UNIVERSIDADE FEDERAL DE ITAJUBÁ

FRUIT QUALITY ASSESSMENTS
MACHINE LEARNING ENGINEERING

ALEX ALVAREZ DUQUE - 2024005730
ARTUR GOMES SIMÃO - 2024006513
PEDRO LUCAS PEREIRA FERREIRA - 2024005982
IESTI-UNIFEI

Itajubá, 17 de junho de 2025

SUMÁRIO

1. RESUMO

Um sistema inteligente de classificação de frutas e vegetais foi desenvolvido com o objetivo de identificar automaticamente o estado de conservação dos alimentos como verde, maduro ou estragado, utilizando técnicas de aprendizado de máquina. A solução foi projetada para rodar em tempo real em dispositivos de baixo custo, como o Raspberry Pi Zero 2W, promovendo automação acessível e eficiente na triagem de alimentos.

O modelo escolhido foi o MobileNetV2, quantizado para int8 e otimizado com TensorFlow Lite, garantindo baixo consumo de memória e alta velocidade de inferência (5 ms). A base de dados foi composta por imagens públicas e próprias, abrangendo cinco frutas: maçã, banana, manga, laranja e tomate. O sistema alcançou acurácia de 82,5% e AUC de 0,98, com desempenho robusto mesmo em condições de hardware limitadas.

A aplicação inclui captura de imagem, pré-processamento, inferência e exibição dos resultados via interface web. Os testes demonstraram alta precisão para frutas frescas e estragadas, com desempenho ligeiramente inferior para frutas verdes, devido à variabilidade visual. A proposta se mostra promissora para aplicações em agricultura inteligente, controle de qualidade e redução de desperdícios na cadeia alimentar.

O repositório do projeto pode ser encontrado no github utilizando o link: https://github.com/12FlyBreads/fruit-quality-assessment.git .

.

2. INTRODUÇÃO

Este projeto tem como foco o desenvolvimento de um sistema inteligente de avaliação visual para a classificação de frutas e vegetais (maçã, banana, manga, laranja e tomate) quanto ao seu estado de conservação - frescas, maduras ou podres. A proposta visa automatizar o processo de triagem de frutas por meio de técnicas de aprendizado de máquina embarcadas em dispositivos de baixo custo, tais como o Raspberry Pi Zero, promovendo maior eficiência e padronização na separação desses alimentos.

A avaliação da qualidade das frutas e vegetais é uma etapa fundamental na cadeia de produção e distribuição de alimentos, influenciando diretamente a experiência do consumidor, a gestão dos estoques e a redução de desperdícios. Métodos tradicionais, como a inspeção visual manual, apresentam limitações significativas, incluindo subjetividade, inconsistência entre operadores e elevado consumo de tempo. Esses fatores comprometem a confiabilidade do processo e dificultam sua escalabilidade. Segundo a Organização das Nações Unidas para Agricultura e Alimentação (FAO), cerca de 45% das frutas e hortaliças produzidas globalmente são perdidas ou desperdiçadas antes de chegar ao consumidor final. Grande parte dessas perdas ocorre durante as etapas de pós-colheita, transporte e comercialização, onde a falta de sistemas precisos de avaliação de qualidade contribui para o descarte prematuro de alimentos que ainda poderiam ser aproveitados.

Diante desse cenário, torna-se evidente a necessidade de soluções tecnológicas capazes de aprimorar os processos de triagem e classificação de frutas e vegetais, especialmente em ambientes com recursos limitados. A adoção de sistemas inteligentes de avaliação visual, baseados em aprendizado de máquina e embarcados em plataformas de baixo custo, representa uma alternativa promissora para reduzir perdas, aumentar a eficiência operacional e garantir maior padronização na seleção de alimentos. O desenvolvimento de uma solução com essas características contribuem diretamente para a modernização da cadeia produtiva e para a promoção de práticas mais sustentáveis no setor alimentício.

Um exemplo prático dessa melhoria é a redução do descarte de frutas e vegetais em centros de distribuição, onde produtos com pequenas imperfeições visuais costumam ser rejeitados, mesmo estando próprios para consumo. Com a aplicação de sistemas inteligentes de avaliação visual, é possível realizar uma triagem mais precisa e padronizada, aproveitando melhor os alimentos e contribuindo para a redução do desperdício.

3. REVISÃO BIBLIOGRÁFICA

3.1. CONCEITOS BÁSICOS

O Machine Learning (Aprendizado de Máquina) é uma área da inteligência artificial que permite que sistemas computacionais aprendam padrões a partir de dados, sem a necessidade de programação explícita para a execução de tarefas específicas. Esse processo envolve a coleta de dados, o treinamento do modelo com os dados coletados, a validação ou teste do modelo para avaliar seu desempenho e, por fim, a aplicação em novos dados para realizar previsões ou classificações. Um exemplo de aplicação é o sistema de filtragem de e-mails, no qual algoritmos aprendem a identificar mensagens de spam analisando padrões presentes em mensagens anteriores.

Dentro do Machine Learning, o Deep Learning (Aprendizado Profundo) é uma subárea que utiliza redes neurais profundas para extrair e aprender automaticamente características complexas de grandes volumes de dados. Diferentemente do aprendizado tradicional, o Deep Learning dispensa a engenharia manual de características, uma vez que a própria rede identifica padrões relevantes. Entretanto, redes profundas requerem considerável poder computacional e grandes quantidades de dados para seu treinamento eficiente. Entre suas aplicações destacam-se o reconhecimento de voz, a tradução automática e a detecção de objetos em imagens.

As Redes Neurais Convolucionais (Convolutional Neural Networks – CNNs) são um tipo específico de rede neural projetada para processar dados com estrutura espacial, como imagens e vídeos. As CNNs realizam a extração de características de forma hierárquica, sendo compostas por camadas convolucionais, que extraem padrões locais como bordas, texturas e formas; camadas de pooling, que reduzem a dimensionalidade preservando informações relevantes; e camadas totalmente conectadas, que realizam a decisão final, como a classificação da imagem. As CNNs são amplamente utilizadas em tarefas de visão computacional, incluindo classificação de imagens, reconhecimento facial e detecção de objetos.

No contexto de análise de imagens, é importante diferenciar classificação e detecção de objetos. A classificação consiste em determinar o conteúdo predominante de uma imagem, atribuindo-lhe uma ou mais classes correspondentes. Por exemplo, uma fotografia contendo um cachorro será identificada apenas como pertencente à classe “cachorro”. Já a detecção de objetos envolve não apenas identificar o que está presente na imagem, mas também determinar a localização exata de cada objeto, geralmente representada por caixas delimitadoras. Em uma fotografia com dois cachorros e um gato, o modelo identifica cada animal individualmente, atribuindo a cada um a classe correspondente e sua posição na imagem.

3.2. TRABALHOS RELACIONADOS E APLICAÇÕES PRÁTICAS

A avaliação da qualidade e o monitoramento da maturação de frutas têm recebido crescente atenção na literatura científica, em função da importância desses processos para a redução de perdas pós-colheita e para a melhoria da produtividade agrícola. Diversos estudos demonstram a aplicação de tecnologias emergentes, incluindo Internet das Coisas (IoT), aprendizado profundo e sistemas de visão computacional, associadas a práticas de manejo agrícola.

Um estudo recente propôs um sistema de IoT para monitoramento da maturação de frutas, integrando sensores de luz e sensores de gás etileno. Os sensores de luz capturam alterações na coloração e pigmentação das frutas, enquanto os sensores de etileno monitoram a produção deste hormônio, diretamente associado ao processo de amadurecimento. Os dados obtidos são

processados por meio de uma rede neural do tipo multilayer perceptron, permitindo prever com precisão o estágio de maturação. Essa abordagem possibilita decisões mais assertivas quanto ao armazenamento e à distribuição dos frutos, contribuindo para a redução de perdas e a otimização da cadeia produtiva.

Em paralelo, a utilização de modelos de visão computacional em tempo real tem se mostrado eficaz na classificação da qualidade de frutas. Um estudo aplicou a arquitetura YOLO (You Only Look Once), combinada a máquinas de vetor de suporte (SVM), para a classificação de laranjas. O modelo YOLO foi empregado na identificação de características visuais das frutas, enquanto o SVM realizava a classificação da qualidade. Os resultados indicaram que o YOLO apresentou desempenho superior em termos de precisão e velocidade de processamento, demonstrando a aplicabilidade de soluções de aprendizado profundo para avaliação automatizada de frutas em ambientes de produção agrícola.

Além das abordagens tecnológicas, a literatura destaca a importância do manejo adequado e do conhecimento técnico para a produção de frutas de alta qualidade. A publicação da Embrapa intitulada “Pequenas Frutas: o produtor pergunta, a Embrapa responde” apresenta informações abrangentes sobre cultivares, condições climáticas, práticas de manejo, controle de pragas e doenças, e estratégias de pós-colheita para pequenas frutas, como morango, mirtilo, amora-preta, framboesa, physalis e uva muscadínia. Este trabalho reforça a necessidade de integrar tecnologias de monitoramento com práticas agrícolas sustentáveis, garantindo a melhoria da produtividade e a preservação da qualidade dos frutos.

Em síntese, os estudos analisados evidenciam que a convergência entre IoT, redes neurais, visão computacional em tempo real e manejo agrícola representa uma estratégia eficiente para monitoramento e avaliação da qualidade de frutas, promovendo decisões mais assertivas, redução de perdas e aumento da eficiência na cadeia produtiva agrícola.

4. METODOLOGIA

4.1. ESCOPO DO PROJETO

O projeto tem como objetivo principal desenvolver um sistema baseado em aprendizado de máquina capaz de avaliar a qualidade de frutas e vegetais (maçã ‘apple’, banana, laranja ‘orange’, manga ‘mango’ e tomate ‘tomato’) por meio de inferência em tempo real de imagens de uma câmera.

A tarefa definida consiste em classificar o tipo da fruta ou vegetal (de maneira individual) e seu estado de maturação ou conservação do alimento em três classes principais: unripe (verde / imaturo), ripe (maduro) e rotten (estragado). A partir dessa avaliação, uma estimativa de tempo de vida da fruta ou vegetal também deve ser informada na aplicação.

A escolha por este problema se deve à sua relevância prática no contexto de agricultura de precisão, controle de qualidade e redução de desperdício de alimentos, além de se tratar de um desafio adequado para um ambiente de processamento limitado, como o Raspberry Pi Zero 2W.

O sistema busca demonstrar que modelos de visão computacional compactos podem ser aplicados com bom desempenho em dispositivos de borda, mantendo precisão e velocidade adequadas.

4.2. COLETA E PREPARAÇÃO DE DADOS

Para o treinamento e validação do modelo, foram utilizadas imagens provenientes de datasets públicos, complementados por imagens capturadas pela equipe e imagens encontradas na web.

Os principais conjuntos de dados públicos considerados foram:

- **Kaggle – Fruits-360 Dataset**: ampla base de imagens de frutas e vegetais com alta diversidade de classes.
- **Kaggle – Fruit Ripeness Dataset**: base contendo frutas em diferentes estágios de maturação (unripe, ripe, overripe).
- **Roboflow – Custom Ripeness Datasets**: datasets com rótulos e formatos prontos para integração com YOLO/TensorFlow.

As imagens foram organizadas em pastas correspondentes a cada classe e divididas em três subconjuntos: 80% para treino e 20% para validação e teste, seguindo boas práticas de generalização de modelos.

O pipeline de preparação de dados envolveu as seguintes etapas:

- **Redimensionamento**: todas as imagens foram redimensionadas para uma resolução reduzida de 160x160 pixels, a fim de compatibilizar com as limitações do hardware.
- **Data Augmentation**: antes do treinamento do modelo da inteligência artificial, uma série de Data Augmentation foi aplicada nas imagens por meio do módulo do Roboflow. As augmentation escolhidas foram: Rotation, Shear, Saturation, Brightness e Exposure.

O dataset completo pode ser visualizado por meio do Roboflow, usando o seguinte link: https://universe.roboflow.com/pencil-versus-scissors-teste/fruits-verification-thwfx .

Essas etapas foram realizadas utilizando o módulo do Edge Impulse e ferramentas complementares de processamento de imagem.

4.3. MODELOS E ARQUITETURAS

Para o treinamento e avaliação, foram considerados modelos leves e otimizados para inferência em dispositivos embarcados, buscando um equilíbrio entre desempenho e precisão.

Os modelos analisados foram:

- **MobileNetV2**: arquitetura eficiente, com baixo número de parâmetros e amplamente utilizada em aplicações de edge computing.
- **EfficientNet-Lite**: variação otimizada da EfficientNet para dispositivos móveis, com bom custo-benefício entre velocidade e acurácia.
- **YOLOv8-Nano (Ultralytics)**: modelo de detecção leve, avaliado em experimentos preliminares para identificar múltiplas frutas em uma mesma imagem.

Após testes iniciais, o modelo MobileNetV2 foi escolhido como base principal devido à sua facilidade de quantização, compatibilidade com TensorFlow Lite e bom desempenho em classificações de pequena escala.

A estrutura da rede utilizada é composta por camadas convolucionais profundas seguidas de blocos depthwise separable convolutions, reduzindo a complexidade computacional. A camada final é composta por uma ativação Softmax, responsável por indicar a probabilidade de cada classe de maturação.

Dessa forma, o modelo foi treinado com a base MobileNetV2, com 200 épocas de treinamento, um batch size de 32, learning rate de 0.0001 e aproximadamente 5000 imagens no dataset.

4.4. OTIMIZAÇÃO PARA O DISPOSITIVO

Com o modelo treinado e validado, a etapa seguinte consistiu na otimização para execução em tempo real no Raspberry Pi Zero 2W, cuja capacidade de processamento e memória é limitada.

Foram aplicadas as seguintes técnicas:

- **Quantização**: conversão do modelo de precisão float32 para int8, por meio de post-training quantization utilizando o TensorFlow Lite Converter. Essa técnica reduziu significativamente o tamanho do modelo e aumentou a velocidade de inferência, com perda mínima de acurácia.
- **Redução de resolução**: testes foram realizados com entradas 160x160 pixels, avaliando o impacto no desempenho e na qualidade da predição.
- **Conversão final**: o modelo otimizado foi exportado para o formato .lite, possibilitando sua execução no ambiente embarcado utilizando o interpretador TensorFlow Lite Runtime.

Essas otimizações permitiram alcançar uma execução fluida, com taxa de processamento superior a 0.5 FPS, atendendo aos requisitos mínimos de desempenho definidos pelo projeto.

4.5. FERRAMENTAS E AMBIENTE

O desenvolvimento e treinamento do sistema foram realizados com base em um conjunto de ferramentas modernas de aprendizado de máquina e visão computacional.

Frameworks utilizados:

- **TensorFlow Lite**: para otimização e inferência no dispositivo final.
- **OpenCV e Pillow (PIL)**: para captura, leitura e pré-processamento de imagens.

Hardware:

- **Raspberry Pi Zero 2W**, equipado com processador Quad-Core 1GHz e 512 MB de RAM, utilizado como plataforma de inferência.
- **Câmera Raspberry Pi v2 (8 MP)**, empregada na captura de imagens em tempo real.

Ambientes de desenvolvimento:

- **VSCode**: ambiente de programação principal.
- **Edge Impulse:** plataforma utilizada para coleta, processamento, anotação e conversão do dataset, além de possibilitar o gerenciamento de modelos e experimentos para dispositivos embarcados.
- **GitHub:** repositório utilizado para controle de versão, colaboração em equipe e armazenamento do código-fonte, garantindo rastreabilidade das alterações e integração contínua entre os membros do projeto.

A integração entre essas ferramentas garantiu uma pipeline contínua desde o treinamento em nuvem até a implantação do modelo no dispositivo de borda.

5. IMPLEMENTAÇÃO

5.1. INSTALAÇÃO

A instalação do sistema é realizada de forma automatizada por meio de um script Bash (setup.sh), que cria o ambiente virtual, instala as dependências necessárias e prepara o modelo para execução. O passo a passo de instalação está disponível no repositório do GitHub do projeto: https://github.com/12FlyBreads/fruit-quality-assessment.git.

Além disso, são necessários pré-requisitos como Python 3.9+, um computador de bordo com LINUX e um módulo de câmera na Raspberry Pi.

Caso a instalação por meio do script dê algum erro, é recomendado fazer a criação do ambiente virtual e instalação das bibliotécas de forma manual. Os comandos são o seguinte:

- python3 -m venv .fnvq-env –system-packages : criar ambiente virtual
- source .fnvq-env/bin/activate : ativar ambiente virtual
- pip install <nome-do-pacote> : para instalar cada biblioteca utilizada

É importante salientar também que o computador de bordo deve estar atualizado préviamente.

5.2. ESTRUTURA GERAL DO SISTEMA

O sistema segue uma arquitetura modular, onde cada componente desempenha uma função específica, garantindo escalabilidade e manutenção facilitada. As etapas da aplicação ocorrem da seguinte forma:

- Captura da câmera: a captura de frames pela camera é realizada pela função 'camera.py' com o auxilio de bibliotecas como PIL e OpenCV, além da biblioteca da câmera nativa da Raspberry Zero Pi 2W (PiCam);
- Pre-processamento: O pré-processamento das imagens capturadas pela câmera são redimensionadas e normalizadas na função 'classification.py' , que também desempenha outras funções muito importantes como a classificação e etiquetação do frame de exibição;
- Inferência: a inferência de classificação em tempo real é desempenhada pelo modelo de classificação localizado na pasta 'models/' do projeto por meio da função de classificação do 'classification.py';
- Display/Output: o display de stream de camêra é realizado por meio de uma interface web, utilizando o Flask, por meio da função 'app.py' (função que liga todos os pontos da aplicação); já o output de descrição da classificação é gerado por meio de uma função no 'fruit_info.py' onde está localizado o database sobre as frutas utilizadas no projeto.

Cada etapa é executada em sequência dentro do código principal (app.py), utilizando módulos dedicados localizados na pasta src/.

5.3. FLUXO DE EXECUÇÃO

O fluxo principal do sistema ocorre da seguinte forma:

- Inicialização dos módulos:
  - Carregamento do modelo TFLite e das bibliotecas.
  - Inicialização das classes auxiliares.

    - Verificação da câmera e inicialização do stream de vídeo.
- Captura da imagem:
    - Uma imagem é capturada via Picamera2.
    - O frame é convertido para o formato RGB e redimensionado para o tamanho de entrada da rede 640x480 (pode ser alterado no ‘utils.py’).
- Pré-processamento:
    - Normalização dos pixels.
    - Redimensionamento para corresponder à entrada do modelo (160x160 px).
- Inferência com modelo TFLite:
    - O modelo quantizado (.lite) é carregado via tflite_runtime.Interpreter.
    - A imagem é passada para o tensor de entrada e o resultado inferido.
    - O modelo retorna a probabilidade de cada classe: fruta + qualidade.
- Exibição dos resultados:
    - Caso o Flask esteja habilitado, uma interface web local exibe o status de qualidade e o tempo de vida útil estimado.

É novamente importante salientar que no modelo criado por esse projeto, as frutas e vegetais selecionados para funcionar na inteligência artificial foram: maça, banana, manga, tomate e laranja.

6. AVALIAÇÃO DE DESEMPENHO

6.1. MÉTRICAS UTILIZADAS

A avaliação do modelo foi conduzida com base nas métricas clássicas de classificação como accuracy, precision, recall e F1-score, além de indicadores de desempenho embarcado, como tempo de inferência, uso de memória e uso de flash.

Essas métricas permitem analisar não apenas a eficácia da classificação, mas também a eficiência operacional em dispositivos de borda.

A seguir está uma imagem retirada do Edge Impulse sobre o desempenho do modelo:

Figura 1: análise de desempenho do modelo.

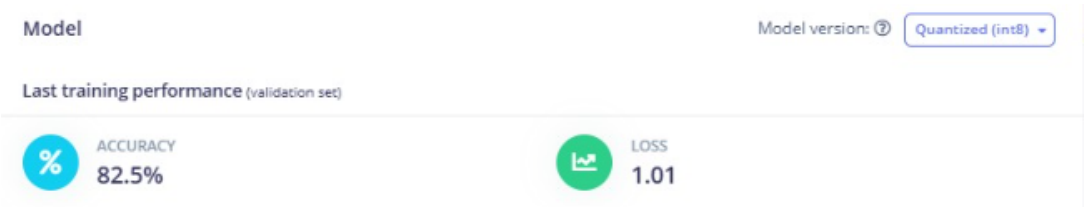

Figura 2: precisão e perda do modelo treinado.

Em geral, o modelo apresentou um bom equilíbrio entre precisão e recall, indicando uma boa capacidade de generalização entre diferentes classes de frutas e estágios de maturação.

6.2. TESTES NO RASPBERRY ZERO PI

O modelo quantizado (ei-fruits-detector-classifier-tensorflow-lite-int8) foi executado no Raspberry Pi Zero 2W, utilizando o runtime TensorFlow Lite (tflite-runtime). A seguir está uma captura de tela sobre os resultados de inferência:

Figura 3: performance no dispositivo de bordo.

Esses resultados indicam excelente eficiência computacional, especialmente considerando o hardware de baixo consumo. O modelo é capaz de realizar inferenciações praticamente em tempo real, com consumo mínimo de memória.

6.3. TESTES NO RASPBERRY ZERO PI

A matriz de confusão do conjunto de validação mostra o desempenho detalhado entre classes de frutas e estados de maturação. A figura a seguir mostra o gráfico de confusão gerado pelo treinamento no Edge Impulse:

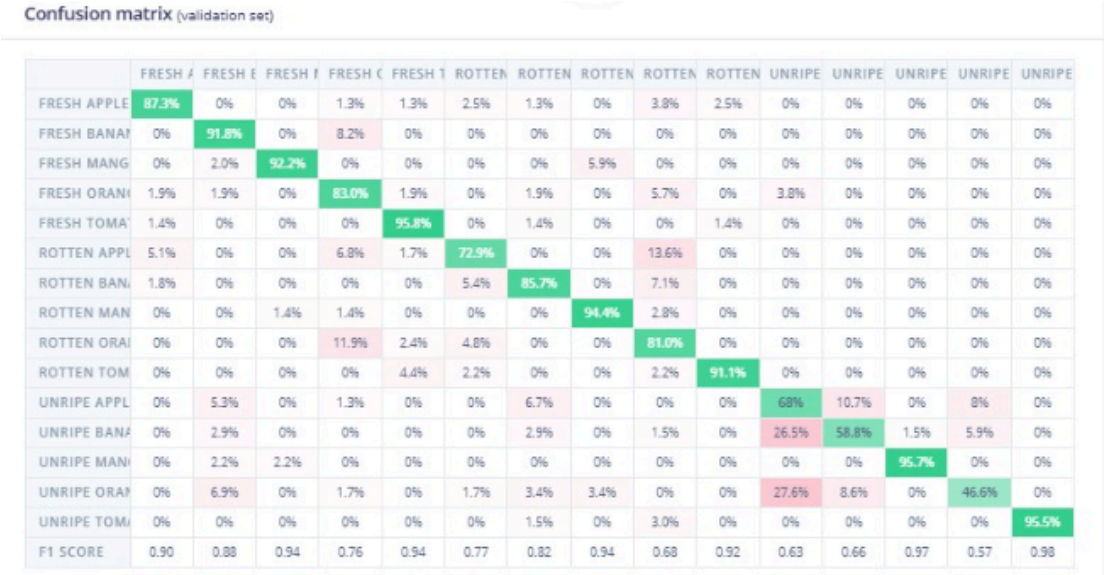

Figura 4: matriz de confusão do modelo treinado.

Observa-se que as frutas e vegetais foram classificados com alta precisão em seus estágios fresh e rotten. Já no estado unripe, o modelo acabou retornando uma precisão mais baixa se comparada ao todo, mas mesmo assim ainda gerando resultados bem sólidos.

6.4. DISCUSSÃO DOS RESULTADOS

O desempenho geral do modelo treinado e quantizado foi considerado satisfatório para aplicações embarcadas, como no caso desse projeto na Raspberry Zero Pi 2W. A versão final int8 apresenta um ótimo custo-benefício entre precisão e eficiência. Sobre isso, algumas observações podem ser realizadas:

- O tempo médio de inferência (12 ms), que é incrivelmente baixo, permite operações em tempo real no Raspberry Pi Zero 2W.

- A acurácia ponderada de 82.5% se torna aceitável, considerando as restrições do dataset e as variações de iluminação e textura.
- A AUC = 0.98 indica excelente capacidade discriminativa global.

Outra discussão cabível de se abordar são os possíveis fatores que impactaram no desempenho do modelo treinado, que estão relacionados às imagens coletadas para o dataset. Alguns possíveis fatores são a escolha de imagens com iluminação muito distintas, a variação de frutas da mesma espécie no mesmo rótulo do dataset e o fundo e enquadramento das fotos utilizadas. Todos esses fatores podem ter influenciado no erro e na confusão da classificação do modelo sobre as frutas.

Podemos assim concluir que mesmo com limitações de hardware, o sistema demonstrou ser capaz de executar classificações rápidas e confiáveis. A arquitetura MobileNetV2 quantizada provou-se adequada para Edge AI de baixo custo, abrindo caminho para aplicações reais em agricultura inteligente, automação e monitoramento de qualidade alimentar.

7. CONCLUSÃO

7.1. CONCLUSÃO GERAL

Depreende-se, portanto, que o projeto demonstrou ser viável a implementação de um sistema inteligente de classificação de frutas e vegetais com boa precisão, atingindo 82,5% de acurácia, por meio de modelos leves de aprendizado de máquina embarcados em dispositivos de baixo custo, como o Raspberry Pi Zero.

Além disso, a arquitetura MobileNetV2, após processo de quantização, apresentou desempenho satisfatório, conciliando eficiência computacional com acurácia adequada para aplicações em tempo real, mesmo em ambientes com recursos limitados.

Dessa forma, a solução proposta contribui significativamente para a modernização da cadeia produtiva agrícola, promovendo maior padronização, redução de desperdícios e viabilidade de automação em processos de triagem e controle de qualidade de alimentos.

7.2. DIFICULDADES ENCONTRADAS

Durante o desenvolvimento do projeto, a principal dificuldade enfrentada esteve relacionada à etapa de treinamento do modelo, especialmente na classificação de frutas no estágio unripe. Inicialmente, o modelo apresentava baixa precisão para essa classe, o que comprometia a confiabilidade da inferência. Para mitigar esse problema, foi aplicada a técnica de data augmentation por meio da plataforma Roboflow, contribuindo para melhorar a generalização e o desempenho do modelo.

Por outro lado, a montagem do código e a estruturação do sistema ocorreram de forma fluida, sem grandes obstáculos. A arquitetura modular e o uso de ferramentas como TensorFlow Lite facilitaram a implementação e a integração dos componentes.

No entanto, durante a construção do dataset, observou-se uma limitação na disponibilidade de imagens públicas com variedade suficiente para algumas frutas, especialmente em diferentes estágios de maturação. Essa escassez exigiu complementação manual com imagens capturadas pela equipe e buscadas na web, o que demandou tempo adicional e atenção à curadoria e à qualidade dos dados.

7.3. MELHORIAS FUTURAS

Apesar dos resultados satisfatórios obtidos, é necessário que melhorias futuras sejam realizadas para que o sistema tenha cada vez mais capacidade de classificar frutas e vegetais de diversas espécies com maior precisão. Nesse sentido, é fundamental expandir o conjunto de dados com imagens mais variadas e bem rotuladas, especialmente para frutas da classe “unripe”, que apresentaram maior taxa de erro na classificação.

Outra possível melhoria a ser implementada para melhorar o nível de precisão das frutas e qualidades seria trocar o modelo de classificação por um modelo mais robusto de detecção. Apesar desse tipo de modelo deixar a inferência um pouco mais demorada, a precisão em ambientes reais se tornaria um pouco mais relevante.

Ademais, recomenda-se a implementação de técnicas de segmentação para melhorar a detecção em imagens com múltiplos objetos. Essa abordagem pode aumentar significativamente o desempenho e a confiabilidade do sistema em cenários mais complexos.

Por fim, é importante aprimorar a interface do sistema, tornando-a mais interativa e responsiva, com dashboards informativos, gráficos de maturação e funcionalidades de exportação de dados para plataformas de gestão agrícola. Essas melhorias são essenciais para que o sistema de classificação evolua continuamente e contribua de forma efetiva com as necessidades reais do setor agrícola.

8. REFERÊNCIAS

FAO — Food and Agriculture Organization of the United Nations. *The State of Food and Agriculture: Food Loss and Waste*. Roma: FAO, 2019. Acesso em: 17 jun. 2025.

HOWARD, A. *et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. arXiv preprint arXiv:1704.04861, 2017. Acesso em: 17 jun. 2025.

FLASK Documentation. *Flask Web Framework*. Disponível em: https://flask.palletsprojects.com/. Acesso em: 17 jun. 2025.

ABADI, M. *et al.* TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Google, 2016. Disponível em: https://www.tensorflow.org/. Acesso em: 17 jun. 2025.

BRADBURY, J. *et al. OpenCV Documentation.* Open Source Computer Vision Library, 2024. Disponível em: https://opencv.org/. Acesso em: 17 jun. 2025.

EDGE IMPULSE. *Edge Impulse Platform – Model Training and Deployment for Edge AI*. Disponível em: https://www.edgeimpulse.com/. Acesso em: 17 jun. 2025.

KAGGLE. *Fruits-360 Dataset*. Disponível em: https://www.kaggle.com/datasets/moltean/fruits. Acesso em: 17 jun. 2025.

PILLOW Developers. *Pillow (PIL Fork) Documentation*. Disponível em: https://python-pillow.org/. Acesso em: 17 jun. 2025.

RASPBERRY PI FOUNDATION. *Raspberry Pi Zero 2W — Technical Specifications*. Disponível em: https://www.raspberrypi.com/. Acesso em: 17 jun. 2025.

ROBOFLOW. *Custom Ripeness Datasets*. Disponível em: https://universe.roboflow.com/pencil-versus-scissors-teste/fruits-verification-thwfx. Acesso em: 17 jun. 2025.