DETECÇÃO DE QUEDAS BASEADA EM VISÃO COMPUTACIONAL: COMPARAÇÃO ENTRE YOLO26N E RT-DETR-L

Victor Vinicius Figueiredo Silva[1] - victorsilva2004@proton.me
Kelton Martins Dias[1] **-** keltonmartinsd@gmail.com
Felipe Augusto Oliveira Mota[1] - felipe.mota@ifnmg.edu.br
[1]Instituto Federal do Norte de Minas Gerais, IFNMG – Januária, MG, Brasil

Resumo. *O envelhecimento populacional tem tornado as quedas em idosos uma preocupação crescente na área da saúde, uma vez que afetam a mortalidade, a morbidade e a capacidade funcional, sendo a detecção baseada em visão computacional uma alternativa não invasiva e de baixo custo para mitigar esse problema. Este trabalho tem como objetivo comparar os modelos de visão computacional YOLO26n e RT-DETR-L aplicados à detecção de quedas, avaliando seus resultados de precisão e de complexidade computacional, a fim de identificar qual modelo possui a melhor relação entre eficiência de detecção e viabilidade de implantação em dispositivos de baixo poder computacional. Para isso, foi reunido e rotulado um dataset a partir de vídeos de quedas e atividades da vida diária, utilizado no treinamento de ambos os modelos com a biblioteca Ultralytics. Os resultados mostram que os dois modelos apresentaram desempenho de detecção muito próximo, com diferença máxima de 0,60% na precisão geral. Entretanto, o YOLO26n se mostrou mais adequado ao objetivo do trabalho, por possuir uma quantidade consideravelmente menor de parâmetros, o que indica maior facilidade de implantação em dispositivos com recursos limitados.*

Palavras-chave: *detecção de quedas, RTDETR, visão computacional, YOLO.*

1. INTRODUÇÃO

O declínio na taxa de natalidade e aumento da expectativa de vida, e consequentemente o envelhecimento populacional tem se tornado uma preocupação crescente. Segundo a Organização Mundial da Saúde (WHO, 2025), o número de pessoas com 60 anos ou mais, que foi de 1 bilhão em 2020, deverá atingir 2.1 bilhões por volta de 2050. De acordo com Ungar *et al*. (2013), as quedas são eventos frequentes entre os idosos e afetam a mortalidade, a morbidade, a perda da capacidade funcional e a institucionalização, além disso, 40% a 60% das quedas ocorrem sem testemunhas.

As quedas são definidas como eventos acidentais nos quais uma pessoa cai quando perde o seu centro de gravidade e não realiza nenhum esforço para recuperar o equilíbrio, ou quando esse esforço é ineficaz. Ungar *et al*. (2013) explica que a causa pode envolver uma crise convulsiva, um acidente vascular cerebral (AVC), uma perda de consciência ou forças externas inevitáveis . Ren e Peng (2019) afirmam que a investigação de sistemas de detecção

e/ou prevenção de quedas constitui um dos principais focos de pesquisa, tendo sido amplamente explorada pelos pesquisadores nos últimos anos.

O trabalho de Ren e Peng (2019) identificou três principais categorias de abordagens na detecção de quedas, sendo elas as baseadas em sensores vestíveis (como acelerômetros), baseadas em sensores de ambiente (como pressão, infravermelho e som) e as baseadas em visão computacional. Os vestíveis, apesar de possuírem baixo custo, podem sofrer com o problema do idoso se esquecer ou se negar a utilizá-lo. Já os baseados em sensores de ambiente tem como fraqueza a sua área de cobertura e grau de interferência de outros fatores. Portanto, a detecção de quedas baseada em visão computacional demonstra ser uma opção viável, possuindo boa usabilidade e sendo não invasiva (pois não armazena nenhuma imagem detectada). Para que essa solução seja aplicável em cenários reais, é necessário que a detecção ocorra em tempo real, possibilitando resposta imediata diante de uma queda.

Nesse sentido, trabalhos recentes têm modificado a arquitetura de modelos YOLO especificamente para a detecção de quedas, como o de Huang *et al.* (2025), que propôs o SDES YOLO, uma modificação do YOLOv8. Outro exemplo é o trabalho de D. Zhao *et al.* (2024), que propôs o YOLOv7-fall, uma modificação do YOLOv7-tiny. Mais recentemente, o YOLO26, assim como detalhado por Chakrabarty (2026), tornou-se o modelo mais recente da família YOLO, sendo lançado em janeiro de 2026 pela Ultralytics, adotando pela primeira vez uma arquitetura *end-to-end* e livre de NMS, removendo a *Distribution Focal Loss* (DFL) e introduzindo o otimizador híbrido MuSGD, o que o torna mais leve e eficiente para dispositivos de baixo poder computacional.

Além da família YOLO, outra arquitetura relevante para detecção em tempo real é o RT-DETR (*Real Time Detector Transformer*), proposto por Y. Zhao *et al.* (2024), foi o primeiro detector de objetos baseado inteiramente em transformers a alcançar desempenho em tempo real, eliminando a necessidade de pós-processamento por *Non-Maximum Suppression* (NMS) através de um encoder híbrido.

Em paralelo ao desenvolvimento de arquiteturas mais eficientes, a *Edge AI* surge como uma uma tendência crescente para implementação de modelos de visão computacional local. Segundo Mendez *et al*. (2022), a *Edge AI* representa a confluência entre a computação de borda e o aprendizado de máquina, permitindo que o processamento dos dados ocorra próximo de sua origem, reduzindo a latência e riscos à privacidade associados ao envio de informações para processamento em nuvem. Haq e Verma (2026) afirmam que é especialmente relevante para aplicações relacionadas à saúde, como o monitoramento remoto de pacientes, pois possibilita a resposta em tempo real mesmo em cenários com a conectividade limitada, além de reduzir os custos operacionais. Com isso, a adoção de modelos leves e otimizados mostra-se essencial para viabilizar sua implantação em dispositivos de borda, viabilizando sistemas não invasivos, de baixo custo e capazes de operar sem depender de servidores externos. O presente trabalho irá utilizar as versões mais leves de cada modelo disponíveis na biblioteca Ultralytics, sendo o YOLO26n (YOLO26 nano) e RT-DETR-L (RT-DETR Large).

Considerando os pontos destacados, este artigo parte da hipótese que modelos de visão computacional leves, com treinamento, são capazes de apresentar bons resultados na detecção de quedas, mesmo com variação de ambientes e iluminação. Esta pesquisa inicial tem como objetivo comparar os resultados de precisão e de complexidade computacional dos modelos YOLO26n e RT-DETR-L a fim de identificar qual modelo possui a melhor relação entre eficiência de detecção e viabilidade de implantação em dispositivos de baixo poder computacional, servindo de apoio para etapas futuras do projeto.

2. METODOLOGIA

Foi realizada uma revisão da literatura, utilizando as bases IEEE *Xplore*, ACM *Digital library,* SBC-*OpenLib* e *ScienceDirect*. A revisão trouxe embasamento sobre a problemática da queda em idosos e a necessidade de tecnologias não invasivas para mitigar esse problema. Também demonstrou trabalhos que já implementaram soluções nesse contexto, e a partir desses trabalhos foi possível perceber a falta de testes com modelos mais atualizados e a falta de modelos voltados para hardware com baixo poder de processamento, para aplicar num conceito *Edge AI*.

O *dataset* utilizado neste trabalho foi reunido a partir de vídeos de quedas e ADL (*Activities of Daily Living* - Atividades da Vida Diária) retirados de *datasets* já utilizados em outros trabalhos, como o *Le2i* (Charfi *et al*., 2013), *Video-Based Fall Detection Dataset with 2017 Activities from 29 Subjects* (Ursul, 2025) e GMDCSA24(Alam, 2024), além de imagens de *UR Fall* (Roboflow, 2024). Dos *datasets*, foram selecionados vídeos com cenas consideradas diversas para compor o *dataset* final, os vídeos foram então separadas em em *frames* no formato JPEG para melhor tratamento dos dados utilizando a ferramenta *open source* FFmpeg com o parâmetro -q:v 2, que especifica qualidade máxima para os frames retirados do vídeo. O *dataset UR Fall*, por se tratar de um *dataset* de imagens, não passou pelo processo anterior, apenas teve a conversão das imagens de PNG para JPEG, também utilizando a ferramenta FFmpeg.

Após a seleção, foi realizada a classificação das imagens no CVAT *Server*, uma ferramenta para anotação de imagens e vídeos que é amplamente utilizada para geração de *datasets* para visão computacional. Em todas as imagens que possuem uma pessoa, foi desenhada uma *bouding box* (BB) em volta da pessoa presente na imagem, e logo após, essa BB é rotulada como em pé, sentada, em queda ou caída. Com a rotulação das imagens finalizadas, o *dataset* foi exportado e convertido para o formato YOLO. Na etapa seguinte, foram excluídas imagens repetidas para evitar redundância no *dataset*, como em momentos em que uma pessoa cai e fica parada, por exemplo. Ao final foram obtidas 14499 imagens em pé, 3425 imagens sentado, 1028 em queda e 1142 caídas. Para evitar o desbalanceamento do *dataset*, as classes foram reduzidas para mil imagens, cada, utilizando amostragem aleatória simples para distribuir as imagens de forma proporcional. Por fim, o *dataset* foi dividido em treinamento, validação e teste com a proporção 70%, 15% e 15% respectivamente, sendo um padrão mais conservador recomendado pela própria Ultralytics.

O treinamento ocorreu em um notebook do Kaggle, por oferecer um ambiente para armazenamento do *dataset* e execução do treinamento em um ambiente de nuvem e com hardware poderoso, utilizando a aceleração de duas GPUs T4. A biblioteca utilizada para importar e treinar os modelos YOLO26n e RT-DETR-L foi a biblioteca da Ultralytics, por possibilitar a implementação e treinamento de modelos de visão computacional de forma fácil e eficiente, além de possuir ampla documentação *online*. Os parâmetros utilizados no treinamento dos dois modelos foram:

- *epochs:* 100
- imgsz: 640
- *batch:* 32
- *device:* [0,1]
- *optimizer:* AdamW
- lr0: 0.001
- lrf: 0.01
- *dropout:* 0.2

As métricas utilizadas para avaliar o desempenho dos modelos foram: precisão(P), *recall*(R) e mAP como métricas de eficiência de detecção; o GFLOPs e número de parâmetros

do modelo como métricas de complexidade computacional; além dos gráficos de perda(loss), que demonstram a queda do erro dos modelos ao longo do treinamento.

3. RESULTADOS

Os resultados dos treinamentos dos modelos YOLO26n e RT-DETR-L são demonstrados na Tabela 1 e Tabela 2.

Tabela 1. Resultados de treinamento RT-DETR-L: métricas de eficiência de detecção.

	em pé	sentado	em queda	caído	média geral
precisão	98.50%	97.70%	93.60%	96.10%	96.50%
recall	94.70%	100.00%	96.00%	98.20%	97.20%
mAP50	98.10%	99.10%	96.80%	97.60%	97.90%
mAP50-95	84.00%	95.90%	91.00%	93.50%	91.10%

Tabela 2. Resultados de treinamento YOLO26n: métricas de eficiência de detecção.

	em pé	sentado	em queda	caído	média geral
precisão	97.80%	99.20%	94.70%	96.80%	97.10%
recall	95.30%	98.00%	97.30%	98.70%	97.30%
mAP50	96.90%	99.40%	97.70%	99.30%	98.30%
mAP50-95	85.00%	94.10%	91.20%	94.30%	91.20%

Ademais, nas métricas de complexidade computacional, o YOLO26n teve 5.2 GFLOPs e 2,375,616 parâmetros e o RT-DETR-L 4 GFLOPs e 31,991,960 parâmetros. As métricas de treinamento e validação ao longo das épocas são representadas na Figura 1 e Figura 2.

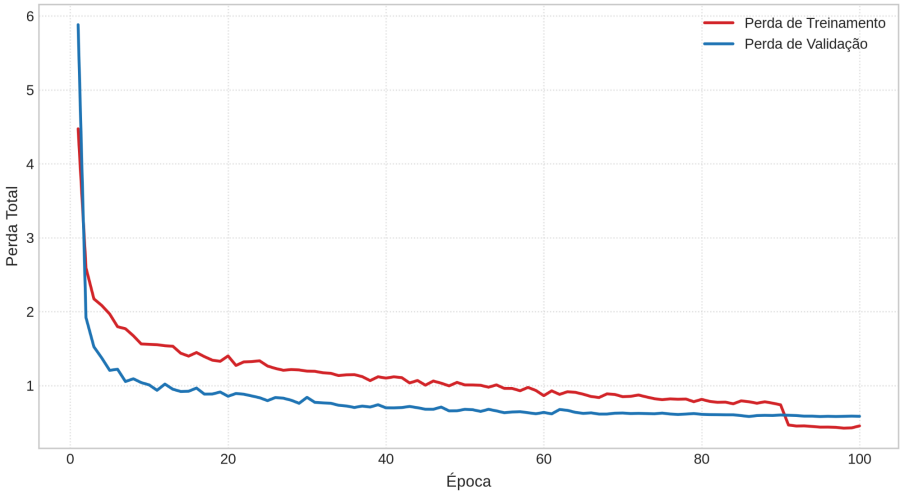

Figura 1. Gráfico YOLO26n: Train e Val Loss.

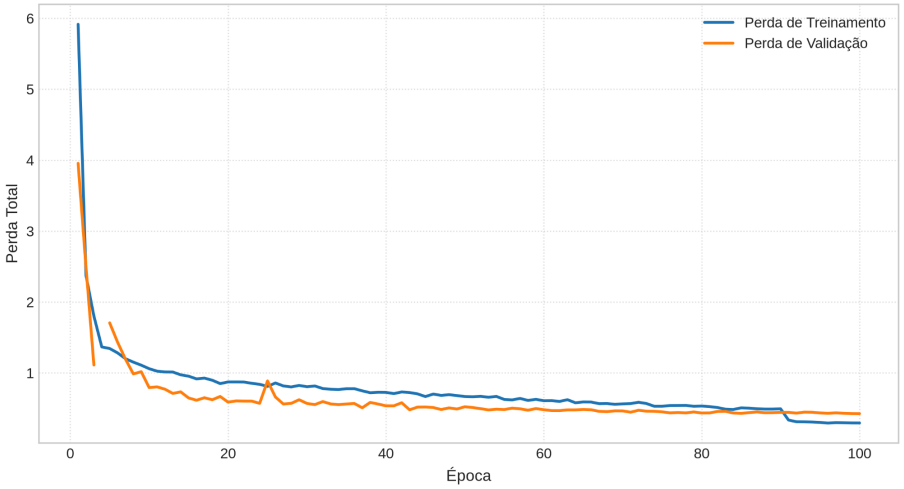

Figura 2. Gráfico RT-DETR-L: Train e Val Loss.

Com os resultados obtidos, é possível observar que, mesmo que o YOLO26n tenha se saído ligeiramente superior, a diferença na qualidade de detecção, medida através das métricas de precisão, *recall* e mAP, entre os dois modelos foi mínima, com a diferença máxima sendo de 0.60% na precisão geral das classes.

Quanto a avaliação do desempenho dos modelos, analisando as métricas GFLOPs e o número de parâmetros foi possível detectar que o modelo RT-DETR-L se prova ser muito mais pesado que o modelo YOLO, tendo um GFLOPs 23.07% menor, porém com aproximadamente 13.5 vezes mais parâmetros. Esse resultado evidencia que a quantidade de parâmetros pode representar um fator mais limitante que o número de operações para implantação em dispositivos com memória restrita.

Analisando os gráficos de perda dos modelos, é possível perceber um padrão clássico de um treinamento bem sucedido, com uma queda acentuada nas primeiras épocas e estabilização em valores baixos nas seguintes, sem caracterizar *overfitting* ou *underfitting*.

4. CONCLUSÃO

O objetivo desta pesquisa foi comparar os modelos YOLO26n e RT-DETR-L quanto ao desempenho de detecção e à complexidade computacional, buscando identificar qual arquitetura demonstra a melhor relação entre eficiência de detecção e viabilidade de implementação em dispositivos de baixo poder computacional. Os resultados mostram que, em termos de qualidade, os modelos apresentam poucas diferenças.

Entretanto, considerando a complexidade computacional, o YOLO26n se mostra a alternativa mais adequada ao objetivo deste trabalho, pois apesar de apresentar GFLOPs ligeiramente maior que o outro modelo, ele possui uma quantidade consideravelmente menor de parâmetros, o que indica em um menor custo de armazenamento e maior facilidade de implantação em dispositivos com recursos limitados.

É importante destacar que esses resultados foram obtidos de uma pesquisa ainda em fase inicial. O *dataset* utilizado, apesar de ser balanceado entre classes, é de tamanho reduzido. Portanto, os resultados devem ser interpretados como indicativos iniciais e não como conclusões definitivas de superioridade de um modelo sobre o outro.

Para trabalhos futuros, prevê-se a estruturação de um *dataset* mais abrangente e diversificado, ampliando tanto o volume de amostras quanto o leque de classes mapeadas. Ademais, pretende-se investigar a inclusão de abordagens de análise temporal, como séries temporais, visando detectar quedas considerando o movimento humano ao longo do tempo.

Por fim, visa-se a implementação de técnicas de otimização e quantização no modelo desenvolvido com o objetivo de viabilizar sua execução em dispositivos baseados em *Edge AI*.

Agradecimentos

Os autores agradecem à Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG) pelo apoio financeiro ao projeto OET-00243-26, bem como pela concessão de bolsa de Iniciação Científica ao primeiro autor. Agradecem também ao Instituto Federal do Norte de Minas Gerais (IFNMG) pelo apoio institucional.

REFERÊNCIAS

Alam, E. (2024), "GMDCSA24: A Dataset for Human Fall Detection in Videos", Zenodo, dataset. Disponível em: https://doi.org/10.5281/zenodo.12921216. Acesso em 20 de Julho, 2026.

Chakrabarty, S. (2026), "YOLO26: An analysis of NMS-free end to end framework for real-time object detection", arXiv preprint arXiv:2601.12882.

Charfi, I. *et al*. (2013), "Optimised spatio-temporal descriptors for real-time fall detection: comparison of SVM and Adaboost based classification", Journal of Electronic Imaging (JEI), vol 22, 041106-041106.

Haq, S., Verma, P. (2026), "An extensive examination of adaptive intelligence in cloud-to-edge systems for Healthcare 5.0", *Computers and Electrical Engineering*, vol 132, 111006.

Huang, X. *et al*. (2025), "SDES-YOLO: A high-precision and lightweight model for fall detection in complex environments", Scientific Reports, vol 15, 2026.

Mendez, J., Bierzynski, K., Cuéllar, M. P., Morales, D. P. (2022), "Edge Intelligence: Concepts, Architectures, Applications, and Future Directions", *ACM Transactions on Embedded Computing Systems*, vol 21, n. 5, 1-41.

Ren, L.; Peng, Y. (2019). Research of fall detection and fall prevention technologies: A systematic review. IEEE access, vol 7, 77702-77722.

Roboflow (2024), "UR Fall *Dataset*", Roboflow Universe. Disponível em: https://universe.roboflow.com/fall-detection-w7nxl/ur-fall. Acesso em: 19 jul. 2026.

Ungar, A. *et al.* (2013), "Fall prevention in the elderly", Clinical Cases in Mineral and Bone Metabolism, vol 10, 91-95.

Ursul, I. (2025), "Video-Based Fall Detection *Dataset* with 2017 Activities from 29 Subjects", figshare, *dataset*.

World Health Organization. (2025). WHO, Ageing and Health. Disponível em: https://www.who.int/news-room/fact-sheets/detail/ageing-and-health. Acesso em: 19 de Julho, 2026.

Zhao, D. *et al*. (2024), "YOLO-Fall: A Novel Convolutional Neural Network Model for Fall Detection in Open Spaces", IEEE Access, vol 12, 26137-26149.

Zhao, Y. *et al*. (2024), "DETRs Beat YOLOs on Real-time Object Detection", Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 16965-16974.