REVISTA DE TECNOLOGIA APLICADA (RTA)
v.13, n.1, Jan-Abr 2024, p. 51-65.
ISSN: 2237-3713

http://dx.doi.org/10.48005/2237-3713rta2024v13n1p5165

Inteligência artificial para automação de caixa no setor de hortifrúti*

Artificial intelligence for checkout automation in the grocery sector

Lucas Taroco Beraldo
Universidade Federal de Santa Catarina
lucas.taroco.beraldo@grad.ufsc.br

Pedro Henrique Centenaro
Universidade Federal de Santa Catarina
pedro.centenaro@grad.ufsc.br

Thiago José Bechtold
Universidade Federal de Santa Catarina
thiago.j.bechtold@grad.ufsc.br

Dr. Alex Fabiano Bueno
Universidade Federal de Santa Catarina
alex.bueno@ufsc.br

RESUMO ESTRUTURADO

O tempo gasto no processo de checkout em caixas de mercados é um problema que afeta tanto os clientes quanto o estabelecimento. Este artigo aborda o desenvolvimento de um processo automatizado destinado a aprimorar a seleção de produtos do setor de hortifrúti no contexto de um caixa de supermercado, visando proporcionar maior acessibilidade e eficiência aos usuários. Através da implementação de modelos de inteligência artificial integrado com um sistema de pesagem e interface gráfica, obteve-se uma taxa de acerto de 95,71% para o melhor modelo, dessa forma verificou-se a viabilidade do uso da aplicação de algoritmos de inteligência artificial para a classificação de produtos hortifruti.

Palavras-chave: Inteligência Artificial, Automatização, C, Python, Supermercados, EfficientNetB0, ResNet-50.

STRUCTURED ABSTRACT

The time spent in the checkout process at supermarket cash registers is a problem that affects both customers and the establishment. This article addresses the development of an automated process aimed at enhancing the selection of products in the fruits and vegetables section within the context of a supermarket checkout, aiming to provide greater accessibility and efficiency to users. Through the implementation of artificial intelligence models integrated with a weighing system and graphical interface, an accuracy rate of 95.71% was achieved for the best model, thus demonstrating the viability of using artificial intelligence algorithms for the classification of fruits and vegetables products.

* Received 02 dezember 2023; accepted in 01 July 2024; published online 05 Agust 2024.

Keywords: Artificial Intelligence, Automation, C, Python, Supermarkets, EfficientNetB0, ResNet-50.

1. INTRODUÇÃO

A proposta deste projeto surge pela identificação de uma problemática presente em diversos supermercados: o tempo perdido durante a compra de alimentos do setor de hortifruti. Nos mais comuns métodos atuais de compra, é necessário que o operador de caixa identifique o produto e insira seu código. Há ainda caixas do tipo *self-checkout*, onde o cliente deve, dentre diversas opções, selecionar o produto que está levando consigo. Ambos os casos elevam o tempo para efetuação da compra, fazendo desta categoria de produtos a mais demorada nos caixas, proporcionalmente à quantidade de produtos.

Tempo perdido é algo que desagrada clientes. Segundo Rinaldi [1], uma pessoa insatisfeita com o tempo de espera em filas de caixa pode não abandonar as compras no presente momento, porém sairá do mercado com uma imagem negativa que posteriormente pode refletir sobre a escolha de um novo local de compra. Sendo assim, a problemática atinge tanto os clientes quanto os pontos de venda.

Desta forma, o uso de inteligência artificial é visto como uma oportunidade de melhoria do sistema. Fazendo a identificação das frutas e legumes com tais ferramentas, é possível retornar ao usuário apenas os itens mais relevantes baseados na imagem captada, agilizando o processo de confirmação da compra.

2. REVISÃO BIBLIOGRÁFICA

2.1. Filas versus satisfação dos clientes

A dinâmica das filas nos estabelecimentos varejistas desempenha um papel fundamental na experiência do consumidor, influenciando diretamente sua satisfação e preferências futuras de compra. Como destacado por OPARA-NADI [2], a fase de checkout nos supermercados representa um ponto crítico nessa jornada do cliente, onde a eficiência e agilidade desempenham papéis cruciais. A análise comparativa entre os métodos de checkout tradicionais assistidos por atendentes e os sistemas de autoatendimento revela uma disparidade significativa no tempo médio de espera. Enquanto os caixas assistidos por atendentes demandam, em média, 5,92 minutos, os autoatendimentos consomem aproximadamente 11,22 minutos. Essa discrepância levanta questões fundamentais sobre a urgência da automação para otimizar os processos de checkout, visando oferecer uma experiência de compra mais ágil e eficiente aos consumidores.

2.2. Classificações utilizando CNNs

A utilização de redes neurais de aprendizado profundo não é uma novidade no processo de identificação e classificação de objetos e imagens. No entanto, o recente destaque das inteligências artificiais resultou em um aumento significativo de sua aplicação em diversos setores, introduzindo uma ferramenta inovadora para a automação de processos que, até pouco tempo atrás, eram exclusivamente executados por agentes humanos.

Como destacado por Dhanush et al. [3] a agricultura desempenha um papel crucial no avanço das nações. Entretanto, o crescimento exponencial da população mundial e os desafios ambientais em curso têm intensificado as dificuldades relacionadas à garantia de segurança alimentar. Diante dessa complexa conjuntura, a automação na agricultura emerge como uma solução viável para mitigar tais desafios. Este estudo se propõe a investigar a viabilidade da

aplicação de técnicas de visão computacional e algoritmos de inteligência na otimização dos processos produtivos agrícolas.

Com ênfase nos algoritmos de inteligência artificial, a pesquisa de Dhanush et al. [3], utiliza uma ampla gama de modelos para a identificação precisa de frutas por meio da análise de imagens. Dentre os modelos aplicados, destacam-se os resultados obtidos pelo algoritmo MASK - RCNN, que utiliza como rede neural *backbone* a ResNet, ResNet-50, e FPN, obtendo uma acurácia de 89.9%. Com base nesse e nos demais resultados obtidos, Danush et al [3]. chega à conclusão que a automação na produção agrícola desempenha um papel crucial no avanço desse setor. A integração efetiva de técnicas de visão computacional aliadas aos algoritmos de inteligência artificial representa uma verdadeira revolução na aquisição e análise de dados em tempo real. Em síntese, essas tecnologias combinadas possuem um potencial significativo para transformar radicalmente os métodos de cultivo e produção de alimentos, tornando-os mais eficientes, economicamente viáveis e sustentáveis.

Ainda sobre suas implementações em novos campos de atuação, de acordo com as observações de Rahman et al. [4], a identificação automatizada de frutas tem ganhado popularidade e se revela cada vez mais relevante. No entanto, sua implementação não é tão trivial, uma vez que a correta categorização dos produtos demanda considerações detalhadas sobre a localização, formato, cores e tamanho dos objetos em questão. Em consonância com essas considerações, realizou-se uma pesquisa visando avaliar a viabilidade dos algoritmos de aprendizado profundo na criação de um sistema automatizado capaz de identificar e categorizar frutas. Este estudo empregou um conjunto de 3240 imagens de alta qualidade pertencentes a 8 classes de frutas, como conjunto de treinamento para diversas redes neurais de aprendizado profundo já estabelecidas, tais como ResNet-50 e MobileNet.

Para avaliar a viabilidade das redes neurais convolucionais, Rahman et al. [4] utilizaram diversas métricas no estudo de categorização de frutas. Entre elas, a acurácia foi um indicador crucial, onde se destacaram os seguintes modelos: ResNet, atingindo 99,1% de acurácia, seguido por VGG-19 com 98,75%, Inception-V3 com 98,43% e ResNet-50 com 51,09%.

A partir dos resultados obtidos na categorização das frutas por meio desses modelos, Rahman et al. [4] constataram que três deles alcançaram uma precisão superior a 98%. Essa performance excepcional superou as expectativas dos pesquisadores, evidenciando de maneira significativa o potencial desses modelos para aplicações práticas. Esses resultados promissores sinalizam um avanço considerável na viabilidade do uso dessas redes neurais convolucionais, representando um substancial aprimoramento na eficiência dos processos associados.

Ainda no âmbito da categorização de frutas na agricultura, o estudo conduzido por Khatun et al. [5] amplia a aplicação das redes neurais, explorando a viabilidade da integração de técnicas de visão computacional e modelos de redes neurais no contexto específico da categorização de frutas no campo. Utilizando um banco de imagens composto por 1260 imagens representando sete classes distintas de frutas, os pesquisadores empregaram uma abordagem que combina processamentos de imagens baseados em conceitos de visão computacional com modelos de redes neurais como MobileNet, Inception v3 e ImageNet.

A segregação do banco de imagens em 85% para o treinamento e 15% para os testes, juntamente com 10 épocas de treinamento, resultou em uma impressionante taxa de acerto

de 98,74%. Essa notável precisão obtida na categorização das frutas demonstra o potencial exponencial da aplicação de redes neurais em sistemas de detecção automática no campo agrícola.

Diante dessa taxa de acerto significativa, Khatun et al. [5] concluem que a categorização por meio de redes neurais apresenta um extraordinário potencial para a implementação em sistemas de detecção automática de frutas no ambiente agrícola. Essa inovação representa não apenas uma ferramenta eficaz para a automatização de processos, mas também abre caminho para a substituição de tarefas anteriormente restritas à intervenção humana por métodos automatizados de alta precisão.

3. METODOLOGIA

Nesta seção, é apresentado o desenvolvimento do sistema de pesagem, do treinamento dos modelos de inteligência artificial e do desenvolvimento do sistema de integração necessários para a implementação do sistema proposto.

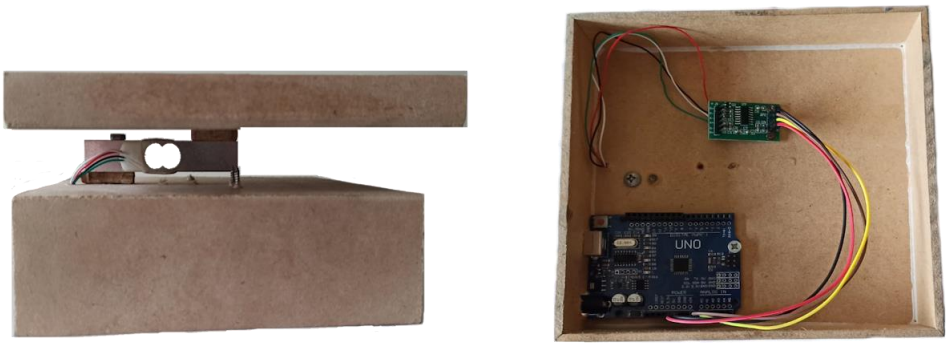
3.1. Desenvolvimento do sistema de pesagem.

Como parte constituinte dos objetivos, a construção de um sistema de pesagem requer o desenvolvimento de um protótipo físico de uma balança digital, que será capaz de se comunicar e integrar-se com o restante do sistema. Dessa forma, ao desenvolver o protótipo da balança, deu-se prioridade à acessibilidade, flexibilidade e à disponibilidade de componentes no mercado regional. Assumindo esse contexto, foram selecionados os seguintes elementos-chave para a construção do protótipo:

1. Célula de Carga
2. Módulo HX711
3. Arduino UNO

Para o primeiro elemento, optou-se por utilizar uma célula de carga de 5kg. Tal escolha se deu baseada no fato de que as células de carga de 5 kg apresentam uma ótima precisão em faixas de medição até esse limite. O segundo elemento, foi escolhido devido a sua excelente compatibilidade com as células de carga e o microcontrolador Arduino. Além disso, a facilidade de obtenção de informações sobre seu uso para o mesmo objetivo contribuiu bastante para tal escolha. Por fim, o terceiro elemento é escolhido devido sua ampla utilização em projetos semelhantes.

Desse modo, iniciou-se a construção da balança para o sistema de pesagem, utilizando uma caixa de MDF como base. Sobre essa base, foram fixados os principais elementos, incluindo a célula de carga, o módulo HX711 e o Arduino UNO, conforme é visto na Figura 1.



(a) (b)

Figura 1: (a) Vista lateral esquerda do protótipo. (b) Vista inferior do protótipo. Fonte: Autores

A realização das conexões entre os componentes é observada no esquema das conexões presentes na Figura 2.

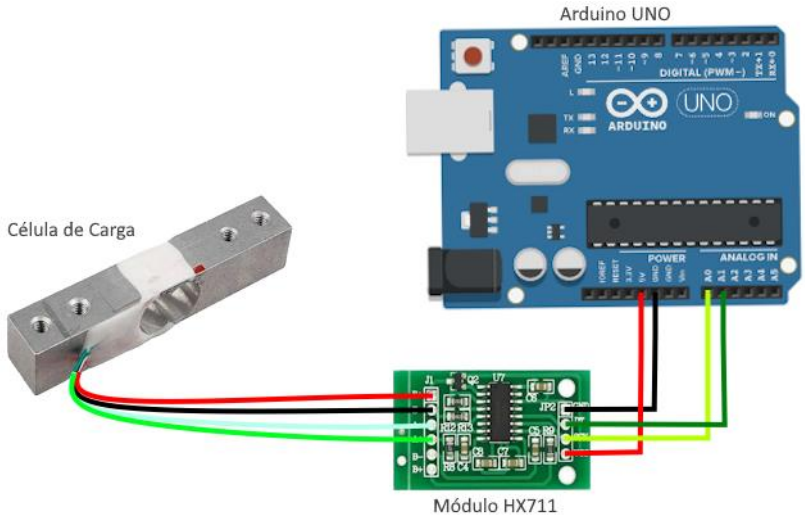


Figura 2: Conexão dos componentes da balança. fonte: Autores

A montagem da parte física do protótipo foi concluída após a conexão dos componentes. No entanto, a etapa de calibração da balança ainda precisava ser realizada, a qual foi conduzida por meio de software. Nesse contexto, utilizou-se a biblioteca "HX711" do Arduino para realizar a calibração empiricamente até se atingir o fator de calibração desejado.

Conforme ilustrado na Figura 3, o processo de calibração envolve a aplicação de um fator de escala ao sinal de saída do módulo HX711, que é lido pelo Arduino. Esse fator é ajustado pelo projetista até que a saída corresponda ao peso correto. Portanto, é fundamental aplicar um peso de referência previamente conhecido na balança para permitir um ajuste preciso.

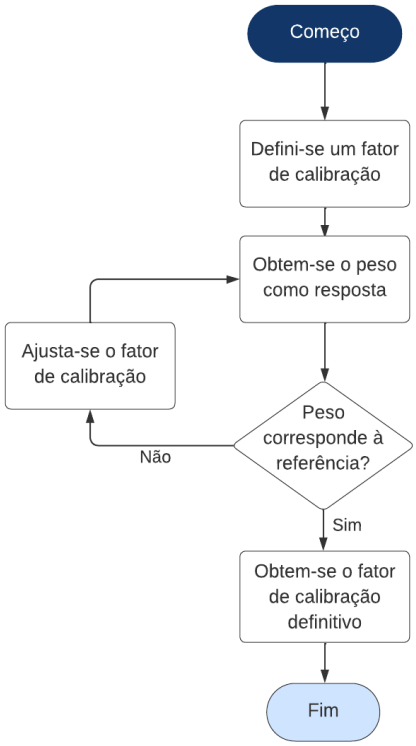


Figura 3: Fluxograma algoritmo de calibração. fonte: Autores

Para esse propósito, foram selecionadas duas moedas de 1 real datadas após 2002, totalizando 14 gramas, conforme determinado pelo banco central. Com a massa de referência posicionada na balança, o fator de calibração foi ajustado até que a saída coincidisse com o valor de referência.

3.2. Obtenção do Dataset.

Entendendo a necessidade de um *Dataset* adequado para o treinamento da inteligência artificial, foram levados em consideração 3 pontos: O plano de fundo das imagens, uma vez que um plano de fundo diferente no treinamento e na aplicação podem levar a resultados inconsistentes, dessa forma serão utilizados os mesmos planos de fundos na produção do *dataset* e no protótipo final, a posição da câmera, da mesma forma que o plano de fundo, um ângulo de câmera diferente do utilizado durante o treinamento pode levar a resultados indesejáveis, e o tamanho das imagens, que caso sejam muito grandes podem levar a necessidade de um poder computacional elevado durante o treinamento, além disso existe uma limitação devido ao hardware disponível, logo, é de extrema importância que o tamanho das imagens seja usado da forma mais efetiva.

Tendo isso em mente, foram escolhidos os seguintes produtos para a criação do dataset:

- Batata
- Berinjela
- Cenoura
- Chuchu

- Maçã
- Rabanete
- Tomate

São produtos que se correlacionam em formato, como o chuchu e a berinjela, ou se parecem com relação à cor, como a maçã, tomate e rabanete. Dessa forma, poderá ser observado se a rede neural será capaz de diferenciar de maneira consistente diferentes frutas com características semelhantes.

Foram feitas cerca de 4200 imagens de resolução 160x160px, esse foi o maior tamanho possível encontrado após vários testes, considerando o hardware disponível, onde 90% serão usadas para treinamento e 10% serão usadas para validação.

3.3. Aplicação de CNNs

O próximo passo após a obtenção do *dataset* é a seleção e treinamento das redes neurais a serem usadas, foram utilizadas as arquiteturas EfficientNetB0 e ResNet50, e a plataforma Google Collaboratory para treinamento, onde pode-se pegar uma placa de vídeo de graça por um limite de tempo e a usar para treinamento. A Figura 4 mostra a arquitetura da rede EfficientNetB0.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Figura 4. Arquitetura EfficientNetB0. Fonte: TAN, Mingxing; LE, Quoc. [7].

Para a compilação do modelo EfficientNetB0 foi utilizado o otimizador *Adam.* O treinamento foi feito com 15 épocas e um *batch size* de 32.

A Figura 5 mostra o gráfico de *accuracy* do treinamento comparado com a validação e a Figura 6 mostra o gráfico de *loss* do treinamento comparado com a validação.

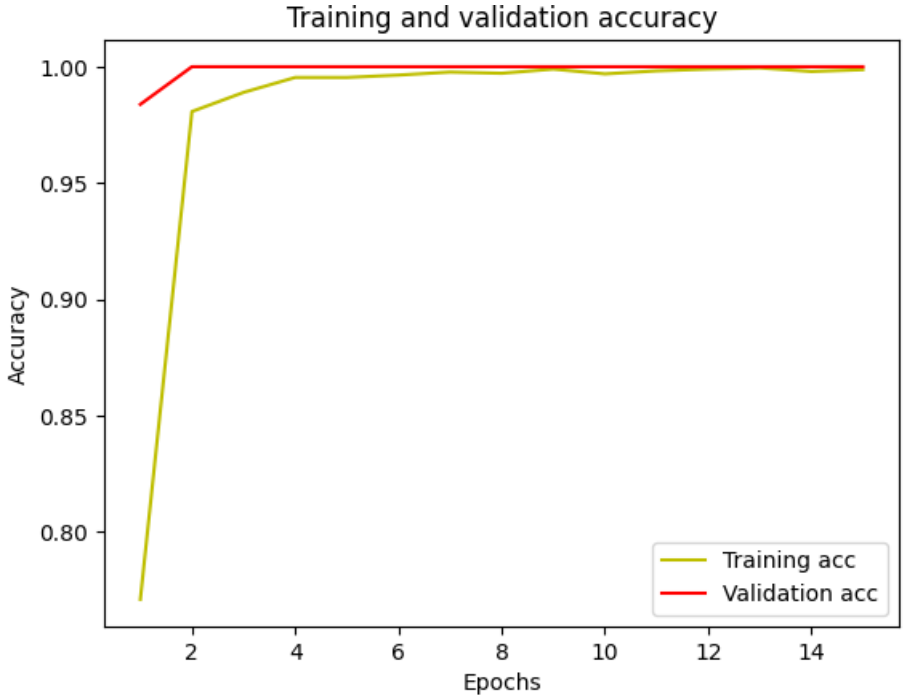


Figura 5. *Accuracy* obtida com a EfficientNetB0. Fonte: Autores.

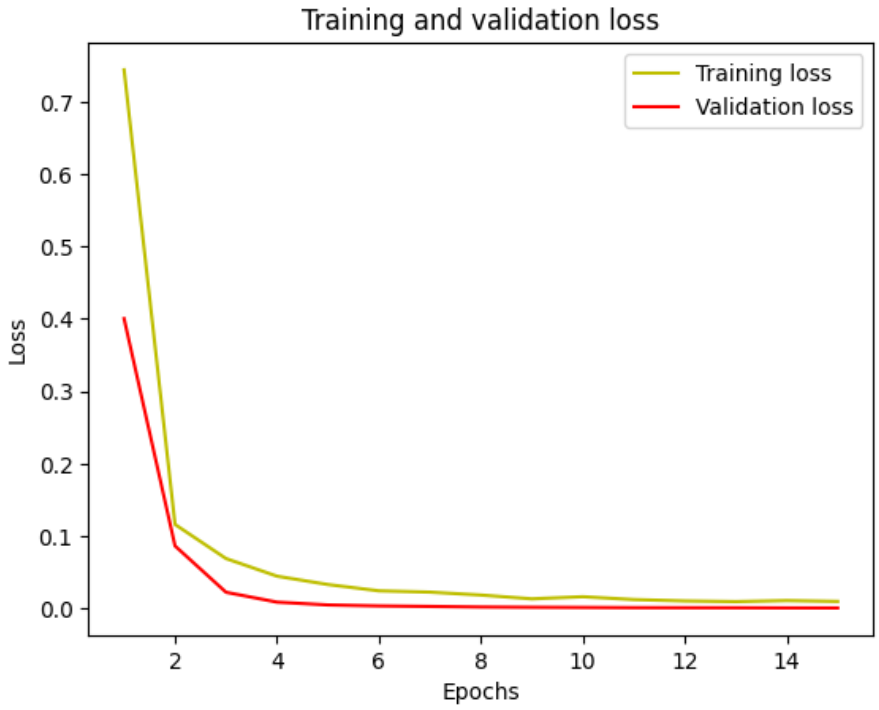


Figura 6. *loss* obtida com a EfficientNetB0. Fonte: Autores.

A Figura 7 mostra as arquiteturas dos modelos ResNet. Interessa-se principalmente pela arquitetura do modelo ResNet50.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figura 7. Arquitetura ResNet50. Fonte: Sarwinda et al. [6].

Para a compilação do modelo ResNet50 também foi utilizado o otimizador *Adam.* O treinamento foi feito com 100 épocas e um *batch size* de 64 utilizando o *Dataset* citado anteriormente.

A Figura 8 mostra o gráfico de *accuracy* do treinamento comparado com a validação e a Figura 9 mostra o gráfico de *loss* do treinamento comparado com a validação.

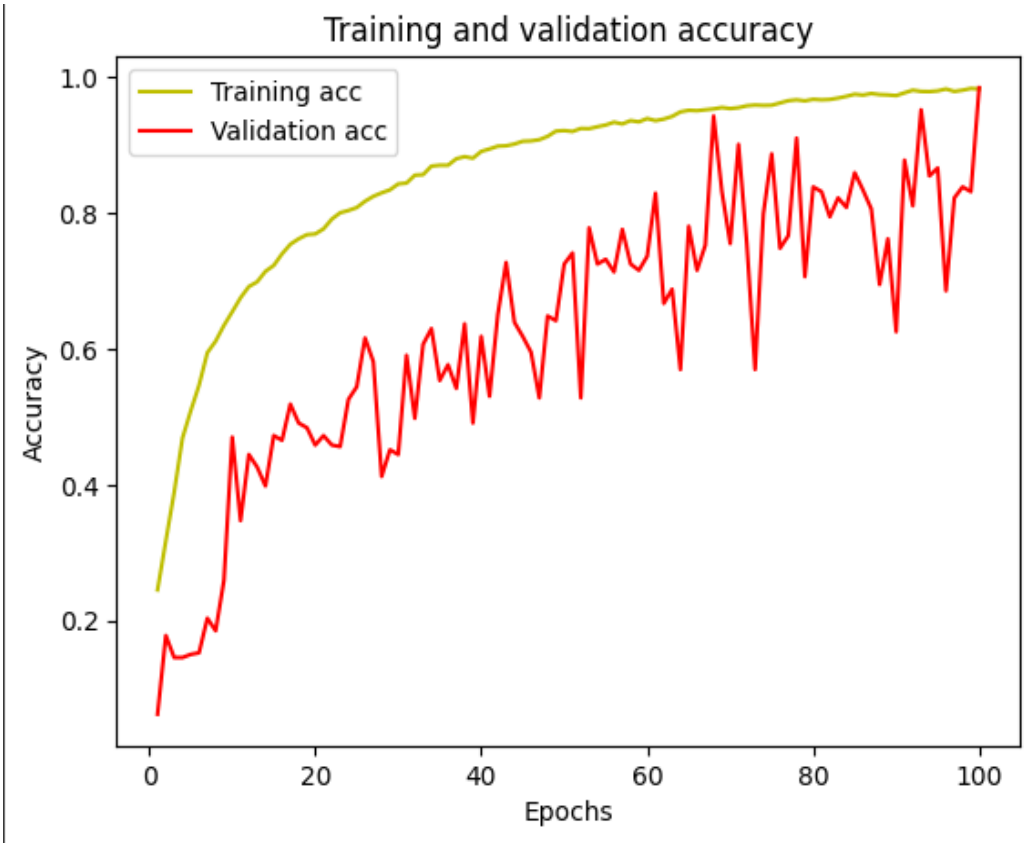


Figura 8. *Accuracy* obtida com a ResNet50. Fonte: Autores.

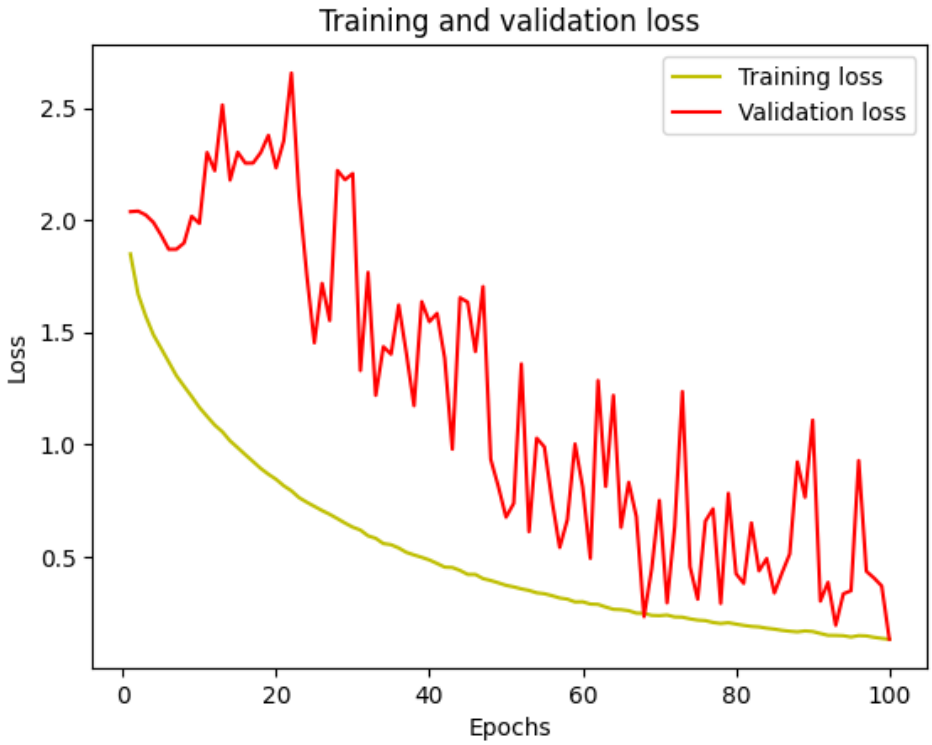


Figura 9. *Loss* obtida com a ResNet50. Fonte: Autores.

3.4. Desenvolvimento da integração dos sistemas

Os desenvolvimentos da inteligência artificial e do sistema de pesagem permitem a criação de um modelo de um sistema integrado de mercado. Nesta seção, a construção de tal sistema é descrita detalhadamente.

3.4.1. Seleção de ferramenta para criação da interface

A ferramenta escolhida é o Godot [8]. Apesar de ser uma *engine* de criação de jogos, Godot também facilita a criação de menus. Além disso, a ferramenta conta com sua própria linguagem de *script*, chamada GDScript [9], com sintaxe semelhante à de Python. Uma das versões do Godot permite a comunicação com C#, linguagem criada pela Microsoft com suporte multiplataforma e facilidade de acesso a recursos de baixo nível.

3.4.2. Desenho da interface

Godot possui um sistema de clica-e-arrasta que permite ao programador visualizar onde os componentes (botões, listas, textos, etc.) ficarão na tela durante a execução do programa.

A interface foi dividida em quatro menus, sendo eles:

A. Tela de carregamento: Uma tela simples para indicar que a IA e os componentes de banco de dados estão sendo carregados pelo programa. Neste estado, nenhuma ação por parte do usuário é considerada. Tal tela só aparecerá quando o sistema for reiniciado, não necessitando a reabertura após cada interação entre usuários.
B. Tela de início de sessão: Uma tela contendo um único botão de “iniciar sessão”. Aparece após a tela de carregamento e toda vez que o usuário anterior finalizar suas compras.

C. Lista de compras: Esta tela contém todos os itens selecionados pelo consumidor. Possui um indicador de preço total a pagar e três botões. O botão “Finalizar” finaliza a compra atual, gerando uma nota fiscal. O botão “Frutas” redireciona o usuário para a lista de produtos. O botão “Cancelar” remove o item selecionado da lista de compras, em caso de engano.
D. Lista de produtos: Esta tela contém um indicador do peso registrado na balança. Além disso, faz uma listagem dos produtos de hortifrúti disponíveis. A listagem possui dois níveis: genérico e específico. O nível genérico serve para produtos de apenas um tipo (por exemplo, existem supermercados que oferecem apenas um tipo de berinjela) e para categorias gerais de produtos (maçãs, por exemplo, costumam ser de diversos tipos, mas todas pertencem à categoria genérica maçã). O nível específico é acessado no segundo caso, ao escolher um produto que possui vários tipos diferentes.
   A tela de lista de produtos contém três botões. O botão “Confirmar” adiciona o produto selecionado à lista de compras e registra seu peso. O botão “Voltar” permite voltar um nível na lista ou retornar à lista de compras. Por padrão, esta tela mostra as três predições mais prováveis feitas pela IA. Se nenhum dos produtos apresentados for o correto, o usuário pode apertar o botão “Lista” para listar todos os produtos genéricos no sistema. Neste caso, o botão “Lista” se transforma em “IA”, que volta a apresentar as previsões da IA quando pressionado.

O fluxograma da Figura 10 ilustra o fluxo normal de execução do programa.

Figura 10. Fluxo normal de execução do programa de interface. Fonte: Autores.

A tela de produtos identificados pela rede neural convolucional é organizada em uma ordem específica, seguindo um padrão predefinido. Nessa disposição, o objeto que a rede neural previu com a maior probabilidade é exibido à esquerda, seguido pelo segundo objeto com a segunda maior probabilidade no centro, e o terceiro objeto, com a terceira maior probabilidade, é exibido à direita. Conforme é ilustrado na Figura 11.

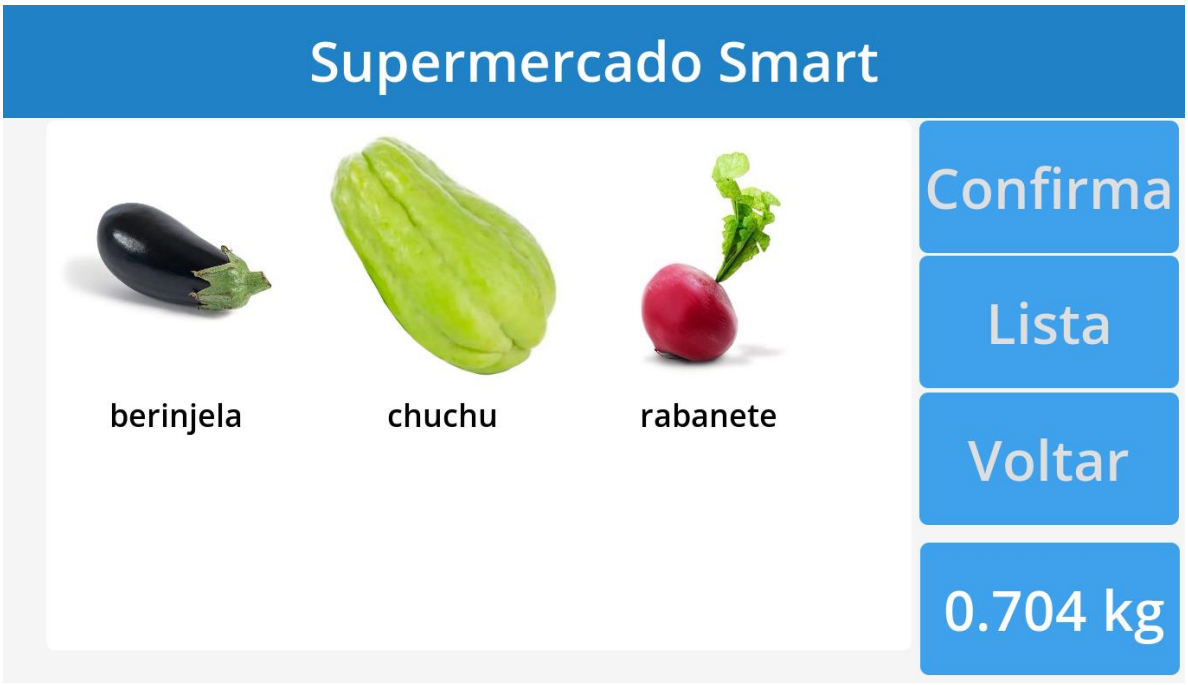


Figura 11. Tela de produtos com opções genéricas previstas pela IA. Fonte: Autores.

3.4.3. Banco de dados

Para desenvolver o banco de dados do projeto, optou-se por criar arquivos em MySQL [10] local. Os produtos foram divididos em duas tabelas, conforme mostrado no diagrama da Figura 12.

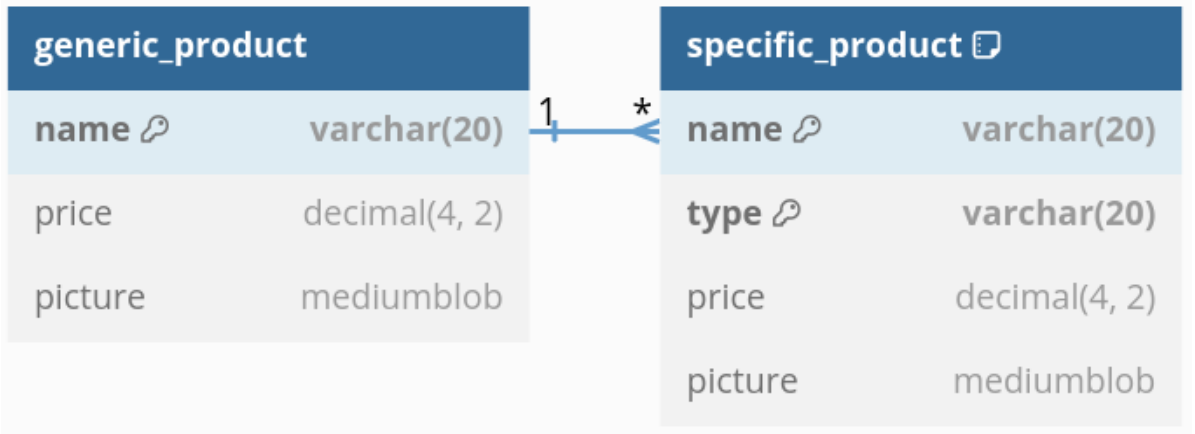


Figura 12. Modelo do banco de dados dos produtos. Fonte: Autores.

Ao iniciar o programa, todos os produtos registrados no banco de dados são carregados na máquina local. Como GDScript não possui suporte a *queries* de SQL, este procedimento é feito utilizando código em C#, que consiste em obter os dados das tabelas linha a linha, pôr as imagens dos produtos em um diretório e, junto com elas, um arquivo que contém os nomes e preços de todos os produtos.

3.4.4. Dados seriais

Os dados captados pelo Arduino Uno da balança são enviados para um canal serial que

pode ser identificado acessando a IDE do Arduino e checando as portas reconhecidas pelo programa.

3.4.5. Aplicação da IA

O código que captura imagens e as envia para avaliação da IA foi desenvolvido em Python. O programa carrega o modelo de IA desejado e, em seguida, aciona as câmeras, que passam a captar imagens ininterruptamente. A cada meio segundo, a IA é acionada para as duas câmeras, e uma média das previsões obtidas é feita.

3.4.6. Integração do sistema

A integração dos componentes do sistema se dá por meio de GDScript. Nos casos do banco de dados e da leitura da balança, os dados relevantes são obtidos com programas em C# cujas variáveis são transferidas para GDScript, permitindo sua inclusão na interface.

Por sua vez, a interpretação de imagens por parte da IA ocorre por meio de script em Python, que não é oficialmente suportado por Godot. Por isso, utiliza-se protocolo de comunicação próprio entre a IA e a interface, realizado por meio da criação e checagem de arquivos no diretório da IA. Desta forma, torna-se possível à interface saber quando a IA foi carregada no script, e à IA saber quando seu funcionamento pode ser encerrado. Também é por meio de arquivos que a interface obtém as informações mais recentes de reconhecimento produzidas pela IA.

4. RESULTADOS

Para obter resultados de comparação, foram feitas 10 imagens de cada classe, em seguida essas imagens foram apresentadas às redes treinadas e seus resultados foram anotados, criando assim uma matriz de confusão. Os resultados são avaliados considerando três critérios principais: velocidade de resposta, tamanho do modelo em megabytes e precisão na identificação dos elementos. A Tabela 1 mostra a matriz de confusão da EfficientNetB0 e a Tabela 2 mostra a matriz de confusão da ResNet50

Tabela 1 - Matriz de confusão para EfficientNet

EfficientNet		**Previsão**						
		Batata	**Beringela**	**Cenoura**	**Chuchu**	**Maçã**	**Rabanete**	**Tomate**
Verdadeiro	**Batata**	7	1	2				
	Beringela		10					
	Cenoura			10				
	Chuchu				10			
	Maçã					10		
	Rabanete						10	
	Tomate							10

FONTE: Autores

- O modelo obteve uma precisão de 95,71%.
- As previsões levaram um tempo médio de 0,074s.
- O tamanho do modelo é de 18 Mb.

Tabela 2 - Matriz de confusão para ResNet50

ResNet50		**Previsão**						
		Batata	**Beringela**	**Cenoura**	**Chuchu**	**Maçã**	**Rabanete**	**Tomate**
Verdadeiro	**Batata**			2	2	1		5
	Beringela			1	2	2	3	2
	Cenoura	1		1	1	2	2	3
	Chuchu	1		2	4	2		1
	Maçã		2			2	1	5
	Rabanete				1	2	6	1
	Tomate				5		2	3

FONTE: Autores

- O modelo obteve uma precisão de 22,86%.
- As previsões levaram um tempo médio de 0.089s.
- O tamanho do modelo é de 93,6 Mb.

O modelo EfficientNetB0 superou as expectativas, obtendo uma acurácia de 95,71%. No entanto, devido a sua arquitetura e forma como foi implementada, modelos como o ResNet-50 acabam não obtendo resultados tão bons, como no apresentado pela Tabela 2 e nos resultados no estudo realizado por Rahman et al [4] e Khatun et al [5], cujos resultados não foram tão precisos para essa arquitetura. Acredita-se que um *dataset* com imagens de maior resolução possa acarretar em um resultado melhor para modelos parecidos com o ResNet-50.

5. CONCLUSÃO

Após a realização deste estudo e com base nos resultados obtidos para ambos os modelos de rede neural convolucional implementados, chega-se à conclusão de que a implementação de forma correta acarreta em resultados extremamente positivos. É fundamental reconhecer que a escolha da arquitetura e as características do dataset desempenham um papel significativo no desempenho dos modelos de redes neurais convolucionais. Portanto, pesquisas futuras podem se beneficiar ao explorar diferentes abordagens de implementação e conjuntos de dados mais abrangentes para aprimorar a eficácia e precisão dessas redes.

6. REFERÊNCIAS

[1] RINALDI, José Gilberto Spasiani. A Importância da Rapidez de Atendimento nos Caixas de Supermercados: um estudo de caso utilizando um modelo analítico de filas com trocas. 2007.

[2] OPARA-NADI, Gregory E. *Electronic self-checkout system vs. cashier operated system: A performance based comparative analysis*. 2005. Tese de Doutorado. Capella University.

[3] DHANUSH, Guduru et al. A comprehensive review of machine vision systems and artificial intelligence algorithms for the detection and harvesting of agricultural produce. **Scientific African**, p. e01798, 2023.

[4] RAHMAN, Md Mahbubur et al. A deep CNN approach to detect and classify local fruits through a web interface. Smart Agricultural Technology, v. 5, p. 100321, 2023.

[5] KHATUN, Mehenag et al. Fruits classification using convolutional neural network. **GRD Journals-Global Research and Development Journal for Engineering**, v. 5, n. 8, 2020.

[6] Devvi Sarwinda, Radifa Hilya Paradisa, Alhadi Bustamam e Pinkie Anggia. "Deep Learning in Image Classification using Residual Network (ResNet) Variants for Detection of Colorectal Cancer". Em: *Procedia Computer Science* 179 (2021). DOI: 10.1016/j.procs.2021.01.025. URL: https://www.sciencedirect.com/science/article/pii/S1877050921000284.

[7] TAN, Mingxing; LE, Quoc. Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. PMLR, 2019. p. 6105-6114.

[8] Godot Engine - Free and open source 2D and 3D game engine. Godot Engine, 2023. Disponível em: https://godotengine.org/. Acesso em: 28 de nov. de 2023.

[9] GDScript reference - Godot Engine (stable) documentation in English. Godot Engine, 2023. Disponível em: https://docs.godotengine.org/en/stable/tutorials/scripting/gdscript/gdscript_basics.html. Acesso em: 28 de nov. de 2023.

[10] MySQL. MySQL, 2023. Disponível em: https://www.mysql.com/. Acesso em: 28 de nov. de 2023.