Vision Scan Insight: Um Assistente Inteligente Utilizando Redes Neurais Profundas para Usuários de Baixa Visão em Supermercados

Paulo S. Cabral[1], **Josenildo C. da Silva**[1], **João B. Diniz**[2], **Daniel L. Gomes Jr.**[1]

[1]Departamento de Computação – Instituto Federal do Maranhão (IFMA)
CEP 65030-005 – Monte Castelo – São Luís, MA – Brazil

[2]Departmento de Informática – Instituto Federal do Maranhão (IFMA)
CEP 650000 – Grajaú, MA – Brazil

cabral.ps@gmail.com, {jcsilva,joao,daniellima}@ifma.edu.br

Abstract. *This article proposes an intelligent tool to assist people with blindness or low vision during shopping visits in a supermarket. The solution utilizes deep neural networks (CNNs) to detect objects, integrated into an application that offers real-time auditory feedback. The system integrates three YOLO variants (v5, v8, and v9), each retrained using both full fine-tuning and restricted transfer learning on three product datasets (Food, $mAP_{50-95} = 0.683$; No-Fridge, $mAP_{50-95} = 0.697$; Groceries, $mAP_{50-95} = 0.916$).*

Resumo. *Este artigo propõe uma ferramenta inteligente para auxiliar pessoas com cegueira ou baixa visão durante as compras em um supermercado. A solução utiliza redes neurais profundas (CNNs) para detectar objetos, integrada a um aplicativo que oferece feedback auditivo em tempo real. O sistema integra três variantes de YOLO (v5, v8 e v9), retreinadas via fine-tuning completo e transfer learning restrito em três bases de dados de produtos (Food, mAP_{50-95}=0,683; No-Fridge, mAP_{50-95}=0,697; Groceries, mAP_{50-95}=0,916).*

1. Introdução

De acordo com o Conselho Brasileiro de Oftalmologia (CBO), existem cerca de 1,5 milhão de brasileiros com diagnóstico de cegueira [Umbelino and de Ávila 2023]. As pessoas com baixa visão geralmente precisam de assistência para atividades cotidianas [de Oliveira et al. 2016], por exemplo, durante as compras em um supermercado que requer a localização dos produtos, navegação e deslocamento entre as prateleiras, além de leitura de informações no rótulo sobre o tipo de produto, ingredientes, código de barras, preço, entre outras coisas. Como consequência, enfrentam dificuldades significativas para realizar tarefas simples de forma autônoma, o que pode impactar negativamente sua qualidade de vida e independência.

Portanto, existe uma demanda por tecnologias assistivas que possam ajudar as pessoas com baixa visão e cegueira a superarem esses desafios e a realizarem suas atividades diárias de forma mais independente e eficiente. As tecnologias assistivas podem incluir aplicativos móveis que identificam produtos por meio de leitura de códigos de barras,

dispositivos de realidade aumentada para auxiliar na navegação dentro do supermercado, sistemas de voz para fornecer informações sobre os produtos, entre outras soluções inovadoras [de Oliveira and Okimoto 2022, Pundlik et al. 2023].

Desta forma, o objetivo principal deste artigo é propor uma ferramenta inteligente para auxiliar pessoas com baixa visão durante uma visita a um supermercado. A abordagem aqui proposta utiliza redes neurais convolucionais profundas (do inglês, *convolutional neural network* - CNNs) para detecção do objeto embarcada em uma aplicação que retorna ao usuário em forma de áudio. As principais contribuições deste trabalho são: a) aplicação de CNNs para detecção de objetos embarcada em assistente inteligente para pessoas com baixa visão em supermercados; b) melhoria da acessibilidade em supermercados para pessoas com baixa visão.

Este artigo está organizado da seguinte forma. Na Seção 2 é apresentado um referencial teórico que contextualiza este trabalho. O método proposto é discutido em detalhes na Seção 3. Os resultados encontrados e a discussão sobre o resultado do trabalho são apresentados na Seção 4. Conclusão e perspectivas futuras são apresentadas na Seção 5.

2. Referencial Teórico e Trabalhos Relacionados

Nesta seção serão descritas as técnicas e conceitos necessários para o melhor entendimento do método proposto.

2.1. Redes Neurais Convolucionais

Redes neurais convolucionais são uma classe de modelos de aprendizagem de máquina que podem ser treinados para uma ampla variedade de tarefas, incluindo reconhecimento de imagem [Lecun et al. 1998, Krizhevsky et al. 2012], processamento de linguagem natural, previsão de séries temporais e sistemas de recomendação. Uma rede neural profunda é uma arquitetura avançada que consiste em múltiplas camadas, permitindo uma aprendizagem complexa de representações dos dados de entrada. Suas principais camadas são convolucionais e de *pooling*, essenciais para a extração eficiente de características e a redução da dimensionalidade dos dados, respectivamente [Goodfellow et al. 2016].

Por meio de operações de convolução, a camada convolucional extrai características relevantes dos dados de entrada, enquanto a camada de *pooling* reduz a resolução espacial, mantendo as características mais importantes [Schmidhuber 2015]. A abordagem de CNNs tem sido amplamente adotada devido à sua capacidade de aprender representações hierárquicas complexas dos dados, resultando em desempenho superior em uma variedade de tarefas de aprendizado de máquina.

2.2. Detecção de Objetos

Detecção de objetos tem como objetivo identificar a presença de uma determinada classe em uma imagem ou vídeo, incluindo sua localização em termos de coordenadas . Esta é uma tarefa essencial em várias aplicações de visão computacional, incluindo vigilância por vídeo, veículos autônomos, sistemas de assistência ao motorista, reconhecimento facial, entre outros [Szeliski 2022], [González and Woods 2008] .

Existem duas principais abordagens para detecção de objetos utilizando redes neurais profundas: estágio único e dois estágios. As abordagens em dois estágios primeiro

propõem regiões candidatas e depois realizam classificação e localização nestas regiões, por exemplo *Region-based Convolutional Neural Networks* (R-CNN) [Xie et al. 2021] e *Fast* R-CNN [Girshick 2015]. Por outro lado, as abordagens de estágio único tentam detectar objetos em uma única passagem pela imagem, sem a etapa de proposta de região, por exemplo *Single-shot Multibox Detector* (SSD) [Liu et al. 2016] e *You Only Look Once* (YOLO) [Hussain 2024, Terven et al. 2023].

2.3. Série de Redes YOLO

O YOLO é um *framework* de detecção de objetos de alto desempenho e código aberto. A *You Only Look Once* (YOLO) é capaz de detectar objetos em uma imagem em uma única passagem e possui uma saída mais simples que abordagens anteriores [Hussain 2024].

YOLOv1 [Redmon et al. 2016] foi proposta em 2016 e depois disso muitas variações foram propostas nos últimos anos. Foi fortemente influenciada pela GoogLeNet substituindo os módulos de *inception* por convoluções (1x1) seguidas de filtros convolutivos (3x3). Foi treinada inicialmente no conjunto de dados ImageNet e utilizava *Leaky ReLU* como função de ativação em todas as camadas, exceto a última. As versões YOLOv2 até YOLOv4 representam avanços sobre as limitações de versões anteriores, mas todas mantêm a arquitetura Darknet como plataforma base (*backbone*).

A partir da YOLOv5 a família YOLO abandonando o *framework* Darknet, passa a ser desenvolvida com Pytoch e é desenvolvida pela Ultralytics. Desde então, YOLO é mantido como projeto de código aberto [Jocher 2020]. As versões YOLOv6 e YOLOv8 representam desenvolvimentos incrementais. A versão YOLOv8 pode ser utilizada para várias tarefas além de detecção de objetos, por exemplo, segmentação e estimação de pose humana. YOLOv9 [Wang et al. 2024] marca um avanço significativo na tecnologia de detecção de objetos em tempo real, introduzindo técnicas inovadoras como o Programmable Gradient Information (PGI) e a Generalized Efficient Layer Aggregation Network (GELAN).

2.4. Aplicativos e Ferramentas Similares

Diversos produtos no mercado possuem funcionalidades similares à ferramenta proposta. Os aplicativos para reconhecimento de imagem incluem o AiPoly Vision [Lomas 2015], que identifica produtos a partir de imagens capturadas, operando offline, e o Seeing AI [Jaiman 2021], da Microsoft, que oferece descrição em tempo real de objetos, pessoas e textos. Para assistência colaborativa, o Be My Eyes [Saliba 2015] e o BeSpecular [Holton 2016] permitem que voluntários auxiliem usuários com deficiência visual, seja por chamadas de vídeo ou por meio da descrição de imagens enviadas. Já o TapTapSee identifica objetos por meio da câmera do dispositivo, utilizando a API da CloudSight para análise de imagem [Holton 2013].

Muitas das ferramentas mencionadas dependem de uma conexão de internet para funcionar corretamente, além de enfrentarem desafios de precisão na identificação de produtos em condições adversas, como iluminação variável ou obstruções parciais. A disponibilidade de voluntários para assistência colaborativa também pode ser uma limitação significativa em algumas dessas ferramentas.

3. Método Proposto

Para realizar a detecção de objetos, a abordagem proposta baseia-se em uma rede YOLO treinada para imagens de produtos de supermercado. As redes YOLO são treinadas em um conjunto de dados geral (COCO) com capacidade de reconhecer 80 classes, incluindo pessoa, ônibus, livro, e algumas frutas. Portanto, foi necessário retreinar a rede a partir dos pesos originais para que se ajustasse a um conjunto de dados com produtos específicos de supermercado.

Esta seção descreve detalhadamente o método proposto. Além disso, como um dos resultados foi a produção de um Produto Mínimo Viável (MVP), sua descrição será abordada na seção de resultados. Para tanto, o treinamento é realizado de modo offline e o modelo produzido é disponibilizado para que a ferramenta possa utilizá-lo na detecção. Para a fase de inferência, a rede é convertida para um formato adequado para uso em dispositivos móveis. Os detalhes da abordagem proposta são discutidos nas seções seguintes (Figura 1).

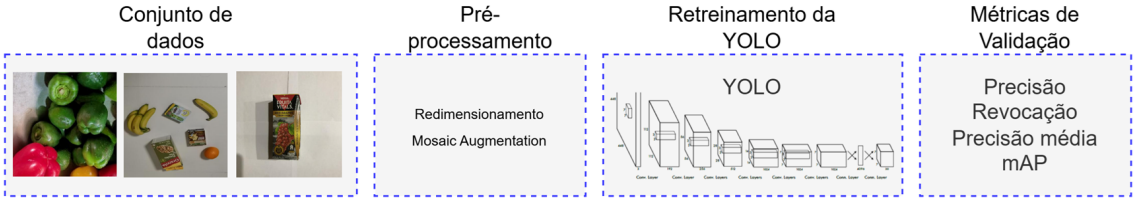

Figura 1: Pipeline de treinamento e conversão do modelo.

3.1. Conjuntos de dados

Neste trabalho, foram utilizadas várias bases de dados com imagens de produtos de supermercado, conforme descritas a seguir.

- Food: o conjunto de dados possui 2346 imagens (após aplicação de augmentations), com 142 classes, coloridas.
- No-Fridge: o conjunto de dados possui 3350 imagens (após aplicação de augmentations), 196 classes, coloridas.
- Grocery: o conjunto de dados do Kaggle possui 2040 imagens (após aplicação de augmentations), de 132 classes, coloridas.

A Figura 2 apresenta uma amostra das imagens em cada dataset.

(a) Food **(b) No-Fridge** **(c) Grocery**

Figura 2: Amostras dos conjunto de dados

3.2. Pré-processamento

Todas as imagens foram modificadas para apresentarem as mesmas dimensões 640x640 antes de serem apresentadas à rede. Esta transformação coloca as imagens no tamanho padrão de entradas para redes YOLO. Além disso, foi utilizado **mosaic augmentation**, incluído na implementação YOLO, que gera recortes de uma imagem e produz novas imagens com reorganização destes recortes. Este processo acontece durante o treino [Dadboud et al. 2021].

3.3. Retreinamento da YOLO

As redes da família YOLO são treinadas com alguns conjuntos de dados conhecidos como, por exemplo, o *Common Objects in Context (COCO)*, logo não foram treinadas para a tarefa específica de identificação de produtos de supermercado. O retreinamento foi realizado por ajuste fino e por transferência de conhecimento. No ajuste fino, a rede é retreinada sem congelamento de pesos, sendo ideal para conjuntos de dados com distribuição de classes distinta do conjunto original de pré-treinamento. Já na transferência de conhecimento, parte dos pesos da rede pré-treinada é congelada, evitando sua alteração durante o retreinamento, o que é recomendado quando os novos dados diferem significativamente da tarefa original da rede.

3.4. Conversão para inferência em dispositivos móveis

Após o treinamento em PyTorch, o modelo é exportado para o padrão ONNX (Open Neural Network Exchange) e então convertido para o formato NCNN [1]. O NCNN é um motor de inferência para redes neurais de vários formatos desenvolvido por Tencent em puro C++ e sem dependências externas. Durante esse processo, realiza-se quantização de pesos de 32-bit para 16-bit (FP16) e remoção de operadores não suportados, resultando em um binário de modelo até 50% menor, especialmente indicado para dispositivos com poucos recursos computacionais.

3.5. Métricas de Avaliação

Neste estudo, utilizamos as seguintes métricas para avaliar a qualidade do modelo treinado:

$$\text{Precisão} = \frac{VP}{VP + FP} \tag{1}$$

$$\text{Revocação} = \frac{VP}{VP + FN} \tag{2}$$

$$\text{AP}_n = \sum_n (R_n - R_{n-1}) P_n \tag{3}$$

$$\text{mAP}_n = \frac{1}{n} \sum_{k=1}^{n} \text{AP}_k \tag{4}$$

onde VP são os verdadeiros positivos e FN são os falsos negativos. AP_n, Eq. (3), é a precisão média (*Average Precision*) para o limite n de revocação e precisão, R_n e P_n,

[1] `https://github.com/Tencent/ncnn`

respectivamente. O mAP$_n$ (*mean Average Precision at n*), Eq. (4), é a média da precisão para diferentes pontos de corte de confiança. Por exemplo, mAP$_{50}$ é o mAP para o limite de $0,50$ de revocação e precisão. Já o mAP$_{50-95}$ é a média da precisão média em uma variedade de limiares de interseção sobre união (IoU), variando de $0,50$ a $0,95$.

4. Resultados e Discussão

Nesta seção serão discutidos os experimentos realizados para testar as diferentes configurações de redes e opções de treinamento.

4.1. Configuração dos Experimentos

Foram avaliadas duas formas de retreinamento da rede: (a) retreinamento por ajuste fino, quando não há congelamento de nenhuma camada na rede pré-treinada; (b) retreinamento por transferência de conhecimento, onde foram congeladas 10 camadas. Estas camadas são conhecidas como *backbone* na arquitetura YOLO.

Em todos os experimentos foi utilizada a plataforma Google Colab, com 1 GPU Tesla T4, 12Gb de RAM, e 80Gb de disco. Como hiperparâmetros foram utilizadas 20 épocas, tamanho de batch 32, otimizador Adam com taxa de aprendizagem $lr = 0.0001$ e $momentum = 0.937$. Todos os experimentos foram realizados com dados de treino, validação e testes em separado, sem repetição de imagens entre os mesmos. Em todos os experimentos, utilizou-se a versão *small* dos modelos disponibilizados pela Ultralytics.

4.2. Desempenho com Ajuste Fino

Foram utilizados os modelos YOLOv5, YOLOv8 e YOLOv9 para experimentos de ajuste fino sem congelamento, treinados com os três conjuntos de dados distintos (Seção 3). Na Tabela 1 são mostrados os resultados produzidos.

Tabela 1: Resultados de várias redes YOLO com ajuste fino com 20 épocas

Conjunto de dados	Modelo	Precision	Recall	mAP50	mAP50-95
Food	YOLOv5su	0.719	**0.823**	0.827	0.645
	YOLOv8s	0.798	0.767	**0.830**	**0.667**
	YOLOv9s	**0.801**	0.753	**0.830**	0.646
No-Fridge	YOLOv5su	0.926	0.948	0.969	0.668
	YOLOv8s	0.926	**0.959**	0.974	**0.689**
	YOLOv9s	**0.941**	**0.959**	**0.981**	0.670
Groceries	YOLOv5su	**0.997**	0.998	0.994	0.910
	YOLOv8s	**0.997**	**1.000**	**0.995**	0.914
	YOLOv9s	0.996	0.999	**0.995**	**0.916**

De acordo com os resultados, nota-se que não há um modelo que atinja melhores métricas para todos os conjuntos de dados. Entretanto, todos os modelos atingem valores muito próximos em todas as métricas. Isto permite que a escolha do modelo final para o desenvolvimento da ferramenta possa basear-se em aspectos operacionais, tais como tempo de resposta e tamanho final do modelo.

O conjunto de dados Groceries produziu os melhores resultados em relação aos outros conjuntos de dados com todos os modelos. Isto pode ser explicado pelo fato de

que há menos classes neste conjunto de dados ou que há menos desbalanceamento entre as classes.

4.3. Desempenho com *Transfer Learning*

Foram utilizados os modelos YOLOv5, YOLOv8, e YOLOv9 para experimentos de *tranfer learning* com congelamento de 10 camadas relativas ao *back bone* da arquitetura YOLO. Os modelos foram treinados com três conjuntos de dados distintos (Seção 3). Na Tabela 2 são mostrados os resultados produzidos.

Tabela 2: Resultados de várias redes YOLO com *transfer learning* com 20 épocas

Conjuntos de dados	Modelo	Precision	Recall	mAP50	mAP50-95
Food	YOLOv5su	**0.876**	0.736	0.844	0.643
	YOLOv8s	0.826	**0.759**	**0.855**	**0.683**
	YOLOv9s	0.875	0.734	0.836	0.665
No-Fridge	YOLOv5su	0.935	0.959	0.972	0.670
	YOLOv8s	**0.952**	0.953	0.976	**0.697**
	YOLOv9s	0.942	**0.969**	**0.980**	0.676
Groceries	YOLOv5su	**0.998**	**0.999**	**0.995**	0.910
	YOLOv8s	0.997	**0.999**	**0.995**	0.917
	YOLOv9s	0.997	**0.999**	**0.995**	**0.924**

Quando comparado com os resultados alcançados com ajuste fino, *tranfer learning* com 10 camadas congeladas conseguiu melhorar resultados dos modelos na maioria das situações (Tabela 1 e Tabela 2).

4.4. Aplicativo Android

Um aplicativo móvel foi desenvolvido como MVP para demonstrar a utilização da ferramenta por um usuário final. Nesta etapa, o foco maior foi na definição da arquitetura do aplicativo e a distribuição do modelo (*deployment*). O aplicativo foi desenvolvido no Android Studio utilizando a biblioteca NCNN, que oferece um desempenho robusto e baixo consumo de recursos, ideal para plataformas móveis.

O aplicativo permite que o usuário aponte a câmera do celular para uma prateleira do supermercado o mesmo realiza a detecção dos produtos presentes na imagem. A interface foi projetada com acessibilidade em mente e, por isso, os botões apresentam esquema de cores com alto contraste e fontes com tamanho grande. A interface do aplicativo é mostrada na Figura 3.

4.5. Discussão

Os resultados dos experimentos mostram um grande equilíbrio entre os modelos. Em geral, os modelos YOLOv8 e YOLOv9 apresentaram os melhores resultados para conjuntos de dados específicos, mas não há diferenças significativas. O modelo YOLOv5su teve pior desempenho quando comparado com os YOLOv8s e YOLOv9s em quase todos os experimentos. O YOLOv9s apresentou superioridade em várias métricas, tanto com ajuste fino quanto com transferência de conhecimento. A superioridade no desempenho da v9 pode ser consequência de diferenças na sua arquitetura interna em relação à v8.

Figura 3: Interface do aplicativo

Como a base de dados `Food` atingiu-se um mAP50 de 0.885 e mAP50-95 de 0.683 usando ajuste fino com 20 épocas. No contexto da ferramenta assistiva, representa um estágio inicial promissor, embora ainda seja necessária uma melhoria significativa no desempenho do modelo. Para tanto, é necessário incluir mais classes e mais exemplos de treinamento de classes pouco representadas.

A quantidade limitada de exemplos para determinadas classes em alguns conjuntos de dados contribui significativamente para o desempenho insatisfatório do modelo em relação a estas classes. Uma das classes sub-representadas é o produto 'Nutella', que aparece apenas uma vez no conjunto de dados *Food*.

Em termos práticos, os níveis de precisão e revocação alcançados indicam que a ferramenta pode identificar corretamente a maior parte dos produtos em condições controladas. Entretanto, em ambientes não controlados, o desempenho atual pode resultar em erros de detecção ou baixa confiança, o que pode afetar significativamente a experiência do usuário final.

Ainda, há vários desafios para a aplicação em cenários reais: variações de iluminação dentro do supermercado, produtos parcialmente ocultos nas prateleiras, embalagens visualmente semelhantes entre diferentes marcas e limitações de processamento em dispositivos móveis são os principais obstáculos a serem superados. A cobertura atual de classes nas bases de dados é ainda restrita, o que dificulta a generalização do sistema para redes de supermercados com catálogos extensos.

5. Conclusão e Trabalhos Futuros

Neste artigo foi apresentado o *Vision Scan Insight*, uma ferramenta inteligente para auxiliar pessoas com baixa visão na tarefa de compras em um supermercado. A ferramenta utiliza YOLO retreinada para a detecção de produtos em supermercados. Embora a arquitetura YOLO seja amplamente consolidada, nossa abordagem introduz várias inovações metodológicas específicas para o domínio assistivo em supermercados. Realizamos ajuste fino integral e experimentos de transferência de aprendizado em três bases de dados distintas (Food, No-Fridge e Groceries), assegurando cobertura de classes variadas com uso de mosaic augmentation para mitigar desequilíbrios e melhorar a robustez a variações de embalagem.

Além disso, desenvolvemos um pipeline de inferência móvel on-device via NCNN em Android, garantindo detecção em tempo real sem dependência de conexão à internet. Incorporamos feedback auditivo e tátil na interface para maximizar a acessibilidade, o que ultrapassa a simples aplicação de YOLO em benchmarks convencionais. A implementação desta tecnologia representa um avanço significativo na acessibilidade, proporcionando uma experiência de compra mais independente e eficiente para pessoas com baixa visão.

Como trabalhos futuros, propõe-se diminuir o tempo de resposta de detecção no aplicativo móvel, o que pode ser alcançado através de otimizações do modelo e processo de conversão de modelo. Além disso, planeja-se aumentar o número de produtos reconhecidos pelo sistema, ampliando sua base de dados e incorporando técnicas de aprendizado contínuo para que a ferramenta possa aprender novos produtos de forma dinâmica.

Agradecimento

Agradecemos ao apoio do Ministério da Ciência, Tecnologia e Inovação (MCTI) por meio dos recursos da Lei nº 8.248, de 23 de outubro de 1991, no âmbito do PPI Softex, coordenado pela Associação para Promoção da Excelência do Software Brasileiro (Softex), e publicado como Residência TIC09 (processo 01245.005714/2022-18), realizado pelo Instituto Federal do Maranhão (IFMA). Reconhecemos o uso de LLM para correção gramatical, ortográfica e também para tradução de termos específicos.

Referências

Dadboud, F., Patel, V., Mehta, V., Bolic, M., and Mantegh, I. (2021). Single-stage uav detection and classification with yolov5: Mosaic data augmentation and panet. In *17th Intl. Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–8. IEEE.

de Oliveira, R. D. and Okimoto, M. L. L. R. (2022). Tecnologias assistivas relacionadas à moda para pessoas com deficiência visual: uma revisão sistemática. *dObra[s] – revista da Associação Brasileira de Estudos de Pesquisas em Moda*, 2022:183–205. https://doi.org/10.26563/dobras.i35.1459.

de Oliveira, S. T., Bozo, J. V., and Okimoto, M. L. L. R. (2016). Assistive technology for people with low vision: Equipment for accessibility of visual information. In *Advances in Ergonomics in Design*, pages 701–710. Springer.

Girshick, R. (2015). Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448. https://doi.org/10.1109/ICCV.2015.169.

González, R. C. and Woods, R. E. (2008). *Digital image processing, 3rd Edition*. Pearson Education. https://www.worldcat.org/oclc/241057034.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Holton, B. (2013). A review of the taptapsee, camfind, and talking goggles object identification apps for the iphone. *AFB Access World*, Jul. 2013.

Holton, B. (2016). Bespecular: A new remote assistant service. *AFB Access World*, Jul. 2016.

Hussain, M. (2024). Yolov1 to v8: Unveiling each variant–a comprehensive review of yolo. *IEEE Access*, 12:42816–42833. https://doi.org/10.1109/ACCESS.2024.3378568.

Jaiman, A. (2021). Seeing ai: An app for visually impaired people that narrates the world around you. *Parliamentarian*, 102(4):380–381. ISSN 0031-2282.

Jocher, G. (2020). Ultralytics YOLOv5. https://github.com/ultralytics/yolov5.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Adv. in neural information processing systems*, 25.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324. https://doi.org/10.1109/5.726791.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, pages 21–37. Springer.

Lomas, N. (2015). Aipoly puts machine vision in the hands of the visually impaired. *TechCrunch*, Aug. 17, 2015.

Pundlik, S., Shivshanker, P., and Luo, G. (2023). Impact of apps as assistive devices for visually impaired persons. *Annual Review of Vision Science*, 9:111–130. https://doi.org/10.1146/annurev-vision-111022-123837.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788. https://doi.org/10.1109/CVPR.2016.91.

Saliba, E. (2015). Be My Eyes app let's vision-impaired people crowdsource sight. *NBC Today*, Jan. 26, 2015.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117. https://doi.org/10.1016/j.neunet.2014.09.003.

Szeliski, R. (2022). *Computer vision: algorithms and applications*. Springer Nature.

Terven, J., Córdova-Esparza, D.-M., and Romero-González, J.-A. (2023). A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. *Machine Learning and Knowledge Extraction*, 5(4):1680–1716. https://doi.org/10.3390/make5040083.

Umbelino, C. C. and de Ávila, M. P. (2023). *As Condições de Saúde Ocular no Brasil 2023*. Conselho Brasileiro de Oftalmologia.

Wang, C.-Y., Yeh, I.-H., and Liao, H.-Y. M. (2024). Yolov9: Learning what you want to learn using programmable gradient information. *arXiv preprint arXiv:2402.13616*.

Xie, X., Cheng, G., Wang, J., Yao, X., and Han, J. (2021). Oriented r-cnn for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3520–3529.