Universidade Federal de Ouro Preto
Escola de Minas
CECAU - Colegiado do Curso de
Engenharia de Controle e Automação

Richard Stanley Mota de Oliveira

Sensoriamento Para a Agricultura de Precisão: Embarcados Para Aeromodelos

Monografia de Graduação

Ouro Preto, 2024

Richard Stanley Mota de Oliveira

Sensoriamento Para a Agricultura de Precisão: Embarcados Para Aeromodelos

Trabalho apresentado ao Colegiado do Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como parte dos requisitos para a obtenção do Grau de Engenheira(o) de Controle e Automação.

Universidade Federal de Ouro Preto

Orientador: Prof. Dr. Fernando Cortez Sica

Coorientador: Prof. Dr. Agnaldo José da Rocha Reis

Ouro Preto

2024

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

O48s Oliveira, Richard Stanley Mota de.
Sensoriamento para a agricultura de precisão [manuscrito]: embarcados para aeromodelos. / Richard Stanley Mota de Oliveira. - 2024.
44 f.

Orientador: Prof. Dr. Fernando Cortez Sica.
Coorientador: Prof. Dr. Agnaldo José da Rocha Reis.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Escola de Minas. Graduação em Engenharia de Controle e Automação .

1. Gerenciamento de recursos de informação - Agricultura de Precisão (AP). 2. Redes neurais (Computação) - Rede neural convolucional. 3. Microcontroladores. I. Sica, Fernando Cortez. II. Reis, Agnaldo José da Rocha. III. Universidade Federal de Ouro Preto. IV. Título.

CDU 681.5

Bibliotecário(a) Responsável: Maristela Sanches Lima Mesquita - CRB-1716

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
ESCOLA DE MINAS
DEPARTAMENTO DE ENGENHARIA CONTROLE E AUTOMACAO

FOLHA DE APROVAÇÃO

Richard Stanley Mota de Oliveira

Sensoriamento Para a Agricultura de Precisão: Embarcados Para Aeromodelos

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Engenheiro de Controle e Automação

Aprovada em 11 de outubro de 2024

Membros da banca

[Doutor] - Fernando Cortez Sica - Orientador (Universidade Federal de Ouro Preto)

[Doutor] - Agnaldo José da Rocha Reis - Coorientador (Universidade Federal de Ouro Preto)

[Mestra] - Regiane de Sousa e Silva Ramalho - Examinadora (Universidade Federal de Ouro Preto)

[Doutor] - Paulo Marcos de Barros Monteiro - Examinador - (Universidade Federal de Ouro Preto)

Fernando Cortez Sica, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 16/10/2024

Documento assinado eletronicamente por **Fernando Cortez Sica**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 16/10/2024, às 12:12, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0795528** e o código CRC **06C0A225**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591533 - www.ufop.br

Agradecimentos

Em primeiro lugar, agradeço aos meus pais, Elias e Aparecida, por serem meus pilares e exemplo de amor, força e dedicação constantes. Aos meus irmãos, Karitha e Elias, pelo carinho e pelo apoio incondicional. À minha família extensa, por toda a compreensão e suporte ao longo desta jornada.

Agradeço à comunidade da UFOP e à equipe docente pelos valiosos ensinamentos que recebi ao longo da minha formação. Em especial, ao meu orientador, professor Fernando Cortez Sica, pela orientação cuidadosa, paciência e suporte fundamental para a realização deste trabalho e também ao professor Agnaldo Reis que prontamente aceitou o convite para coorientação do projeto.

Àqueles que fizeram parte desta caminhada e tornaram o percurso mais leve e significativo, expresso minha gratidão: aos amigos de curso, João, Júlia, Luana, Pedro, Henrique, Thallys e Letícia, pelo companheirismo nas madrugadas de estudo e pelas lições aprendidas juntos; aos amigos de trabalho, Thomaz, Carlos, Matheus e Maria, pela parceria e suporte nos momentos de conciliar as demandas profissionais e acadêmicas; e aos amigos da República Badalação – Gabriel, Lucas M., Lucas A., Emanuel, Danilo, Vinícius, Luís, Guilherme, Henrique Q., Ângelo, Brenner e Fernando – pelos momentos de descontração e apoio que tornaram os dias mais alegres.

Aos amigos de Ouro Preto, Ana, Arthur H., Arthur S., Rodolfo., Júlio N., Mary, Gabriela, Vanessa, agradeço por serem parte da minha vivência nessa cidade tão especial. Aos amigos de Resplendor, Sayro, Jhonatas, Edson e Fernanda, agradeço pelo carinho e pela amizade de longa data, que sempre me acompanharam, independentemente da distância.

Por fim, agradeço a todos aqueles que, direta ou indiretamente, forneceram contribuições valiosas, seja por meio de insights, apoio moral, recursos ou suporte prático, para a realização desta pesquisa. A todos, o meu muito obrigado!

"Quando não tinha nada, eu quis." (Chico César)

Resumo

Este trabalho apresenta o desenvolvimento de um sistema para identificar doenças em folhas de soja, utilizando o microcontrolador ESP32-CAM para a captura de imagens em tempo real. As folhas são classificadas em três categorias: saudáveis, com Mancha Olho-de-Rã ou com Mosaico Amarelo. Para essa classificação, foi desenvolvido um modelo de rede neural convolucional capaz de reconhecer padrões nas imagens. O modelo foi treinado e validado, apresentando resultados satisfatórios e uma boa precisão na identificação das doenças. No entanto, a qualidade das imagens capturadas pela ESP32-CAM influenciou negativamente o desempenho. Dessa forma, o uso de uma base de dados gerada diretamente pela câmera pode aprimorar ainda mais os resultados futuros. O sistema foi integrado a uma interface gráfica em Python, que facilita a visualização dos resultados de maneira clara e intuitiva. Essa abordagem oferece uma solução eficiente para auxiliar os agricultores no monitoramento de doenças, contribuindo para um manejo agrícola mais preciso e eficaz.

Palavras-chaves: Agricultura de precisão. Rede neural convolucional. Microcontrolador.

Abstract

This work presents the development of a system for identifying diseases in soybean leaves, using the ESP32-CAM microcontroller for real-time image capture. The leaves are classified into three categories: healthy, affected by Frogeye Leaf Spot, or affected by Yellow Mosaic. A convolutional neural network model was developed to recognize patterns in these images. The model was trained and validated, yielding satisfactory results and good accuracy in disease identification. However, the quality of the images captured by the ESP32-CAM negatively impacted performance. Thus, the use of a dataset generated directly by the camera could further enhance future results. The system was integrated with a Python graphical interface, providing clear and intuitive visualization of the results. This approach offers an efficient solution to assist farmers in disease monitoring, contributing to more precise and effective agricultural management.

Key-words: Precision agriculture. Convolutional neural network. Microcontroller.

Lista de ilustrações

Figura 1 – Mapa mundial da produção de soja 15
Figura 2 – Representação da porcentagem de severidade da Doença do Mosaico Amarelo na folha de soja. 15
Figura 3 – Soja infectada pelo fungo Cercospora sojina 16
Figura 4 – A 24-bit RGB color cube 19
Figura 5 – Modelo de um neurônio de McCulloch e Pitts 20
Figura 6 – Funções de Ativação 20
Figura 7 – Representação de uma imagem 5x5 em escala de cinza 21
Figura 8 – Kernel Sobel Horizontal 3x3 21
Figura 9 – Aplicação do kernel na posição central 21
Figura 10 – Folha de soja com mancha olho-de-rã 22
Figura 11 – Kernel Top Sobel 22
Figura 12 – Folha após aplicação do Kernel Top Sobel 23
Figura 13 – Kernel Bottom Sobel 23
Figura 14 – Folha após aplicação do Kernel Bottom Sobel 23
Figura 15 – Kernel de contorno 24
Figura 16 – Folha após aplicação do Kernel de contorno 24
Figura 17 – Pinout ESP32-CAM 27
Figura 18 – ESP32-CAM e FTDI 29
Figura 19 – Amostra de folhas saudáveis 30
Figura 20 – Amostra de folhas com Mosaico Amarelo 30
Figura 21 – Amostra de folhas com Mancha Olho-de-Rã 31
Figura 22 – Interface 35
Figura 23 – Estrutura do projeto 36
Figura 24 – matriz de confusão 37
Figura 25 – Desempenho Do Modelo Por Número De Kernels 38

Lista de tabelas

Tabela 1 – Especificações do Dispositivo 28
Tabela 2 – Camadas da rede neural e seus parâmetros 34
Tabela 3 – Resultados da CNN por classe 37

Lista de abreviaturas e siglas

ANN	Rede Neural Artificial
CNN	Rede Neural Convolucional
VANT	Veículo Aéreo Não Tripulado

Sumário

1 INTRODUÇÃO 12
1.1 Objetivos 13
1.1.1 Objetivo Geral 13
1.1.2 Objetivos Específicos 13
1.2 Organização e estrutura 13

2 FUNDAMENTAÇÃO TEÓRICA 14
2.1 Origem da Soja e a Chegada no Brasil 14
2.2 Mosaico Amarelo 15
2.3 Mancha Olho-de-rã 16
2.4 Veículos Aéreos Não Tripulados 17
2.5 Microcontroladores 17
2.6 Imagem Digital 18
2.7 Redes Neurais Artificiais 19
2.8 Convolução 20
2.9 Redes Neurais Convolucionais 24

3 DESENVOLVIMENTO 27
3.1 ESP32-CAM 27
3.2 Base de Dados 29
3.3 Configuração do Ambiente 31
3.3.1 Python 31
3.3.2 Bibliotecas 31
3.3.3 IDE PyCharm 32
3.3.4 Anaconda 32
3.4 Desenvolvimento do Modelo 32
3.4.1 Pré-processamento de imagens 32
3.4.2 Construção e treinamento da CNN 33
3.5 Implementação do Sistema de Classificação em Tempo Real com Interface Gráfica 34
3.5.1 Arquitetura do sistema 34
3.5.2 Carregamento do modelo 34
3.5.3 Processamento da Imagem 34
3.5.4 Estrutura da Interface 35
3.6 Estrutura Final do Projeto 35

4 RESULTADOS 37

5 CONCLUSÃO 40

6 TRABALHOS FUTUROS 41
6.1 Desenvolvimento de uma Interface Web 41
6.2 Utilização de Autoencoders 41

Referências 42

1 Introdução

A chegada das inovações relacionadas à automação agrícola tem sido um diferencial significativo na rotina dos produtores, aumentando consideravelmente o rendimento operacional. Essas inovações otimizam o uso da mão de obra disponível e eliminam processos sujeitos a falhas decorrentes da fadiga física dos funcionários. Além disso, a automação permite uma maior eficiência, com o uso de tecnologias como *Big Data*, Internet das Coisas (IoT) e Inteligência Artificial, que integram diferentes dados do campo para melhorar a gestão agrícola (HEXAGON, 2021).

A agricultura de precisão oferece uma série de ferramentas para maximizar a produtividade, como eletrônica embarcada, geoestatística e sensores. Essas tecnologias permitem a coleta e análise de dados em tempo real, com a tecnologia da informação sendo usada para lidar com a variabilidade espacial e temporal, fatores críticos que podem influenciar a produtividade e a rentabilidade (SANTOS, 2024). Segundo a Associação Brasileira de Agricultura de Precisão e Digital (AsBraAP), muitos produtores já utilizam essas tecnologias para obter ganhos de produtividade, resultando em melhores resultados financeiros e redução de insumos (SANTOS, 2024).

A adoção dessas inovações auxilia o produtor não apenas na tomada de decisões, mas também na diminuição do impacto ambiental, uma vez que permitem o uso mais eficiente dos recursos disponíveis. Dessa forma, a agricultura de precisão torna-se essencial para um melhor gerenciamento da produção agrícola. Para Balastreire(2000), essa abordagem é, acima de tudo, um sistema de gestão agrícola completo, que integra diferentes aspectos da operação.

O uso de VANTs tem ganhado espaço na agricultura brasileira, especialmente no monitoramento de grandes áreas. De acordo com Viana(2018), o uso de VANTs é cada vez mais comum no Brasil para monitorar plantações, facilitando a identificação de pragas e doenças e possibilitando uma intervenção mais rápida e precisa. Além disso, os VANTs oferecem a vantagem de monitorar vastas áreas de forma ágil e sem dificuldades, coletando dados importantes sobre as culturas com o auxílio de sensores (FREEMAN; SILVA; PEREIRA, 2015).

Essas tecnologias, além de proporcionarem maior eficiência no uso da mão de obra, também oferecem benefícios ambientais, uma vez que promovem uma agricultura mais sustentável e de baixo impacto. No Brasil, empresas como a Fendt e a AGCO estão na vanguarda da inovação agrícola, utilizando tecnologia embarcada em máquinas autônomas e sistemas de monitoramento digital para otimizar as operações no campo (SANTOS, 2024).

1.1 Objetivos

1.1.1 Objetivo Geral

Desenvolver um sistema de classificação de doenças em folhas de soja utilizando imagens capturadas em tempo real por uma câmera ESP32-CAM, com a classificação feita por uma interface Python baseada em um modelo de rede neural convolucional, para auxiliar na identificação de folhas saudáveis, com *Frogeye Leaf Spot* (Mancha olho-de-rã) ou *Yellow Mosaic* (Mosaico Amarelo).

1.1.2 Objetivos Específicos

- Implementar a captura de imagens em tempo real utilizando a ESP32-CAM para transmitir as imagens.
- Desenvolver um modelo de rede neural convolucional para a classificação de imagens de folhas de soja em três categorias: saudáveis, *Frogeye Leaf Spot* e *Yellow Mosaic*.
- Integrar o modelo de classificação com uma interface gráfica em Python para realizar a classificação em tempo real e exibir os resultados de maneira amigável ao usuário.
- Treinar e validar o modelo de classificação utilizando um conjunto de dados de folhas de soja.

1.2 Organização e estrutura

Este trabalho está estruturado da seguinte maneira:

- No Capítulo 2, será apresentado a revisão bibliográfica a fim de uma melhor compreenção sobre a aquisição, envio e análise dos dados;
- O Capítulo 3, descreverá as etapas do desenvolvimento do projeto;
- No Capítulo 4 será apresentado os resultados de todas as técnicas empregadas neste trabalho, bem como algumas discussões relevantes acerca dos resultados obtidos;
- O Capítulo 5 trará uma conclusão sobre o trabalho;
- E por fim o Capítulo 6 apresentará sugestões para trabalhos futuros.

2 Fundamentação Teórica

2.1 Origem da Soja e a Chegada no Brasil

A soja cultivada tem sua origem no leste da Ásia, especialmente no nordeste da China, uma área também conhecida como Manchúria. Reconhecida como uma das culturas mais antigas, a soja foi introduzida no Ocidente entre o final do século XV e o início do século XVI. Mesmo após sua domesticação na China, a soja permaneceu confinada ao Oriente por cerca de dois milênios, em grande parte devido à ausência de intercâmbio agrícola significativo entre a China e outras regiões do mundo naquela época (Silva et al., 2022, apud Harlan, 1975).

De acordo com Silva (2022), a produção de grãos de soja em escala comercial começou no Rio Grande do Sul por volta de 1935. A Alemanha foi o primeiro país a importar soja brasileira em 1938. Em 1941, a soja aparece nas estatísticas do Rio Grande do Sul pela primeira vez, com uma área cultivada de 702 hectares. A partir da década de 1950, a soja expandiu-se para outras regiões do Brasil, como Sudeste, Norte e Nordeste. Inicialmente, no Sul do Brasil, a soja era produzida para alimentar suínos in natura, mas na década de 1950 foi instalada a primeira indústria de extração de óleo no país, aumentando o valor da soja na cadeia produtiva.

Apesar de a soja ter sua origem na China, atualmente o Brasil se consolidou como o maior produtor e exportador mundial desse grão, com destaque para os estados de Mato Grosso, Paraná, Rio Grande do Sul, Goiás e Mato Grosso do Sul. Esses estados contribuem significativamente para a posição de liderança do Brasil no mercado global de soja. No cenário internacional, os principais produtores que seguem o Brasil são os Estados Unidos, Argentina, China, Índia, Paraguai, Rússia, Canadá, Ucrânia e Bolívia (USDA, 2024). A figura 1 apresenta o mapa mundial da produção de soja e reforça esse quadro, mostrando uma concentração expressiva de produção nas regiões brasileiras, além de áreas relevantes nos Estados Unidos, especialmente no centro-norte, e na Argentina.

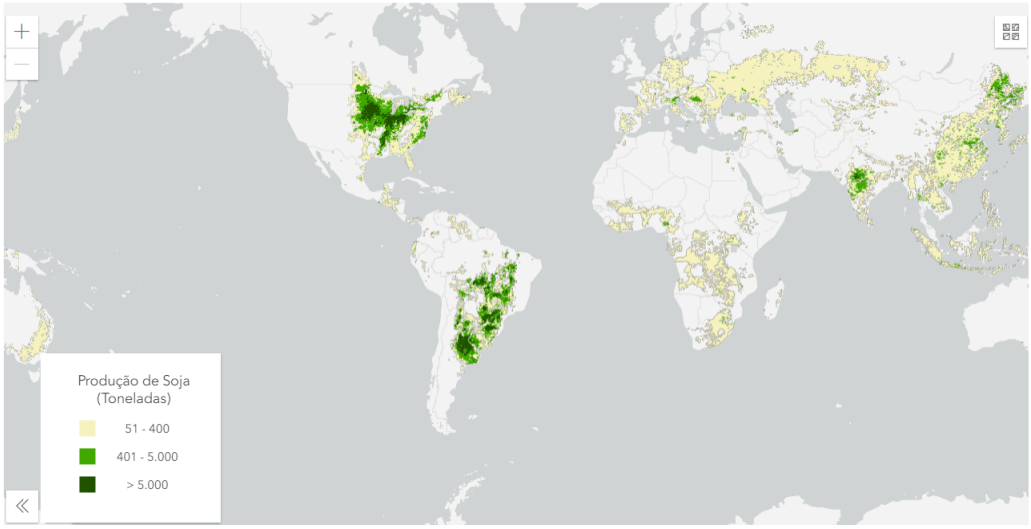

Figura 1 – Mapa mundial da produção de soja. Fonte: (USDA, 2024).

Ainda em relação à figura 1, nota-se que a produção asiática, liderada pela China, também se destaca, assim como as atividades menores em partes da Europa e da Índia. Esse panorama evidencia o papel crucial que o Brasil e outros países desempenham no abastecimento global de soja.

2.2 Mosaico Amarelo

O Mosaico Amarelo é uma das doenças virais mais prejudiciais para a cultura da soja, causando sérios impactos na produtividade. A doença é causada principalmente por dois vírus, o *Mungbean Yellow Mosaic Índia Virus* e o *Mungbean Yellow Mosaic Virus*, ambos pertencentes ao grupo dos *Legume Yellow Mosaic Viruses*. Esses vírus causam padrões de mosaico amarelo nas folhas da planta, que se manifestam como manchas irregulares amarelas intercaladas com áreas verdes. Em infecções severas, essas manchas podem evoluir para necrose e queda das folhas (RAHMAN et al., 2023). A figura 1 exibe os diferentes níveis da infecção na folha.

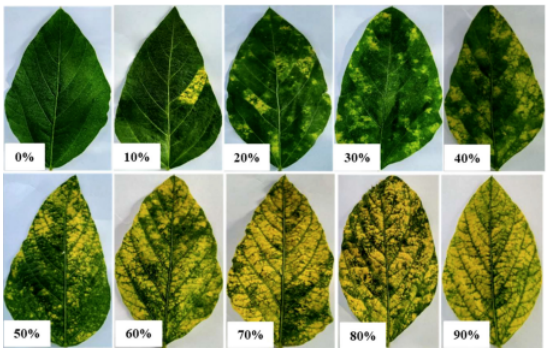

Figura 2 – Representação da porcentagem de severidade da Doença do Mosaico Amarelo na folha de soja.. Fonte: (RAHMAN et al., 2023).

Estudos realizados em regiões como Índia e Paquistão mostram que, quando a infecção ocorre em estágios iniciais do desenvolvimento da planta, as perdas de produtividade podem ser totais, especialmente em condições favoráveis à disseminação do vetor. Em 1996, a doença causou perdas de cerca de 105.000 toneladas métricas de soja (RAHMAN et al., 2023).

2.3 Mancha Olho-de-rã

A mancha olho-de-rã na soja, causada pelo fungo *Cercospora sojina*, é uma das principais doenças fúngicas que afetam a cultura da soja, provocando danos consideráveis na qualidade e no rendimento dos grãos (REIS, 1974). Os sintomas iniciais aparecem cerca de duas semanas após a emergência da planta, manifestando-se como pequenas manchas angulares e castanho-avermelhadas nas folhas unifolioladas. Em condições favoráveis, a doença pode se espalhar para os primeiros trifólios, resultando em desfolha severa (HENNING et al., 2014). À medida que a infecção avança, surgem pontuações pardas, com menos de 1 mm de diâmetro, que evoluem para manchas maiores com halos amarelados e centro castanho, medindo até 4 mm de diâmetro. Em casos mais severos, a mancha olho-de-rã pode causar maturação precoce, afetando diretamente a produtividade da lavoura.

O fungo *Cercospora sojina* sobrevive em restos culturais, o que favorece sua reemergência em ciclos seguintes, especialmente em sistemas de cultivo contínuo (HENNING et al., 2014). A infecção é favorecida por condições quentes e úmidas, com a dispersão dos esporos ocorrendo pela ação da água e do vento. O desenvolvimento da doença é mais intenso quando há períodos prolongados de molhamento foliar, com um mínimo de 6 horas, e temperaturas entre 15 °C e 30 °C, sendo a temperatura ideal para a infecção por volta de 25 °C (HENNING et al., 2014).

Além disso, a doença pode afetar as sementes, gerando rachaduras e manchas que comprometem tanto a germinação quanto o vigor das plântulas. Isso pode resultar em uma redução na qualidade da próxima safra, destacando a importância de monitorar a presença da doença desde os estágios iniciais do ciclo de crescimento (REIS, 1974). A figura 3 apresenta uma folha com a infecção.

Figura 3 – Soja infectada pelo fungo Cercospora sojina. Fonte: (HENNING et al., 2014).

O manejo eficaz da mancha olho-de-rã envolve a adoção de práticas de manejo integrado, como a rotação de culturas, que é uma estratégia eficaz para reduzir a sobrevivência do fungo nos restos de cultura. Melhorias nas condições físico-químicas do solo, como a adubação potássica, também contribuem para o controle da doença. Em anos de alta pluviosidade, o controle químico com fungicidas pode ser necessário para prevenir maiores perdas (HENNING et al., 2014).

2.4 Veículos Aéreos Não Tripulados

De acordo com a Associação Brasileira de Aeromodelismo (ABA), um VANT é definido como ”um veículo capaz de voar na atmosfera, projetado ou modificado para operar sem a necessidade de um piloto humano, sendo controlado remotamente ou de maneira autônoma”(JORGE; INAMASU, 2014). O uso dessa tecnologia trouxe uma nova era para a agricultura de precisão, permitindo monitoramento aéreo detalhado de lavouras e otimização de processos produtivos. Além disso, a possibilidade de capturar imagens em alta resolução e em tempo real, somada ao uso de sensores avançados, permite análises mais precisas e tomadas de decisão fundamentadas. De acordo com Jorge(2014), esse avanço é resultado direto da miniaturização de componentes tecnológicos, tornando os VANTs mais acessíveis e eficientes.

2.5 Microcontroladores

Nos últimos anos, a agricultura de precisão tem se destacado como uma prática essencial para otimizar o uso de recursos, aumentar a produtividade e minimizar os impactos ambientais. Nesse cenário, os microcontroladores desempenham um papel crucial ao viabilizar sistemas autônomos de monitoramento e controle. De acordo com Borges(2006), em sistemas de automação, é quase sempre necessária uma unidade de processamento, a menos que a lógica envolvida seja muito simples. Microcontroladores, como unidades centrais de processamento (CPUs), são amplamente utilizados por integrarem memória, interfaces de entrada e saída de dados e periféricos como conversores analógico-digitais (ADC), temporizadores e interfaces de comunicação serial.

Dispositivos como ESP32 e Arduino são exemplos de soluções acessíveis para a agricultura, pois permitem a integração de sensores que monitoram umidade e temperatura, facilitando a automação de sistemas de irrigação. Conforme apontado por Shamshiri(2018), o avanço das redes de sensores sem fio e o uso de Internet das Coisas (IoT) tornaram possível o monitoramento em tempo real de vastas áreas agrícolas, promovendo uma gestão mais eficiente de água e insumos.

Adicionalmente, Gebbers(2010) destacam que a automação baseada em microcon-

troladores oferece dados detalhados sobre a variabilidade do solo e dos cultivos, o que é fundamental para práticas agrícolas mais sustentáveis e adaptadas às necessidades específicas de cada área, otimizando recursos e minimizando desperdícios.

2.6 Imagem Digital

Uma imagem pode ser definida como uma função bidimensional f(x,y), onde x e y representam coordenadas espaciais no plano, e a amplitude de f em cada par de coordenadas é chamada de intensidade ou nível de cinza da imagem naquele ponto. Quando essas coordenadas e valores de intensidade são discretos e finitos, chamamos a imagem de digital. O processamento de imagem digital refere-se ao uso de um computador digital para manipular essas imagens. Cada imagem digital é composta por elementos com localizações e valores específicos, denominados pixels, que são os componentes básicos de uma imagem digital (GONZALEZ; WOODS, 2017).

Para imagens cromáticas teremos três unidades básicas para sua descrição radiância (energia total que flui da fonte de luz, medida em watts por metro quadrado por esteradiano), fluxo luminoso (energia que um observador percebe da fonte, medida em lúmens) e brilho, que é uma descrição subjetiva da intensidade e é um dos fatores chave na percepção das cores.

No modelo RGB, as cores são representadas por seus componentes primários de vermelho, verde e azul, organizados em um sistema de coordenadas cartesianas. O subespaço de cor-luz é representado por um cubo, onde as cores primárias ocupam três vértices, e as secundárias (ciano, magenta e amarelo) ocupam outros três. O preto está na origem, e o branco no ponto mais distante da origem.

As imagens RGB são compostas por três planos de imagem (um para cada cor primária), e a profundidade de pixel de uma imagem RGB de 24 bits resulta em um total de 16.777.216 cores possíveis. Esse modelo é amplamente utilizado para a representação digital de cores (GONZALEZ; WOODS, 2017). A figura 4 mostra o cubo de cores RGB de 24 bits, onde os valores das cores são escalados para o intervalo representável pelos bits da imagem. Para imagens digitais de 8 bits, por exemplo, o intervalo de valores ao longo de cada eixo do cubo vai de 0 a 255. Nesse caso, o branco estaria representada pelo ponto [255, 255, 255] no cubo.

Figura 4 – A 24-bit RGB color cubea. Fonte: (GONZALEZ; WOODS, 2017).

2.7 Redes Neurais Artificiais

Conforme exposto por Haykin(2001), a habilidade do ser humano em executar tarefas complexas, especialmente a capacidade de aprender, é fruto do processamento paralelo e distribuído de redes de neurônios no cérebro. O córtex cerebral, que é a camada mais externa do cérebro, desempenha um papel fundamental no processamento cognitivo. Novas informações ou experiências podem levar a mudanças estruturais no cérebro, como o fortalecimento ou enfraquecimento das conexões neuronais. Essas alterações resultam do rearranjo das redes de neurônios.

Os neurônios artificiais foram originalmente propostos por Mcculloch(1943) como uma forma de abstrair matematicamente o funcionamento dos neurônios humanos. A partir dessa ideia, surgiram as ANN, que são modelos computacionais compostos por conjuntos de neurônios artificiais capazes de reconhecer padrões em dados de entrada por meio de um processo de treinamento. As ANN podem ser estruturadas de forma simples, com um único neurônio, ou de maneira mais complexa, utilizando múltiplas camadas conectadas entre si.

Na figura 5, Rauber(2005) apresenta o modelo de um neurônio artificial de McCulloch e Pitts, e descreve um comparativo com a realidade biológica que ocorre em uma célula do sistema nervoso. A informação fornecida por outros neurônios entra em D entradas xj (=sinapses) no neurônio processador. O processamento consiste de uma combinação linear das entradas. A cada entrada está associado um peso wj que reflete a importância da entrada xj. O resultado dessa combinação linear é o valor net. Se esse valor ultrapassar um limiar µ, o neurônio "dispara" o valor 1 na saída binária y, se não ultrapassar o limiar a saída fica passiva em y=0, a função de ativação por soma ponderada é dada por:

$$y = \Theta \sum_{j=1}^{D} w_j x_j - \mu \tag{2.1}$$

Segundo Rauber(2005) a função de ativação no modelo de McCulloch e Pitts não é a única forma de determinar a saída de um neurônio. E exibe na figura 6 diferentes

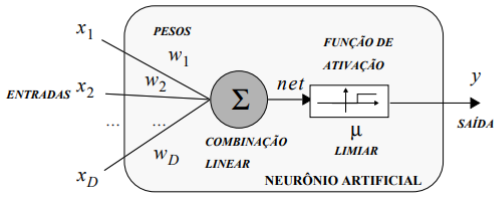

Figura 5 – Modelo de um neurônio de McCulloch e Pitts. Fonte: (RAUBER, 2005).

tipos de funções de ativação. A função linear gera uma saída contínua e proporcional à entrada, enquanto a função de escada produz uma saída binária (discreta e não linear). Já a função sigmoidal resulta em uma saída contínua e não linear, sendo amplamente utilizada em redes neurais para aproximar comportamentos mais complexos.

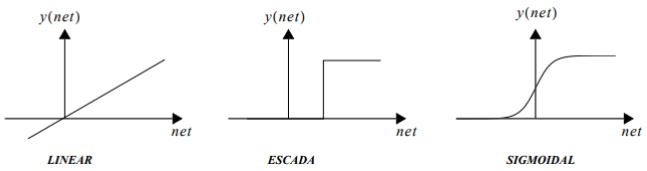

Figura 6 – Funções de Ativação. Fonte: (RAUBER, 2005).

Um dos principais exemplos de aplicação das redes neurais é a tarefa de classificação. Nesse processo, a rede calcula as probabilidades de um dado de entrada pertencer a uma determinada classe de saída. Esse tipo de aprendizado, conhecido como aprendizado supervisionado, envolve a coleta de características por meio de entradas fornecidas, onde os valores de saída já são conhecidos. A rede, então, busca aprender e representar as características comuns, ajustando seus parâmetros.

A classificação nas ANN envolve, além da camada de entrada e de saída, a chamada camada oculta, onde a rede processa e refina as características dos dados. As redes com múltiplas camadas conseguem representar relações mais complexas, o que as torna particularmente adequadas para a solução de problemas que exigem uma generalização eficiente e precisa dos dados apresentados durante o treinamento.

2.8 Convolução

A convolução, no contexto matemático e em processamento de imagens, é a operação entre duas funções, tipicamente uma função de entrada f(x,y) (a imagem) e uma função g(x,y) (o filtro ou *kernel*). A operação de convolução gera uma terceira função, que pode ser vista como uma versão modificada da função original f (PARKER, 2010). Em termos matemáticos, a convolução de duas funções f e g é definida como:

$$(f * g)(x, y) = \sum_{m} \sum_{n} f(m, n) \cdot g(x - m, y - n) \tag{2.2}$$

Sendo, f(m,n) o valor da função (imagem) em uma posição (m,n), e g(x - m, y - n) o valor do filtro em uma posição deslocada, o que permite a aplicação local do filtro sobre toda a imagem.

A figura 7 representa uma imagem 5x5 pixels em escala de cinza, onde os valores variam de 0 a 255, sendo 0 preto e 255 branco, e a figura 8 o Kernel Sobel de detecção de bordas horizontais.

$$\begin{bmatrix} 10 & 50 & 80 & 50 & 10 \\ 60 & 120 & 200 & 120 & 60 \\ 90 & 180 & 255 & 180 & 90 \\ 60 & 120 & 200 & 120 & 60 \\ 10 & 50 & 80 & 50 & 10 \end{bmatrix}$$

Figura 7 – Representação de uma imagem 5x5 em escala de cinza. Fonte: Elaboração própria.

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Figura 8 – Kernel Sobel Horizontal 3x3. Fonte: Elaboração própria.

Durante a convolução, o *kernel* percorre a matriz da imagem e para cada posição, o kernel faz a multiplicação dos valores correspondentes da imagem e do *kernel*, e os resultados são somados para formar um novo valor de pixel na imagem de saída. A figura 9 representa a aplicação do kernel na posição central da imagem.

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 120 & 200 & 120 \\ 180 & 255 & 180 \\ 120 & 200 & 120 \end{bmatrix} = \begin{bmatrix} -120 & 0 & 120 \\ -360 & 0 & 360 \\ -120 & 0 & 120 \end{bmatrix}$$

Figura 9 – Aplicação do kernel na posição central. Fonte: Elaboração própria.

A soma dos resultados é igual a zero indicando que é uma área de baixa transição.A figura 10 apresenta uma folha de soja com mancha olho-de-rã, as figuras 12, 14, 16 foram obtidas utilizando o *software* online Sentosa, e exibem o resultado da aplicação dos *kenels Top Sobel* figura 11, *Bottom Sobel* figura 13 e contorno figura 15 respectivamente.

Figura 10 – Folha de soja com mancha olho-de-rã. Fonte: (SETOSA, 2024).

1	2	1
0	0	0
-1	-2	-1

Figura 11 – Kernel Top Sobel . Fonte: (SETOSA, 2024).

Figura 12 – Folha após aplicação do Kernel Top Sobel. Fonte: (SETOSA, 2024).

-1	-2	-1
0	0	0
1	2	1

Figura 13 – Kernel Bottom Sobel. Fonte: (SETOSA, 2024).

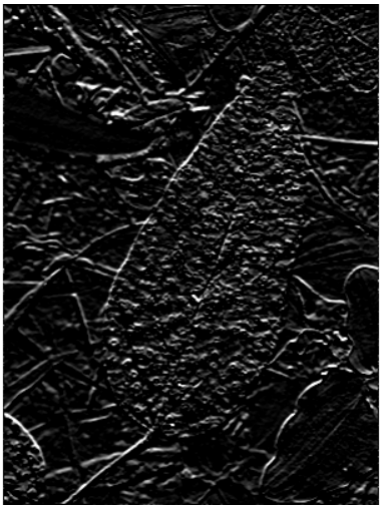

Figura 14 – Folha após aplicação do Kernel Bottom Sobel. Fonte: (SETOSA, 2024).

-1	-1	-1
-1	8	-1
-1	-1	-1

Figura 15 – Kernel de contorno. Fonte: (SETOSA, 2024).

Figura 16 – Folha após aplicação do Kernel de contorno. Fonte: (SETOSA, 2024).

2.9 Redes Neurais Convolucionais

Uma Rede Neural Convolucional é uma rede hierárquica de múltiplas camadas do tipo *feedforward*, na qual cada camada aplica múltiplas transformações utilizando um conjunto de *kernels* convolucionais (LECUN; BENGIO; HINTON, 2015). Dessa forma, a rede recebe imagens como entrada e, por meio de pesos e vieses, consegue identificar objetos ou diferenciá-los entre si.

A arquitetura das CNNs é inspirada no padrão de organização do córtex visual humano, onde os neurônios respondem a estímulos externos e transmitem a informação para os neurônios próximos. Esse método permite capturar relações espaciais e temporais, utilizando filtros que reduzem o número de parâmetros.

A CNN foi introduzida pela primeira vez com a arquitetura LeNet, desenvolvida por Lecun (1998), cujo foco principal era o reconhecimento de dígitos manuscritos no conjunto de dados MNIST. Embora essa arquitetura tenha estabelecido os fundamentos

das CNNs, foi apenas com a introdução da AlexNet que as CNNs se tornaram amplamente reconhecidas (DRAELOS, 2019). Além da camada de convolução citada anteriormente, outros componentes são essenciais nessas redes, como:

- **Padding**: O *padding* é a adição de valores extras ao redor da borda da imagem de entrada. O objetivo é preservar as dimensões da imagem após a aplicação dos filtros convolucionais. Sem *padding*, as dimensões da imagem diminuem a cada operação de convolução, o que pode resultar na perda de informações importantes (ZHANG et al., 2021).

- **Funções de ativação**: As funções de ativação introduzem não-linearidade no modelo, o que é essencial para permitir que a rede aprenda representações complexas dos dados. Sem elas, as camadas convolucionais funcionariam apenas como operações lineares, limitando o poder da rede.

- **Pooling**: O pooling é uma técnica usada para reduzir a dimensionalidade dos mapas de características, resumindo as informações locais em uma área menor. Isso ajuda a reduzir o número de parâmetros e a complexidade computacional da rede, além de fornecer invariância a pequenas variações e distorções nas imagens.

- **Dropout**: O *dropout* é uma técnica de regularização usada para evitar o *overfitting*, que ocorre quando o modelo aprende características muito específicas do conjunto de treinamento e tem desempenho ruim em novos dados. Durante o treinamento, o dropout desativa aleatoriamente uma fração dos neurônios de uma camada a cada iteração, forçando a rede a generalizar melhor. No final, todos os neurônios são usados durante a fase de inferência.

- **Flatten**: A camada *Flatten* é um componente simples, mas essencial em uma CNN. Seu principal objetivo é converter um tensor multidimensional em um vetor unidimensional. Isso é necessário porque, após as operações de convolução e pooling, os dados de entrada ainda estão organizados como mapas de características em 2D ou até mesmo em 3D. No entanto, para alimentar esses dados em uma camada densa, é preciso ”vetorizar”a estrutura.

- **Camada densa (Dense)**: Uma camada densa conecta todos os neurônios da camada anterior a cada neurônio da camada atual. Essa camada aparece normalmente nas partes finais de uma CNN, onde as características extraídas pelas camadas convolucionais são combinadas para fazer a classificação final. A camada densa recebe as saídas das camadas anteriores e gera uma predição, como a probabilidade de uma imagem pertencer a uma determinada classe.

Esses componentes formam a “espinha dorsal”das CNNs, permitindo que elas capturem informações hierárquicas de uma imagem, desde características simples até representações mais complexas que ajudam a realizar tarefas como reconhecimento de objetos, segmentação e classificação de imagens.

3 Desenvolvimento

Neste capítulo, são apresentados os materiais e passos necessários para o desenvolvimento do sistema proposto. Na seção 3.1, são fornecidas as informações sobre o microcontrolador ESP32 e as configurações desenvolvidas para captura e transmissão da imagem. A seção 3.2 apresenta a base de dados utilizada para o treinamento do modelo, seguida da seção 3.3, que descreve as bibliotecas e linguagens utilizadas. Já a seção 3.4 exibe o desenvolvimento do modelo e, por fim, na seção 3.5, é descrito o desenvolvimento das integrações e da interface.

3.1 ESP32-CAM

O ESP32-CAM é um módulo de hardware compacto que integra um microcontrolador ESP32 e uma câmera, proporcionando a captura de imagens e a sua transmissão via comunicação Wi-Fi. Suas características técnicas tornam este módulo uma solução adequada para projetos que requerem monitoramento em tempo real e processamento de imagens, especialmente em cenários que demandam baixo custo e alta eficiência, aliado a um design compacto e de baixo consumo energético. A figura 17 apresenta o *Pinout* do microcontrolador e a tabela 1 sua ficha técnica.

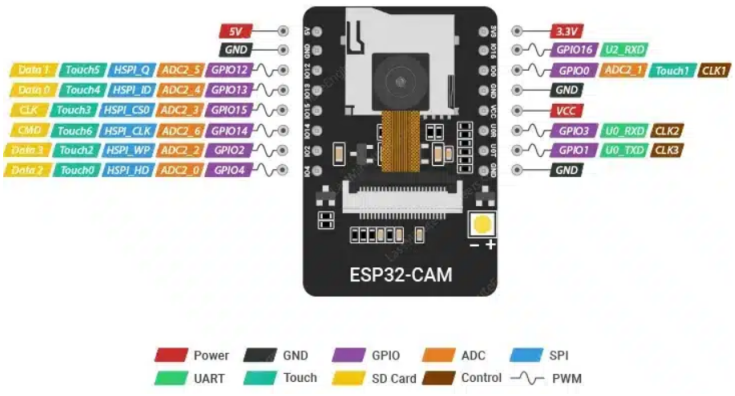

Figura 17 – Pinout ESP32-CAM. Fonte: (MAKERHERO, 2024)

.

Especificação	**Detalhes**
Processador:	Xtensa® Dual-Core 32-bit LX6
Memória Flash Programável:	4 MB (dos quais 0,9 MB são usados pelo bootloader)
Memória RAM:	520 KBytes
Memória ROM:	448 KBytes
Clock Máximo:	240 MHz
Pinos Digitais GPIO:	11 (todos com PWM)
Resolução do PWM:	até 16 bits (ajustável via código)
Wireless 802.11 b/g/n - 2.4Ghz (antena integrada):	ADC
Modos de Operação:	Access Point/Estação/Access Point + Estação
Bluetooth Low Energy:	padrão 4.2 integrado

Tabela 1 – Especificações do Dispositivo. Fonte (ESPRESSIF SYSTEMS, 2024)

O *firmware* para o ESP32-CAM foi desenvolvido utilizando a IDE do Arduino e tem como principal função configurar o módulo para capturar imagens e transmiti-las via rede Wi-Fi a um servidor local. Inicialmente, o *script* inicia um servidor HTTP enviando as imagens diretamente ao sistema de classificação.

O ESP32-CAM, por si só, não possui uma interface USB nativa para programação direta, sendo necessário o uso de um conversor serial TTL para USB, como o FTDI, que possibilita a transferência de dados entre o microcontrolador e o ambiente de desenvolvimento.

O FTDI é utilizado para conectar o ESP32-CAM ao computador via portas USB, permitindo a gravação do firmware no microcontrolador e a depuração durante o processo de desenvolvimento. Para realizar essa comunicação, é necessário conectar os pinos TX (Transmissão) e RX (Recepção) do FTDI aos pinos RX e TX do ESP32-CAM, respectivamente, e também garantir a alimentação do módulo, utilizando os pinos VCC e GND. Durante o processo de upload do *firmware*, o pino IO0 do ESP32-CAM deve ser conectado ao GND, colocando o dispositivo em modo de ”boot”, o que permite a gravação do código no microcontrolador. A Figura 18 apresenta a montagem do sistema.

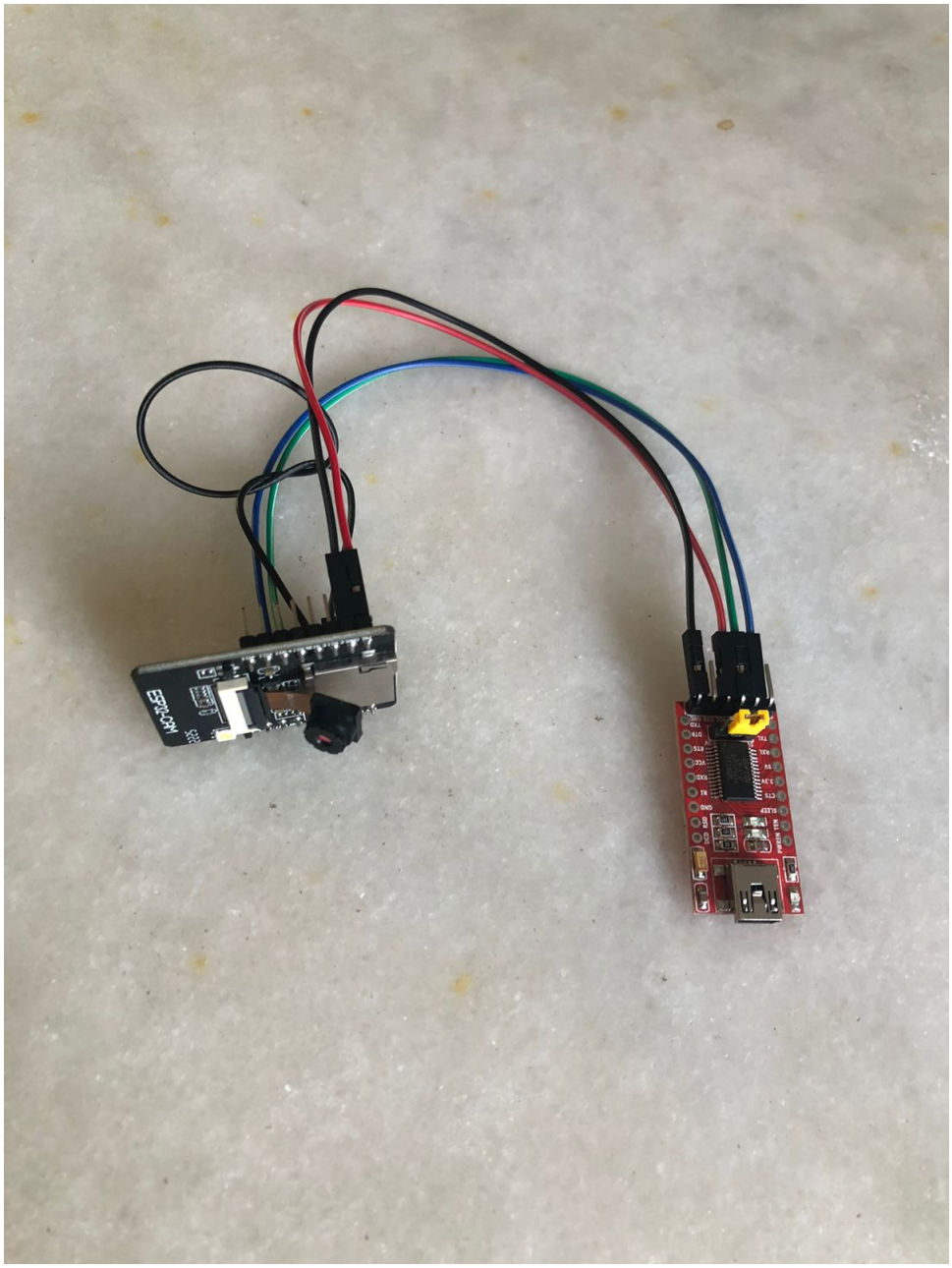

Figura 18 – ESP32-CAM e FTDI. Fonte: Elaboração própria.

3.2 Base de Dados

A base de dados utilizada no projeto foi obtida a partir do repositório "*Soybean Leaf Dataset for Disease Classification*", disponível no Kaggle (BHUJADE, 2022), e contém diversas categorias de classificação de doenças em folhas de soja. No entanto, para os propósitos deste trabalho, foram selecionadas apenas três categorias: folhas saudáveis, folhas afetadas pela doença Mancha Olho-de-Rã e folhas afetadas por Mosaico Amarelo. O conjunto de dados consiste em imagens de alta resolução, coletadas em diferentes condições, e cada imagem está devidamente rotulada, permitindo o treinamento e validação de modelos de

aprendizado profundo para a classificação das doenças. As figuras 19, 20 e 21 representam amostras das classes selecionadas.

Figura 19 – Amostra de folhas saudáveis. Fonte: (BHUJADE, 2022).

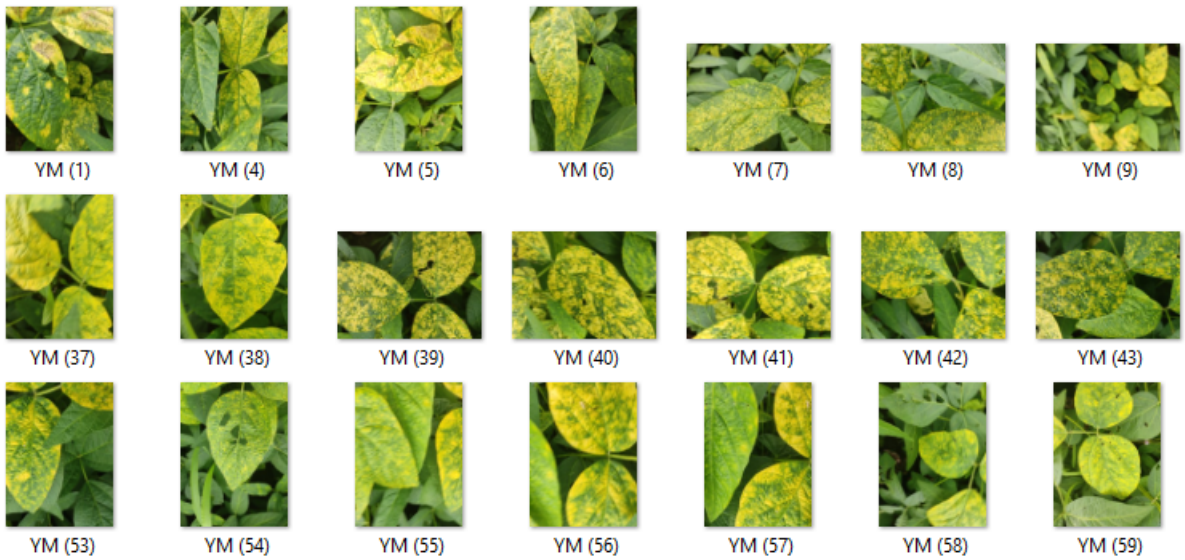

Figura 20 – Amostra de folhas com Mosaico Amarelo. Fonte: (BHUJADE, 2022).

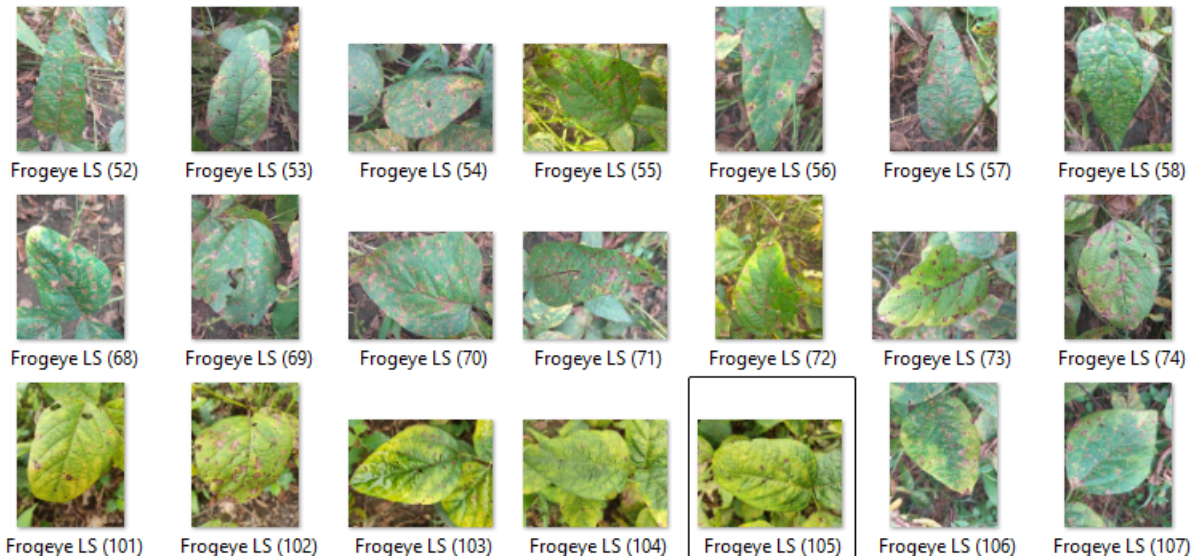

Figura 21 – Amostra de folhas com Mancha Olho-de-Rãs. Fonte: (BHUJADE, 2022).

3.3 Configuração do Ambiente

3.3.1 Python

Python foi a principal linguagem de programação utilizada no projeto, devido à sua versatilidade e ao amplo suporte para bibliotecas de aprendizado de máquina e processamento de imagens. A linguagem foi utilizada tanto para a criação da interface gráfica quanto para o processamento das imagens e a implementação do modelo de rede neural convolucional.

3.3.2 Bibliotecas

- **OpenCV**: Esta biblioteca foi essencial para o processamento e manipulação das imagens capturadas pelo ESP32-CAM. OpenCV permitiu o pré-processamento das imagens, incluindo redimensionamento, filtragem e detecção de objetos (folhas).
- **TensorFlow**: Utilizado para a construção e execução do modelo de rede neural convolucional. TensorFlow forneceu as ferramentas para carregar e utilizar o modelo treinado para classificar as folhas de soja nas categorias saudáveis, Mancha Olho-de-Rã e Mosaico Amarelo.
- **Keras**: A biblioteca Keras, integrada ao TensorFlow, foi utilizada para definir a arquitetura do modelo de rede neural. Sua simplicidade de uso facilitou a construção e treinamento da CNN.

- **NumPy**: Foi utilizada para a manipulação eficiente de arrays e matrizes, estruturas de dados fundamentais no processamento de imagens e no treinamento de redes neurais.
- **Matplotlib e Seaborn**: Essas bibliotecas foram empregadas para a visualização dos resultados, incluindo gráficos de desempenho do modelo, como a matriz de confusão e métricas de acurácia, permitindo a análise detalhada da performance do modelo.

3.3.3 IDE PyCharm

A IDE PyCharm foi escolhida como ambiente de desenvolvimento devido à sua robustez no gerenciamento de projetos em Python e ao suporte a pacotes e bibliotecas de aprendizado de máquina. A interface intuitiva da IDE facilitou o desenvolvimento e a depuração do código, além de integrar facilmente o ambiente com o Anaconda para gerenciamento de dependências.

3.3.4 Anaconda

Anaconda foi utilizada para gerenciar o ambiente de desenvolvimento Python, especialmente no que se refere à instalação e manutenção das bibliotecas necessárias, como TensorFlow, Keras, OpenCV e NumPy. Esse ambiente proporcionou maior controle sobre as dependências e as versões das bibliotecas, garantindo a compatibilidade com o código desenvolvido.

3.4 Desenvolvimento do Modelo

3.4.1 Pré-processamento de imagens

O pré-processamento das imagens é uma etapa crucial para a preparação do conjunto de dados. O script utiliza a classe `ImageDataGenerator` da biblioteca Keras para aplicar transformações às imagens, como normalização dos valores dos pixels (0-1), rotações aleatórias, espelhamento horizontal, transformações de cisalhamento e zoom aleatório, o que aumenta a robustez do modelo ao gerar novas variações das imagens. Duas instâncias desse gerador foram criadas: uma para o conjunto de treinamento, que aplica as transformações como mostra a figura Listagem 3.1, e outra para o conjunto de teste, que apenas normaliza as imagens.

```
1 train_data_generator = ImageDataGenerator(rescale=1./255,
2                                           rotation_range=7,
3                                           horizontal_flip=True,
4                                           shear_range=0.2,
```

```
5                                                   height_shift_range=0.07,
6                                                   zoom_range=0.2)
7 training_dataset = train_data_generator.flow_from_directory(
8     r'C:\Users\richa\PycharmProjects\TCC_LOCAL\Tcc_data\training_set'
          ,
9     target_size=(250, 250),
10    batch_size=8,
11    class_mode='categorical',
12    shuffle=True
13 )
```

Listing 3.1 – Training Dataset. Fonte: Elaboração própria.

O conjunto de dados foi organizado em pastas para facilitar o uso da função `flow_from_directory`, que carrega as imagens diretamente do diretório especificado.

3.4.2 Construção e treinamento da CNN

A CNN foi implementada utilizando a API `Sequential` do Keras, uma estrutura que permite empilhar camadas de forma linear. A arquitetura foi composta por múltiplas camadas de convolução seguidas por camadas de pooling. Essas camadas são responsáveis pela extração e redução das características das imagens, respectivamente. Após o processo convolucional, a estrutura é "achatada" para ser processada por camadas densas, que desempenham a classificação, para o treinamento foram utilizadas duzentos e setenta imagens da base criada por Bhujade (2022).

A arquitetura final possui três camadas convolucionais e três camadas densas, sendo a última uma camada de saída com três neurônios, correspondentes às três classes (saudável, Mancha Olho-de-Rã, Mosaico Amarelo), com a função de ativação `softmax` para prever probabilidades. Utilizando a função `summary` podemos ter uma visão geral da CNN como mostra a tabela 2.

O modelo foi compilado utilizando o otimizador `Adam` e a função de perda `categorical crossentropy`, apropriada para problemas de classificação com múltiplas classes. O treinamento foi realizado com 25 épocas, onde o modelo ajustou seus pesos com base nos dados de treinamento.

Além disso, o modelo treinado foi salvo para ser reutilizado posteriormente. O script exporta tanto a arquitetura da rede neural quanto os pesos em dois formatos diferentes: `JSON` para a estrutura e `HDF5` para os pesos, garantindo que o modelo possa ser recarregado e utilizado em novas inferências sem a necessidade de retreinamento.

Layer (type)	**Output Shape**	**Param #**
conv2d (Conv2D)	(None, 248, 248, 32)	896
max_pooling2d (MaxPooling2D)	(None, 124, 124, 32)	0
conv2d_1 (Conv2D)	(None, 122, 122, 32)	9,248
max_pooling2d_1 (MaxPooling2D)	(None, 61, 61, 32)	0
conv2d_2 (Conv2D)	(None, 59, 59, 32)	9,248
max_pooling2d_2 (MaxPooling2D)	(None, 29, 29, 32)	0
flatten (Flatten)	(None, 26912)	0
dense (Dense)	(None, 300)	8,073,900
dense_1 (Dense)	(None, 300)	90,300
dense_2 (Dense)	(None, 3)	903

Tabela 2 – Camadas da rede neural e seus parâmetros

3.5 Implementação do Sistema de Classificação em Tempo Real com Interface Gráfica

3.5.1 Arquitetura do sistema

O sistema desenvolvido possui dois componentes principais:

- **Interface gráfica**: permite ao usuário monitorar o vídeo capturado em tempo real, pausar e retomar a análise, e visualizar o gráfico que acompanha o percentual de ocorrência de cada uma das classes detectadas.
- **Classificação em tempo real**: a partir de imagens capturadas pela câmera, as folhas de soja são classificadas nas categorias "*Healthy*", "*Frogeye*"ou "*Yellow Mosaic*"com base em uma rede neural convolucional treinada.

3.5.2 Carregamento do modelo

Como apresentado na Seção 3.4.2, o modelo foi treinado previamente e salvo em arquivos separados: um arquivo JSON contendo a arquitetura do modelo e um arquivo HDF5 contendo os pesos. No script, esses arquivos são carregados e o modelo é recompilado para que possa ser utilizado nas previsões.

3.5.3 Processamento da Imagem

Cada *frame* capturado pela câmera é redimensionado para 250x250 pixels, normalizado e passado ao modelo para classificação. A função `process_frame` realiza essa tarefa, assegurando que a análise só é realizada se a certeza da classificação for superior a 75%. O resultado é exibido em tempo real na interface.

3.5.4 Estrutura da Interface

A interface foi desenvolvida utilizando Tkinter, que oferece uma maneira simples de criar janelas, botões e elementos gráficos. O vídeo ao vivo capturado pela câmera é exibido diretamente na janela, enquanto o resultado da análise é apresentado logo abaixo. Além disso, os resultados das classificações são registrados ao longo do tempo e exibidos em um gráfico dinâmico, utilizando o `matplotlib` integrado ao Tkinter por meio do `FigureCanvasTkAgg`. Este gráfico mostra as porcentagens de ocorrência das três classes analisadas: "Healthy", "Frogeye"ou "Yellow Mosaic". O sistema permite também que o usuário pause ou retome a análise através de botões. A figura 22 exibe o resultado final da interface desenvolvida.

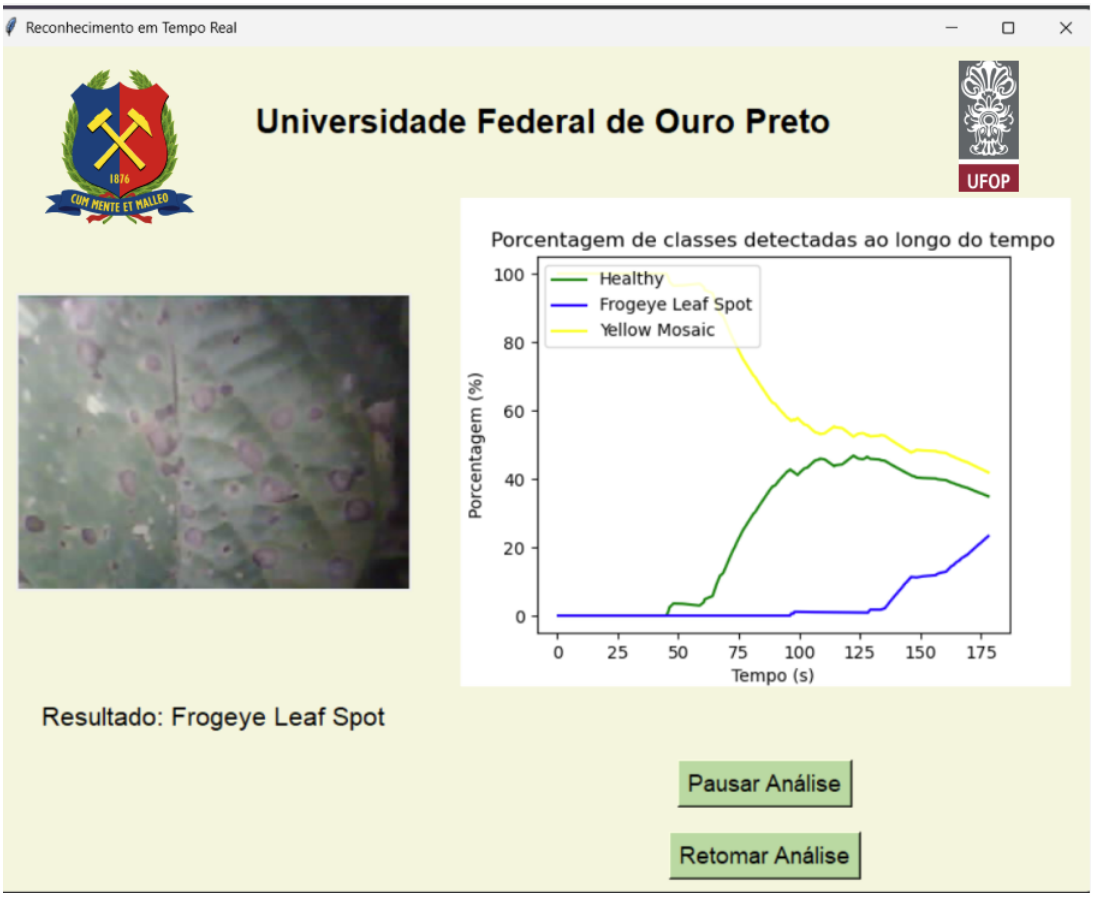

Figura 22 – Interface. Fonte: Elaboração própria.

3.6 Estrutura Final do Projeto

A figura 23 ilustra o fluxo de trabalho do sistema desenvolvido, utilizando uma ESP32-CAM conectada a um computador central para a análise das imagens em tempo real. O processo se inicia com a captura de imagens pela ESP32-CAM, que transmite os dados via Wi-Fi para o computador. O computador, por sua vez, executa um modelo de rede neural convolucional previamente treinado, capaz de classificar as folhas em três categorias: saudáveis, com Mancha Olho-de-Rã ou com Mosaico Amarelo.

Após a análise, os resultados são exibidos em uma interface gráfica desenvolvida em Python, onde o usuário pode visualizar tanto a imagem capturada quanto as informações sobre a classificação.

Esse fluxo integrado permite que o sistema realize a classificação de imagens de forma ágil e precisa, utilizando uma abordagem de captura em tempo real e oferecendo uma solução prática para o monitoramento de doenças em lavouras.

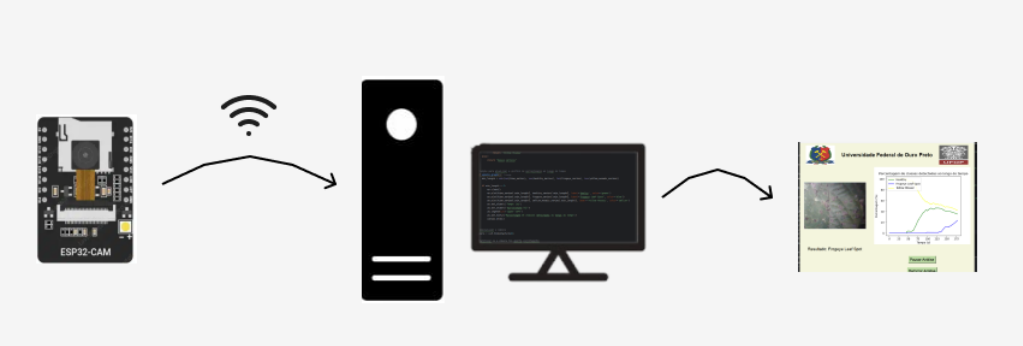

Figura 23 – Estrutura do projeto. Fonte: Elaboração própria.

4 Resultados

Neste capítulo, apresentam-se os resultados obtidos com o modelo de classificação desenvolvido para identificar três categorias de folhas de soja: saudáveis, afetadas por Mancha Olho-de-Rã e afetadas por Mosaico Amarelo. A avaliação do desempenho do modelo foi realizada utilizando as métricas de precisão, recall e f1-score. A Tabela 3 resume os principais resultados para cada classe.

Classe	Precisão	Recall	F1-Score
Frogeye Leaf Spot	87%	65%	74%
Healthy	81%	85%	83%
Yellow Mosaic	83%	100%	91%
Acurácia Total	**83%**		
Média Macro	**84%**	**83%**	**83%**
Média Ponderada	**84%**	**83%**	**83%**

Tabela 3 – Resultados da CNN por classe. Fonte: Elaboração própria.

Pode-se observar que o modelo apresentou uma precisão geral de 83%, indicando um bom desempenho na maioria das amostras. No entanto, a classe *Frogeye Leaf Spot* apresentou um recall mais baixo, de 65%, o que sugere que algumas amostras dessa categoria foram incorretamente classificadas, como também evidenciado na matriz de confusão 24.

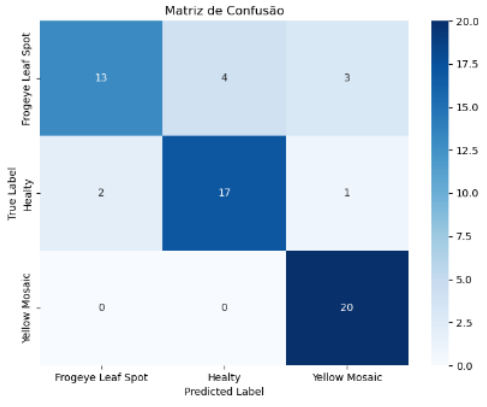

Figura 24 – matriz de confusão. Fonte: Elaboração própria.

Durante o desenvolvimento, foram realizadas variações quanto ao número de kernels (8, 16 e 32), permitindo observar uma diferença no desempenho do modelo. Definindo o número de kernels para 8, o modelo apresentou boa precisão para a classe *Frogeye Leaf Spot*, mas cometeu mais erros na classe *Healthy*. Com 16 kernels, o desempenho da classe

Healthy melhorou significativamente, mas o modelo passou a ter mais dificuldade em classificar corretamente a classe *Frogeye Leaf Spot*. Por fim, com 32 kernels, o modelo apresentou um desempenho mais equilibrado entre as classes, mostrando uma melhor capacidade de generalização. A classe *Yellow Mosaic* manteve um desempenho constante em todas as configurações, sendo sempre classificada corretamente. Esses resultados demonstram que o ajuste do número de kernels impacta diretamente a capacidade de o modelo, a Figura 25 apresenta os resultados obtidos para as diferentes classes.

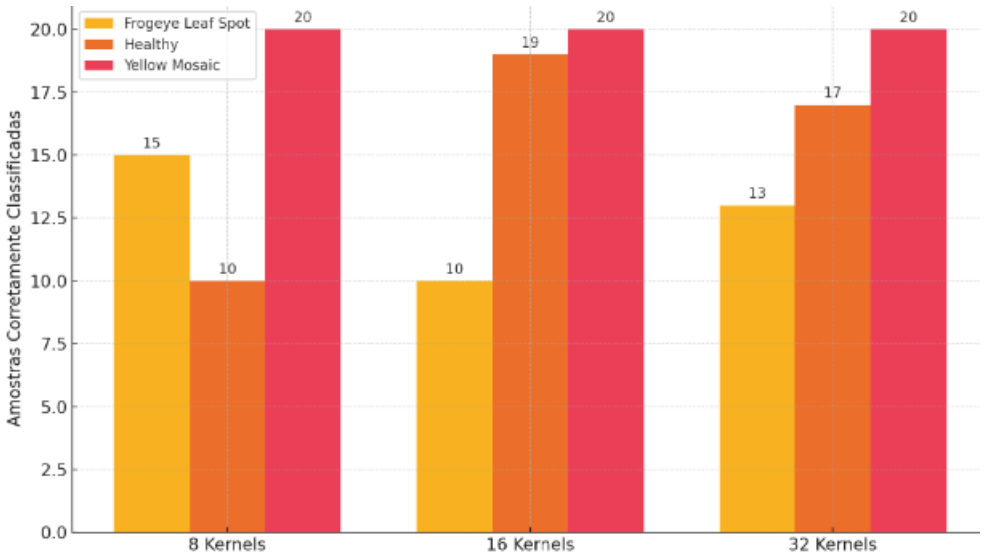

Figura 25 – Desempenho Do Modelo Por Número De Kernels. Fonte: Elaboração própria.

Ao utilizarmos imagens capturadas pela ESP32-CAM para a classificação, observou-se uma diminuição na acertividade do modelo. Essa queda pode ser atribuída à qualidade inferior das imagens fornecidas pela câmera do microcontrolador, que possui uma resolução mais baixa e pode gerar ruídos visuais que afetam a correta identificação dos padrões. Embora o modelo tenha apresentado um desempenho satisfatório com imagens de alta qualidade, a utilização de imagens de menor resolução impactou negativamente os resultados, mostrando que o modelo é sensível à qualidade das entradas.

No entanto, é importante destacar que, com a construção de uma base de dados composta por imagens capturadas diretamente pela ESP32-CAM, o desempenho do modelo poderia ser aprimorado. Ao treinar o modelo com dados mais próximos da realidade de aplicação no campo, ele seria capaz de se ajustar melhor às condições de captura e realizar classificações mais precisas, mesmo em cenários com qualidade de imagem reduzida.

Além disso, o modelo desenvolvido apresenta grande potencial de escalabilidade. A arquitetura da CNN utilizada permite que novas classes de doenças ou outras variáveis sejam incorporadas ao sistema sem a necessidade de reestruturar completamente o modelo. Isso facilita sua adaptação a diferentes necessidades e contextos futuros, ampliando as possibilidades de uso.

Por fim, o projeto pode ser facilmente adaptado para outras áreas da automação agrícola, como a detecção de pragas, monitoramento do crescimento de plantas e avaliação da qualidade de frutos. Fora do setor agrícola, essa tecnologia tem potencial para ser utilizada na automação industrial, por exemplo, no monitoramento da qualidade de produtos em linhas de produção, inspeção em gasodutos e correias transportadoras de minério, identificação de defeitos ou mesmo em sistemas de segurança, com a detecção de intrusos ou eventos anômalos. A flexibilidade e o potencial de escalabilidade do sistema tornam-no aplicável a diversas áreas além da agricultura, o que reforça sua relevância no contexto da automação baseada em análise de imagens.

5 Conclusão

O presente trabalho desenvolveu e implementou um sistema de classificação de doenças em folhas de soja utilizando uma câmera ESP32-CAM para captura de imagens em tempo real, integrando uma Rede Neural Convolucional para a análise dessas imagens. O sistema foi capaz de identificar e classificar folhas saudáveis, com Mancha Olho-de-Rã ou com Mosaico Amarelo de forma eficiente, contribuindo para o monitoramento de doenças na agricultura de precisão.

A arquitetura da CNN, composta por três camadas convolucionais e três camadas densas, projetada para maximizar a acurácia na classificação das folhas de soja. A integração entre a ESP32-CAM e o modelo de rede neural possibilitou que o sistema realizasse a análise das imagens em tempo real, facilitando a identificação precoce das doenças, algo fundamental para o manejo agrícola eficiente.

Um dos maiores desafios enfrentados durante o desenvolvimento foi a impossibilidade de realizar os experimentos em uma lavoura real, o que limitou o acesso a uma maior diversidade de cenários práticos e a coleta de dados diretamente do campo. Para contornar essa dificuldade, foi necessário utilizar bases de dados públicas e imagens previamente coletadas, o que, apesar de limitar os experimentos em campo, não impediu a validação eficaz do modelo proposto.

Este trabalho não só demonstra a eficácia da aplicação de CNNs para o reconhecimento de padrões em imagens agrícolas, como também abre caminho para futuras pesquisas na área de monitoramento automatizado de plantações utilizando veículos aéreos não tripulados. A aplicação de tais tecnologias pode contribuir significativamente para o avanço da agricultura de precisão, oferecendo aos produtores uma ferramenta poderosa para o controle de doenças e otimização da produção.

Por fim, o sistema desenvolvido cumpre seu objetivo de promover uma solução prática e eficiente para a detecção de doenças em folhas de soja, com potencial para ser expandido para outras culturas e condições de campo, tornando-se uma importante ferramenta no contexto agrícola moderno.

6 Trabalhos Futuros

Este trabalho apresentou uma solução eficiente para a classificação de folhas de soja em tempo real, integrando a captura de imagens por uma ESP32-CAM e a utilização de redes neurais convolucionais para a análise das mesmas. No entanto, há várias possibilidades de expansão e melhoria do sistema proposto, algumas das quais são discutidas abaixo:

6.1 Desenvolvimento de uma Interface Web

A implementação de uma interface web permitiria uma maior flexibilidade no gerenciamento dos modelos de classificação. Uma aplicação web poderia ser utilizada para armazenar e acessar de forma centralizada as redes neurais treinadas e os pesos correspondentes. Isso facilitaria a integração de novas versões de modelos, possibilitando atualizações constantes e o gerenciamento de diferentes modelos de acordo com as necessidades específicas de cada cenário de uso. Além disso, essa interface poderia oferecer uma visualização mais amigável dos resultados das classificações e relatórios de performance, além de permitir o compartilhamento e versionamento dos modelos de rede.

6.2 Utilização de Autoencoders

Uma outra direção promissora para futuros trabalhos seria a integração de *autoencoders*. Os *autoencoders* são redes neurais capazes de aprender uma representação compacta (ou codificação) de um conjunto de dados, sendo úteis para identificar padrões intrínsecos nas imagens antes mesmo de realizar a classificação. A introdução de um *autoencoder* poderia permitir uma pré-análise dos dados, destacando características relevantes das folhas, como texturas ou padrões de coloração associados a doenças. Essa abordagem poderia aumentar a robustez e a precisão do modelo, melhorando a capacidade de reconhecimento das imagens, mesmo em cenários de baixa qualidade ou com variações de iluminação, como é o caso da utilização da ESP32-CAM.

Essas melhorias não só aprimorariam a funcionalidade do sistema, como também abririam portas para novas aplicações em outras áreas da agricultura de precisão, onde a análise em tempo real de imagens é uma ferramenta essencial para tomada de decisão.

Referências

BALASTREIRE, L. A. *Agricultura de precisão: conceito, ferramentas e tecnologias*. São Paulo: Ed. da Universidade de São Paulo, 2000. Citado 0 vez na página 12.

BHUJADE, V. *Soybean Leaf Dataset for Disease Classification*. Kaggle. 2022. Disponível em: https://www.kaggle.com/datasets/vaishaligbhujade/soybean-leaf-dataset-for-disease-classification. Acesso em: 23 set. 2024. Citado 1 vez nas páginas 29–31, 33.

BORGES, G. A. et al. *Desenvolvimento com microcontroladores Atmel AVR*. por Computador, 2006. Citado 0 vez na página 17.

DRAELOS, Rachel. *A short history of convolutional neural networks*. 2019. Disponível em: https://glassboxmedicine.com/2019/04/13/a-short-history-of-convolutional-neural-networks/. Acesso em: 21 set. 2024. Citado 1 vez na página 25.

ESPRESSIF SYSTEMS. *ESP32 Technical Reference Manual*. 2024. Acesso em: 22 set. 2024. Disponível em: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf. Citado 0 vez na página 28.

FREEMAN, K. M.; SILVA, A. M.; PEREIRA, J. A. Veículos aéreos não tripulados: monitoramento agrícola e seus impactos na eficiência produtiva. *Agricultural Journal*, 2015. Citado 1 vez na página 12.

GEBBERS, R.; ADAMCHUK, V. I. Precision Agriculture and Food Security. *Science*, v. 327, n. 5967, p. 828–831, 2010. DOI: 10.1126/science.1183899. Citado 0 vez na página 17.

GONZALEZ, Rafael C.; WOODS, Richard E. *Digital Image Processing*. 4. ed.: Pearson, 2017. Citado 2 vezes nas páginas 18, 19.

HAYKIN, S. *Redes Neurais: Princípios e Prática*. Bookman Editora, 2001. Citado 0 vez na página 19.

HENNING, A. A. et al. *Manual de identificação de doenças de soja*. Embrapa, 2014. Citado 4 vezes nas páginas 16, 17.

HEXAGON. *Como a automação agrícola está mudando o agronegócio*. 2021. Disponível em: https://hexagon.com/pt/resources/resource-library/how-farming-automation-changing-agriculture. Acesso em: 18 set. 2024. Citado 1 vez na página 12.

JORGE, L. D. C.; INAMASU, R. Y. Uso de veículos aéreos não tripulados (VANT) em agricultura de precisão. In. Citado 1 vez na página 17.

LECUN, Y.; BENGIO, Y.; HINTON, G. Aprendizagem profunda. *Natureza*, v. 521, p. 436–444, 2015. Citado 1 vez na página 24.

LECUN, Yann; BOTTOU, Leon et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, IEEE, v. 86, n. 11, p. 2278–2324, 1998. Citado 0 vez na página 24.

MAKERHERO. *Módulo ESP32-CAM com câmera OV2640 2MP*. 2024. Disponível em: https://www.makerhero.com/produto/modulo-esp32-cam-com-camera-ov2640-2mp/. Acesso em: 22 set. 2024. Citado 0 vez na página 27.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943. Citado 0 vez na página 19.

PARKER, J. R. *Algorithms for image processing and computer vision*. John Wiley & Sons, 2010. Citado 1 vez na página 20.

RAHMAN, S. U. et al. A source of resistance against yellow mosaic disease in soybeans correlates with a novel mutation in a resistance gene. *Frontiers in Plant Science*, v. 14, p. 1230559, 2023. DOI: 10.3389/fpls.2023.1230559. Citado 2 vezes nas páginas 15, 16.

RAUBER, T. W. *Redes neurais artificiais*. 29: Universidade Federal do Espírito Santo, 2005. Citado 0 vez nas páginas 19, 20.

REIS, E. M. A mancha foliar "olho de rã"em soja. *Lavoura Arrozeira*, Porto Alegre, v. 27, n. 279, p. 4–8, 1974. Disponível em: https://www.infoteca.cnptia.embrapa.br/handle/doc/1164023. Acesso em: 2024. Citado 2 vezes na página 16.

SANTOS, E. *Revolução no campo: como a agricultura de precisão está transformando a produção brasileira*. 2024. Disponível em: https://www.noticiasagricolas.com.br/artigos/artigos-geral/378246-revolucao-no-campo-como-a-agricultura-de-precisao-esta-transformando-a-producao-brasileira.html. Acesso em: 18 set. 2024. Citado 3 vezes na página 12.

SETOSA. *Image Kernels*. Acesso em: 22 set. 2024. 2024. Disponível em: https://setosa.io/ev/image-kernels/. Citado 0 vez nas páginas 22–24.

SHAMSHIRI, R. et al. Research and development in agricultural robotics: A perspective of digital farming. *IJAB*, 2018. Citado 0 vez na página 17.

SILVA, F. et al. *Soja: do plantio à colheita*. Oficina de Textos, 2022. Citado 0 vez na página 14.

USDA, Foreign Agricultural Service -. *Soybean 2024 world production*. Disponível em: https://ipad.fas.usda.gov/cropexplorer/cropview/commodityView.aspx?cropid=2222000. Acesso em: 11 set. 2024. Citado 1 vez nas páginas 14, 15.

VIANA, G.; MOREIRA, R.; CASTRO, J. Aplicações de VANTs na agricultura de precisão: monitoramento de lavouras e uso sustentável de insumos. *Journal of Precision Agriculture*, 2018. Citado 0 vez na página 12.

ZHANG, A. et al. *Dive into deep learning*. 2021. arXiv: 2106.11342 [cs.LG]. Citado 1 vez na página 25.