DESEMPENHO E CUSTO TÉRMICO DE ACELERADORES DE HARDWARE PARA SEGMENTAÇÃO DE INSTÂNCIAS EM TEMPO REAL EM DISPOSITIVOS MÓVEIS

MARCELO C. RIBEIRO[1]; TIAGO T. PRIMO[1]; ULISSES B. CORRÊA[1]; RAFAEL G. LUND[2]; MARILTON S. DE AGUIAR[1]

[1] *Universidade Federal de Pelotas – {mcribeiro,tiago.primo,ulisses,marilton}@inf.ufpel.edu.br*
[2] *Universidade Federal de Pelotas – rafael.lund@gmail.com*

1. INTRODUÇÃO

O processamento em tempo real de vídeo médico diretamente em dispositivos móveis, voltado à segmentação de instâncias para destacar estruturas anatômicas, ainda é pouco explorado. Câmeras acopladas a procedimentos como exames endoscópicos e videolaringoscopia orientam a inspeção e manipulação de estruturas internas; na videolaringoscopia, por exemplo, a visualização indireta das vias aéreas é especialmente relevante na intubação endotraqueal (WU et al., 2025).

A identificação automática dessas estruturas pode facilitar a interpretação do exame pelo profissional. Técnicas de *deep learning* têm avançado em detecção, classificação e segmentação de imagens médicas (WANG et al., 2021), mas tarefas em tempo real exigem arquiteturas otimizadas para baixa latência, diferente de uma classificação isolada.

Nesse contexto, destaca-se a família YOLO (*You Only Look Once*), conhecida pela detecção de objetos em tempo real (REDMON et al., 2015). Suas variantes com segmentação de instâncias permitem delimitar espacialmente regiões de interesse, tornando-as adequadas a imagens médicas. Executar esses modelos diretamente no dispositivo móvel, em vez de depender de processamento remoto, aproxima o processamento da captura da imagem, reduzindo latência, dependência de infraestrutura externa e exposição de dados sensíveis (DONG et al., 2023), características relevantes em ambientes clínicos, onde conectividade nem sempre é garantida e a privacidade do paciente é essencial.

Apesar do potencial dessa abordagem, há pouca caracterização do desempenho de modelos de segmentação em tempo real executados diretamente em *smartphones*, especialmente na comparação entre diferentes caminhos de aceleração de hardware disponíveis nesses aparelhos. Este trabalho busca preencher essa lacuna, apresentando um aplicativo Android que captura vídeo de uma câmera USB externa (conectada via OTG) e executa a segmentação de instâncias no próprio dispositivo. A avaliação caracteriza o desempenho da solução sob diferentes condições, comparando dois caminhos de aceleração de hardware disponíveis no aparelho, sob duas taxas de alimentação da câmera, com medição separada das etapas de pré-processamento, inferência e pós-processamento.

2. METODOLOGIA

A solução segue o fluxo típico de uma aplicação de visão computacional embarcada: captura do quadro pela câmera, pré-processamento, inferência, pós-processamento e exibição do resultado. O aplicativo foi desenvolvido para Android, com interface em *Flutter* (visando futuro reaproveitamento para iOS) e rotinas de captura e inferência em *Kotlin*, comunicando-se por canais de método e de eventos, dado o acesso nativo exigido pelas bibliotecas de captura USB e aceleração de hardware. A câmera é um dispositivo UVC (*USB Video Class*) conectado por cabo OTG e controlado pela biblioteca AUSBC, que entrega os quadros já convertidos

para NV21. Optou-se por câmera USB externa em vez da câmera integrada, cabendo ao dispositivo móvel o processamento dos dados e a visualização do exame.

O processamento de cada quadro divide-se em três etapas, medidas separadamente. No pré-processamento, o quadro é ajustado para a resolução de entrada da rede (640×640), preservando a proporção original, e normalizado. Na inferência, o dado é submetido à rede. O pós-processamento, etapa mais elaborada, decodifica as saídas para obter detecções e máscaras: seleciona-se a classe de maior escore, aplica-se supressão de não máximos às detecções acima do limiar de confiança, e as máscaras são reconstruídas a partir dos coeficientes e protótipos da rede.

Versões mais recentes da família YOLO, como YOLO11 e YOLO26, apresentam incompatibilidade com o compilador do fabricante do dispositivo, impedindo a conversão para o formato proprietário da unidade de processamento neural (*NPU*). Por isso, adotou-se a YOLOv8n-seg, pré-treinada no conjunto COCO (LIN et al., 2014), por ser a versão mais recente com segmentação compatível com ambos os destinos avaliados. O modelo funciona exclusivamente como carga de referência para a avaliação de processamento, sem que sua acurácia seja objeto deste trabalho.

Dois caminhos de execução foram implementados, alternáveis sem recompilação: um genérico e portável, outro otimizado e proprietário. O primeiro emprega o modelo em LiteRT para uso na GPU, via exportação no formato `.tflite`. O segundo recorre ao *framework* ENN da Samsung, que executa um modelo `.nnc` compilado especificamente para o *chipset* de destino, acessado por uma ponte JNI (*Java Native Interface*) própria, em troca de maior aproveitamento do hardware dedicado.

Os experimentos foram conduzidos em um Samsung Galaxy S25 FE (chip Exynos 2400, Android 16), com câmera OTG configurada em 1280×720 no formato MJPEG. Avaliaram-se quatro configurações, combinando os dois aceleradores com duas taxas de alimentação da câmera (30 e 60 quadros por segundo). Cada configuração foi executada por dez minutos sobre uma cena estática, com iluminação constante e três objetos das categorias do conjunto COCO, de modo a isolar o efeito do hardware sobre o desempenho.

O protocolo de medição registra, para cada quadro, uma linha com os tempos das três etapas e o número de detecções, usado como controle de validade: sua estabilidade garante que a cena permaneceu inalterada e que as medições são comparáveis entre si. Consumo de memória gráfica, temperatura e estado térmico do sistema foram amostrados a cada minuto. O primeiro minuto de cada execução foi descartado da análise, por corresponder a um regime transitório de escalonamento de frequência que, em medições preliminares, chegou a distorcer os tempos em 18%.

3. RESULTADOS E DISCUSSÃO

Esta seção apresenta o desempenho das quatro configurações avaliadas, comparando os dois caminhos de aceleração sob as duas taxas de alimentação da câmera, e discute o custo térmico associado a cada abordagem. A Tabela 1 apresenta as medianas das três etapas de processamento, além da taxa de quadros efetivamente processada. Todas as configurações sustentam taxas entre 27 e 33 quadros por segundo, compatíveis com uma aplicação de processamento em tempo real. A inferência responde por cerca de dois terços do tempo de quadro; o

pré e o pós-processamento somados custam menos de 10 ms.

Tabela 1: Desempenho por configuração. Tempos em milissegundos (medianas).

Acelerador	Câmera	Pré	Inferência	Pós	Total	FPS
NPU (ENN)	60 fps	5	**21**	3	**30**	**32,5**
NPU (ENN)	30 fps	5	22	4	31	28,9
GPU (LiteRT)	60 fps	6	25	3	35	26,9
GPU (LiteRT)	30 fps	6	25	3	34	28,2

Esse resultado agregado esconde, porém, um comportamento menos previsível ao longo do tempo. A execução sobre a NPU manteve-se estável desde a primeira amostra, com variação de apenas 21 para 22 ms ao longo dos dez minutos. Já sobre a GPU, a variação foi pronunciada: partindo de 24 ms, o tempo de inferência subiu progressivamente até 38 ms por volta do quinto minuto, retornando em seguida a 24 ms e permanecendo estável até o fim da execução. O fenômeno repetiu-se nas duas configurações de GPU, na mesma janela temporal, e não se explica pela cena (inalterada) nem por limitação térmica, já que o aparelho seguiu aquecendo enquanto o desempenho se recuperava; hipóteses como aquecimento de cache ou *warm-up* de compilação permanecem em aberto, sem confirmação pelos dados coletados. Descartada essa oscilação, as quatro configurações convergem para tempos totais de 32 a 33 ms, reduzindo a diferença entre os aceleradores a cerca de 2 ms de inferência.

A alimentação da câmera a 30 ou a 60 quadros por segundo não produziu diferença expressiva em regime permanente, o que sugere que o fator limitante é o *pipeline* de processamento, não a taxa de captura. A configuração a 60 quadros por segundo apresentou maior dispersão nos tempos, coerente com o descarte de quadros que a taxa excedente impõe. Ainda que o desempenho bruto dos dois aceleradores seja próximo, o custo para sustentá-lo difere de maneira significativa, como mostra a Tabela 2.

Tabela 2: Temperatura máxima, estado térmico do sistema e memória gráfica.

Acelerador	Câmera	SoC	Superfície	Estado	Memória
NPU (ENN)	60 fps	**47,8 °C**	**38,8 °C**	**0**	**189 MB**
NPU (ENN)	30 fps	**46,2 °C**	**38,3 °C**	**0**	**189 MB**
GPU (LiteRT)	60 fps	54,7 °C	42,9 °C	2	212 MB
GPU (LiteRT)	30 fps	53,2 °C	42,2 °C	2	213 MB

Nas duas configurações de GPU, o sistema operacional declarou limitação térmica de nível 2 (moderado), ao passo que as configurações de NPU permaneceram em nível zero durante toda a execução, com temperatura de superfície cerca de 4 °C inferior e consumo de memória gráfica 23 MB menor. A diferença não aparece na taxa de quadros, mas importa no cenário de uso pretendido: o aparelho é manipulado pelo profissional durante o exame, e execuções prolongadas em estado de limitação térmica comprometem o conforto e a previsibilidade do desempenho. Para cargas de trabalho de custo semelhante à avaliada, a unidade de processamento neural sustenta o mesmo desempenho com margem térmica significativa, o que reforça a adequação desse caminho de aceleração ao uso clínico continuado.

Cabe também delimitar o escopo desta avaliação. As medições foram obtidas em um único aparelho, e a dependência do formato `.nnc` em relação ao *chipset* impede a generalização direta a outros dispositivos. A carga avaliada corresponde a um modelo treinado para 80 classes; medições preliminares indicam que o tempo de execução na NPU cresce aproximadamente proporcional ao volume do tensor

de saída, de modo que modelos com menor número de classes tendem a ampliar a vantagem observada.

4. CONCLUSÕES

Este trabalho caracterizou o desempenho de modelos de segmentação em tempo real executados em *smartphones*, por meio de um aplicativo que processa, no próprio dispositivo, vídeo de câmera USB externa. As quatro configurações avaliadas sustentaram taxas entre 27 e 33 quadros por segundo, com desempenho bruto semelhante entre os dois caminhos de aceleração testados. A principal diferença esteve no custo térmico: a GPU levou o sistema a limitação térmica moderada, enquanto a NPU sustentou o mesmo desempenho em nível térmico zero, o que preserva o conforto de manuseio e reduz o risco de degradação em uso clínico prolongado, tornando-a o caminho mais adequado.

Como trabalhos futuros, pretende-se avaliar modelos com menor número de categorias e validar a acurácia sobre imagens de exames anotadas por especialistas, etapa não contemplada nesta avaliação. Os resultados indicam a viabilidade da segmentação de instâncias em tempo real embarcada em dispositivos móveis, com vantagem prática da NPU em uso prolongado.

5. AGRADECIMENTOS

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio à realização desta pesquisa, por meio da concessão de bolsas e auxílio financeiro (Processos CNPq: 406417/2024-5 e 382388/2025-9).

6. REFERÊNCIAS BIBLIOGRÁFICAS

DONG, C.; LI, T. Z.; XU, K.; WANG, Z.; MALDONADO, F.; SANDLER, K.; LANDMAN, B. A.; HUO, Y. Characterizing browser-based medical imaging ai with serverless edge computing: towards addressing clinical data security constraints. In: **Proceedings of SPIE**. Bellingham, WA, USA: SPIE, 2023. v. 12469.

LIN, T.-Y.; MAIRE, M.; BELONGIE, S.; BOURDEV, L.; GIRSHICK, R.; HAYS, J.; PERONA, P.; RAMANAN, D.; ZITNICK, C. L.; DOLLáR, P. **Microsoft COCO: Common Objects in Context**. 2014.

REDMON, J.; DIVVALA, S. K.; GIRSHICK, R. B.; FARHADI, A. You only look once: Unified, real-time object detection. **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, p. 779–788, 2015.

WANG, J.; ZHU, H.; WANG, S.-H.; ZHANG, Y.-D. A review of deep learning on medical image analysis. **Mobile Networks and Applications**, Springer, Berlin, Germany, v. 26, n. 1, p. 351–380, 2021.

WU, J.; GUO, W.; CHEN, Z.; HU, H.; LI, H.; ZHANG, Y.; HUANG, J.; LIU, L.; XU, Z.; XU, T.; ZHOU, M.; ZHU, C.; CUI, H.; XU, W.; ZOU, Z. A segmentation network based on cnns for identifying laryngeal structures in video laryngoscope images. **Computerized Medical Imaging and Graphics**, v. 124, p. 102573, 2025. ISSN 0895-6111.