meio da plataforma *Google Colab* e da máquina disponibilizada

pela Universidade de Brasília. Usamos a biblioteca *Tensorflow* para compilar a CNN, além de outras APIs para a manipulação de dados e gráficos.

A parte de preparação das imagens segue o mesmo padrão do artigo do qual nosso estudo se inspira, dividindo as imagens desejadas do *dataset* em treinamento e teste, em uma proporção de 80% e 20%, respectivamente. Em seguida dividindo o set de teste em um set de validação e um set usado para medir a precisão do modelo.

Primeiro, estendemos o *dataset* do artigo base com as imagens do conjunto de dados *Images of Soybean Leaves* [2] de folhas afetadas pela lagarta. Especificamente, utilizamos as seguintes quantidades: 1632 *healthy*, 1202 *caterpillar*, 484 *bacterial blight*, 1727 *cercospora leaf blight*, 652 *downy mildew*, 1540 *frogeye*, 1034 *potassium deficiency*, 1627 *soybean rust*, e 1110 *target spot*. Então, como as redes neuronais convolucionais precisam que todos os dados tenham o mesmo tamanho, recortamos todas as imagens do *dataset* para que tenham o formato de 240×240 *pixels*.

Para aumentar o conjunto de dados e garantir que todas as classes tenham quantidades dados similares, aplicamos a técnica de *data augmentation*, que consiste em fazer pequenas alterações em uma porção do *dataset* original, como translações, rotações e alterações no brilho e no contraste.

Como o treinamento de CNNs costuma ser lento, podemos reaproveitar os pesos previamente obtidos em um *dataset* similar, modificando apenas as últimas camadas da rede neuronal para se adaptar ao nosso problema. Esse processo é chamado de transferência de aprendizado.

2.2. Especificação das Máquinas Usadas

O treinamento do modelo foi possível por meio do uso de máquinas disponibilizadas pela Universidade de Brasília (UnB), as especificações das máquinas estão disponíveis a seguir:

Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
CPU(s): 96
Model: 85
Model name: Intel(R) Xeon(R) Gold 5220R CPU @ 2.20GHz
CPU MHz: 1000.000
GPU1 product: GV100GL [Tesla V100S PCIe 32GB] Nvidia
GPU2 product: GV100GL [Tesla V100S PCIe 32GB] Nvidia

2.3. Parâmetros do Modelo

Os parâmetros do nosso modelo são diferentes dos usados no artigo [1], usando *hyperparameter tuning*, conseguimos reduzir o *overfitting* e melhorar a generalização para nosso experimento, encontramos hiper parâmetros que atuassem melhor em nossa versão *DenseNet201* e os pesos obtidos do banco de dados *ImageNet* via transferência de aprendizado, nos proporcionando 4 camadas:

1. uma camada *global average pooling*, para reduzir a quantidade de parâmetros necessários;
2. uma camada de 128 neurônios usando a função de ativação *ReLU*, com *dropout* de 0.2;
3. uma camada de 128 neurônios usando a função de ativação *ReLU*, com *dropout* de 0.4;
4. uma camada de 9 neurônios usando a função de ativação *softmax*.

Hyperparameter tuning também nos mostrou que a melhores taxas de *learning rate* e *epoch* são respectivamente:

- learning rate, com 0.001 ;
- epoch, com 10.

3. Resultados

Como é possível reparar, o projeto consta com uma base de imagens de grande escala, na faixa das 10500 imagens. Sendo assim, para que fosse possível progredir com o projeto, foi necessário iniciá-lo utilizando poucas imagens de amostra, visto que as máquinas pessoais dos autores não apresentam hardware para um processamento de alta qualidade. Dessa forma, iniciou-se os testes com 90 imagens e aos poucos a quantidade foi aumentada até que foi possível incluir todas as imagens da base.

O maior valor de precisão obtido nos testes foi de 97.4%, com o gráfico de perda disponível na Figura 2. Como podemos ver, apesar das oscilações nos períodos entre 20 e 40 e entre 80 e 100 , tanto o treinamento quanto a validação estão minimizando a perda, demonstrando a alta capacidade de predição do modelo.

A Figura 2 representa dados recolhidos durante o processo de testes e melhoria do projeto, tendo sido criada ao utilizar uma amostra de apenas 90 imagens. Devido ao longo tempo requisitado para gerar os dados necessários para criação dos gráficos, não houve tempo hábil para gerar o gráfico de perda atualizado com o uso total da amostra de imagens da base de dados, sendo assim, a figura citada apresenta resultados defasados quando se compara aos resultados finais.

A Figura 3 representa a matriz de confusão do nosso modelo, o conjunto de testes com a melhor precisão foi o conjunto das lagartas enquanto o pior foi o de míldio, com um percentual de 88.7% de precisão

Na Tabela 1 disponibilizamos a pontuação *precision*,*recall* e f1.

Figura 2. Gráfico de perda em 100 *Epochs*

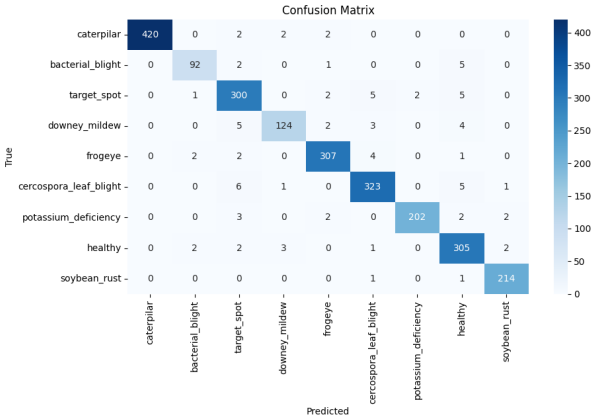

Figura 3. Matriz de confusão

	precision	recall	f1-score	support
Caterpillar	0.98	1.00	0.99	426
Bacterial Blight	0.96	0.94	0.95	97
Target Spot	0.96	0.95	0.96	320
Downey Mildew	0.99	0.97	0.98	128
Frogeye	0.93	0.98	0.96	314
Leaf Blight	0.97	0.96	0.96	337
Potassium Deficiency	0.98	1.00	0.99	202
Healthy	0.99	0.93	0.96	328
Soybean Rust	0.99	0.98	0.98	219
accuracy			0.97	2371
macro avg	0.97	0.97	0.97	2371
weighted avg	0.97	0.97	0.97	2371

Tabela 1. Métricas de desempenho dos testes

4. Discussões

Nosso objetivo com esse projeto foi trabalhar conceitos de aprendizado supervisionado dentro de um contexto prático, para alcançar nosso objetivo usamos um artigo existente para nos auxiliar. A partir desse artigo conseguimos analisar o uso de redes neuronais convolucionais e transferência de aprendizado e aplicar esses métodos ao nosso código.

No processo de utilizar esses conceitos, embora tenhamos sido capazes de apresentar bons resultados, ainda é possível melhorá-los manipulando os dados de treino de outra maneira, ou até mesmo usando outro *dataset* ou outra rede neuronal convolucional.

5. Agradecimentos

Este trabalho foi feito com o apoio da Universidade de Brasília e com o aluno Luís Fernando Ferreira Pereira Lopes do curso de Computação(Noturna).

Referências

[1] N. Bevers, E. J. Sikora, and N. B. Hardy. Soybean disease identification using original field images and transfer learning with convolutional neural networks. *Computers and Electronics in Agriculture*, 203(107449), 2022.

[2] M. E. Mignoni. Images of soybean leaves. `https://data.mendeley.com/datasets/bycbh73438/1`, 2021.

[3] plantix. Lagarta da soja. `https://plantix.net/pt/library/plant-diseases/600094/velvetbean-caterpillar`.

[4] E. Soja. Soja. `https://www.embrapa.br/soja/cultivos/soja1`.