

TinyScope: Lightweight Hyperspectral Tissue Classification Using TinyML Inference on a Single-board Computer

RokunuzJahan Rudro ^{a,b}, Ling Ma ^{a,b}, Kelden Pruitt ^{a,b}, Baowei Fei ^{a,b,c,*}

^a Center for Imaging and Surgical Innovation, University of Texas at Dallas, Richardson, TX

^b Department of Bioengineering, University of Texas at Dallas, Richardson, TX

^c Department of Radiology, University of Texas Southwestern Medical Center, Dallas, TX

*Corresponding author: bfei@utdallas.edu, Website: <https://fei-lab.org>

ABSTRACT

Hyperspectral imaging (HSI) provides high-resolution spectral information capable of distinguishing subtle physiological differences across tissue types, offering significant promise for surgical guidance and point-of-care diagnostics. However, the high dimensionality and computational demands of HSI data have posed barriers to its real-time deployment on low-power embedded platforms. In this work, we develop an end-to-end, deployable pipeline for tissue-level HSI classification on a Raspberry Pi 5 single-board computer, enabling fast and memory-efficient inference without the need for specialized accelerators. A dataset comprising of 630 hyperspectral images (*i.e.*, hypercubes) of *ex vivo* abdominal tissues across six organ types and four animal models were curated using snapshot HSI cameras. Our approach uses a ResNet-18 model pretrained on ImageNet dataset and modified to accept 16-channel inputs, was fine-tuned to perform multi-tissue classification, achieving a top 1 accuracy of 85.26% using full-precision (FP32) weights. The trained models were then exported to TorchScript and subjected to post-training dynamic quantization. For the INT-8 ResNet-18 model, this resulted in a fourfold reduction in parameter memory (from 44.9 MB to 11.2 MB) and a substantial decrease in peak RAM usage on Pi 5, from 25.15 MB to 6.31 MB, with no loss in classification accuracy. Benchmarks conducted on the Raspberry Pi 5 CPU indicate that the quantized INT-8 model achieves a mean inference latency of 149.42 ms per $128 \times 128 \times 16$ hyperspectral data cube, corresponding to a throughput of 6.37 hypercubes per second. These results demonstrate that high performance HSI classification can be achieved on an embedded hardware costing less than \$200, using widely available convolutional architectures and quantization techniques. The proposed pipeline thus offers a practical, accessible, and compact solution for deploying HSI analytics in clinical settings, particularly promising for real-time surgical guidance and point-of-care applications.

Keywords: Tiny machine learning (TinyML), hyperspectral imaging (HSI), single-board computer, deep learning, optimization, classification, deployment, embedded machine learning, quantization

INTRODUCTION

Hyperspectral imaging (HSI) acquires a full reflectance spectrum at every image pixel, offering rich biochemical and structural insights that often elude traditional RGB or fluorescence systems. In early applications of medical HSI, researchers highlighted its potential for perfusion mapping, cancer margin detection, and organ differentiation¹, while large amount of HSI data requires high computation power for image acquisition and processing¹. Recent advancements have addressed hardware limitations, for example, Pruitt *et al.* developed a high-speed laparoscopic HSI system capable of capturing up to 25 frames per second, validating its technical feasibility in simulated intraoperative settings and motivating our *ex vivo* data collection effort². Video-rate laparoscopic systems now achieve 25 hypercubes per second by integrating snapshot cameras into standard surgical optics, demonstrating the feasibility of real-time intra-operative spectral capture³. Along with these developments, low-cost, Raspberry-Pi-based spectrometers have emerged for point-of-care (POC) applications, highlighting a growing demand for compact HSI hardware⁴. Yet the analytics pipeline has not kept pace, as state-of-the-art medical HSI studies continue to rely on high-performance clusters and expensive graphic processing units (GPUs) for inference, posing significant barriers in terms of cost and accessibility, even when targeting time-critical tasks such as brain tumor delineation⁵ or large-scale organ fingerprinting⁶.

Concurrently, the tiny-machine-learning (TinyML) community has shown that edge devices can execute sophisticated models for electrocardiogram monitoring, anomaly detection, and other biomedical workloads at milliwatt power budgets⁷. In this work, we build upon this premise and address the gaps and limitations by presenting, to our knowledge, the first tissue-level HSI classification pipeline deployable on a Raspberry Pi 5 single-board computer platform⁸. We build upon the ResNet-18 architecture⁹ and modify the first convolutional layer to accept 16 channels hyperspectral input. The network is then fine-tuned on a multi-species, multi-organ dataset to learn spectral-spatial features relevant to tissue classification. Finally, we applied dynamic INT-8 dynamic quantization for deployment. This helps reduce the memory usage (ROM and RAM) as well as the inference latency without relying on specialized GPU accelerators¹⁰. By demonstrating a complete workflow from acquisition to embedded inference, our study offers a concrete template for translating the rich diagnostic contrast of HSI from the benchtop to affordable, POC hardware, potentially facilitating real-time surgical guidance and other applications.

METHODS

Figure 1 summarizes the end-to-end workflow from hyperspectral data ingestion to the compressed model deployment on a Raspberry Pi 5. The pipeline includes preprocessing, supervised fine-tuning, INT-8 dynamic quantization, and optimized inference at the edge.

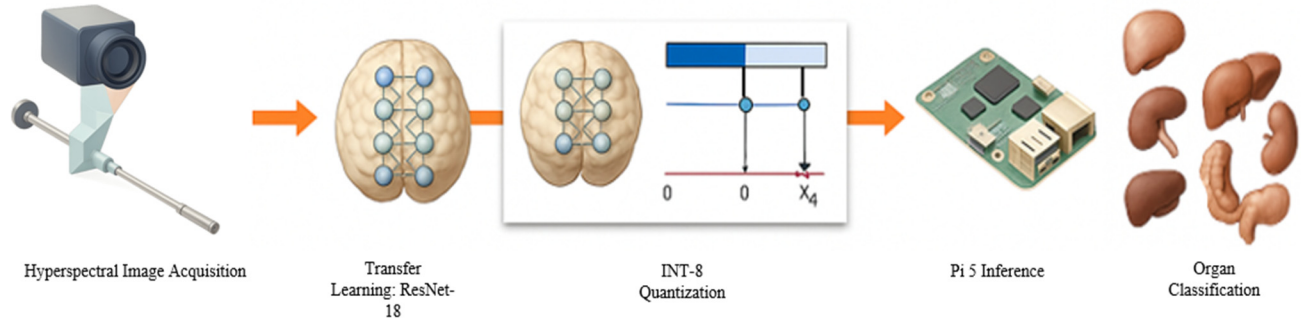


Figure 1. Methodology pipeline of the proposed hyperspectral imaging and hardware-based classification framework.

2.1 Dataset Acquisition and Preprocessing

We acquired HSI dataset from six organ classes (liver, intestine, spleen, stomach, kidney, and uterus), which were collected with a custom laparoscopic HSI system². Figure 2 shows some sample HSI-synthetic RGB images from our dataset. The resulting synthetic RGB images exhibit minor spatial-spectral misregistration artifacts at the image periphery. These non-overlapping regions are attributed to optical chromatic aberration and the boundary constraints of the spectral demosaicing algorithm, which limit the valid spectral reconstruction area at the sensor edges. Three synchronized snapshot HSI cameras covered complementary wavelengths between 460–960 nm and were coupled via a beam-splitting optical path to a 10 mm, 0° laparoscope. Illumination was provided by a 150 W halogen light source. The system captured hyperspectral images of *ex vivo* organ specimens from pigs, cows, chickens, and mice under controlled illumination. Acquired cubes were dark-current corrected and calibrated with a 95% reflectance reference tile:

$$I_{\text{reflectance}}(\lambda) = \frac{I_{\text{processed}}(\lambda)}{I_{\text{WR}}(\lambda)}$$

where I_{WR} is the white reference and $I_{\text{processed}}$ is the dark current-corrected intensity at wavelength λ .

Each camera captured a different spectral range, providing 15, 16 and 24 bands respectively. The raw images were individually calibrated to correct for sensor specific spectral and intensity variations. Following calibration, the images were spatially registered to ensure alignment across all three camera views. Finally, the calibrated and aligned spectral stacks were concatenated along the spectral axis, producing a unified spectral data cube of size $510 \times 270 \times 55$, where 55 represents the total number of spectral bands obtained by combining the three sensor outputs². Since deployment was targeted for resource-constrained hardware, we retained the first 16 visible-range bands for further processing. We stratified the dataset into 70% training, 15% validation, and 15% test splits, resulting in 441 and 94 images for training

and validation, and 94 images held out for testing. The test set was only preserved for final benchmarking on Pi 5 by exporting the test samples into NumPy format which enabled offline inference thereby simulating real-world deployment scenarios for embedded HSI classification. This approach simulates a realistic deployment scenario, where test samples are passed directly to the model in a resource-constrained environment, while training and validation were conducted entirely on a high-performance workstation. Figure 3 summarizes the training, validation and testing distribution.

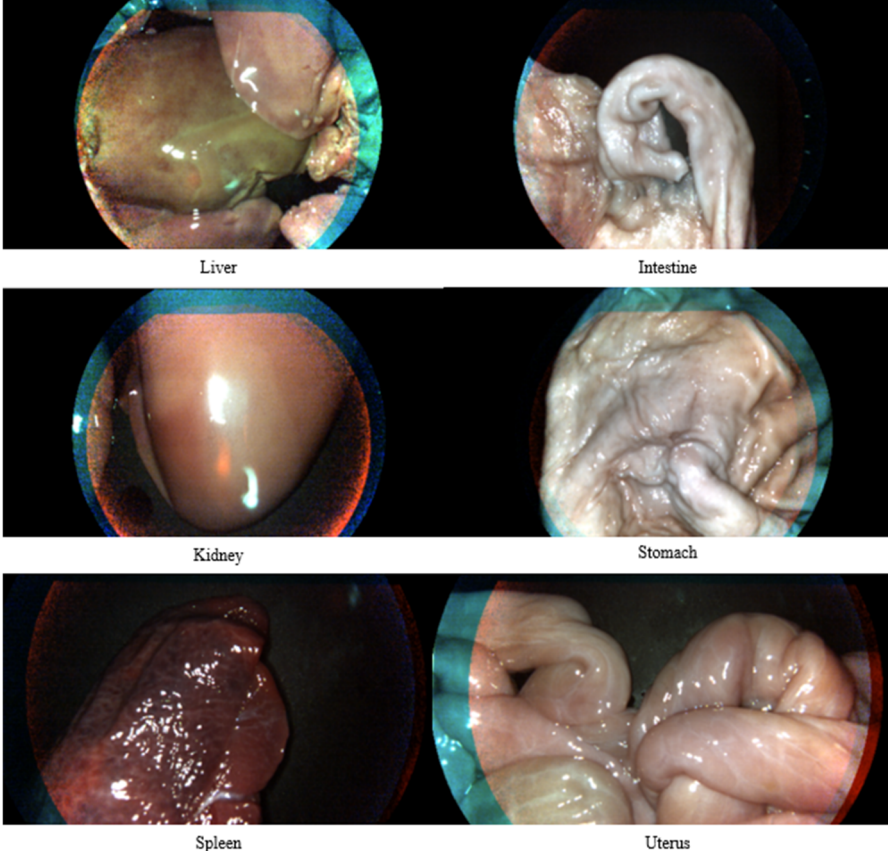


Figure 2. Example images of the six organs (liver, intestine, spleen, stomach, kidney, and uterus) in the hyperspectral image dataset.

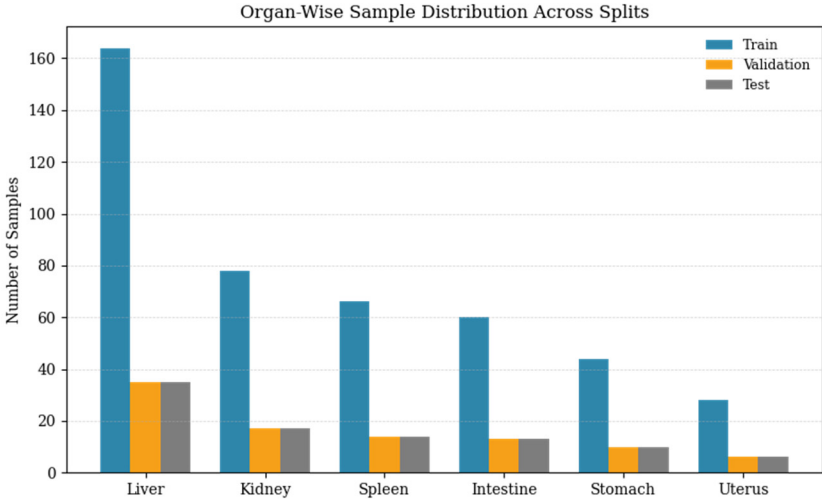


Figure 3. Dataset distribution for model training, validation and testing.

2.2 Model Training

We employed a transfer learning strategy leveraging the widely used PyTorch vision library, which offers numerous state-of-the-art pretrained models^{9,11,12}. Specifically, the ImageNet pretrained ResNet-18⁹ was adopted as the backbone of the model, as its residual skip connections deliver state-of-the-art accuracy with only 11M parameters, making it an effective foundation for edge deployment. This approach allowed us to benefit from pretrained feature representations and accelerate convergence despite the limited size of our domain-specific dataset¹¹. The first 7×7 convolution was re-initialized to accept 16 spectral bands, and the classification head was replaced with a six-unit linear layer for each of the six classes. The network was fine-tuned for 100 epochs with batch size of 64 using the AdamW optimizer with an initial learning rate (LR) of 3×10^{-4} and weight-decay of 10^{-2} . The ReduceLROnPlateau learning rate scheduler was used to automatically reduce the LR by a factor of 0.2 when the validation loss plateaued for three consecutive epochs¹³. Cross-entropy loss was implemented with label smoothing $\varepsilon = 0.05$:

$$\mathcal{L}_L = -(1 - \varepsilon) \log p_y - \frac{\varepsilon}{k - 1} \sum_{k \neq y} \log p_k$$

where k is the number of organ classes and p_k is the softmax probability for class k . The model was trained on a workstation with an NVIDIA RTX-A200 with 12 GB memory and 64 GB RAM. Figure 4 illustrates the evolution of the training and validation loss over the training period. Effective learning and reduced overfitting from the model can be seen from the downward loss curves and increasing accuracy curves.

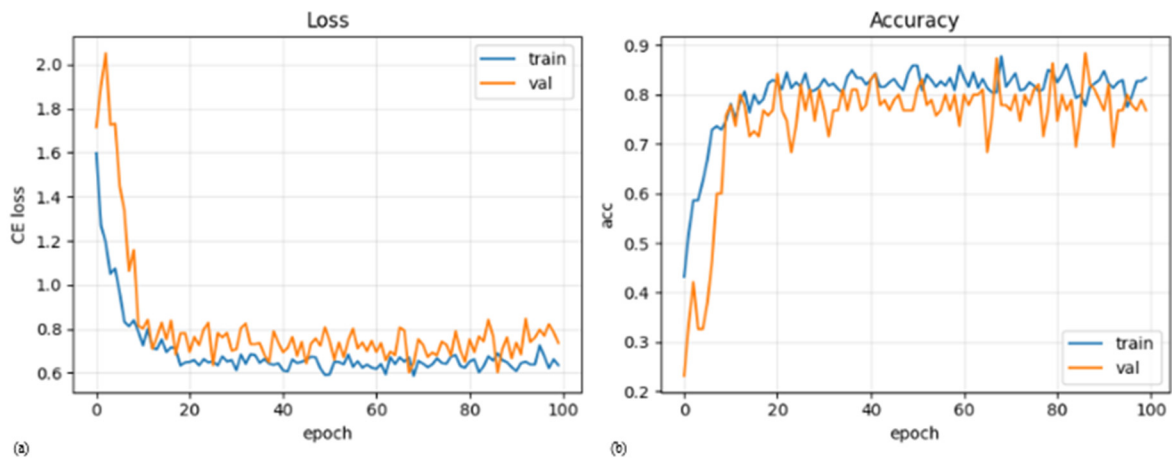


Figure 4. Model training overview. (a) Training and validation loss. (b) Convergence behavior during training.

2.3 Model Compression

After training the ResNet-18 model in full precision, we applied post-training dynamic quantization to reduce the precision of the fully connected (FC) layers from 32-bit floating point to 8-bit integers. We used PyTorch's dynamic quantization API to selectively quantize only the FC subgraph, while retaining all convolutional layers in FP32¹⁴. Dynamic quantization was chosen over static quantization because it does not require a separate calibration dataset or additional preprocessing: activation ranges are estimated on the fly at inference time, rather than from a fixed calibration set. Keeping the convolutional backbone in FP32 preserves the spatial feature representations learned by the early and intermediate layers, which are typically more sensitive to post-training quantization noise.

Despite this mixed-precision design, the quantized model still benefits from optimized integer matrix multiplications on the deployment hardware. On the Raspberry Pi 5, the ARMv8-A NEON SIMD unit in the Cortex-A76 CPU cores provides native support for 8-bit dot-product instructions, enabling INT8 GEMM kernels to execute with higher throughput and lower memory bandwidth than their FP32 counterparts. In our configuration, these INT8 kernel optimizations apply to the FC layers, which are evaluated as $\text{INT8} \times \text{INT8} \rightarrow \text{INT32}$ matrix multiplies followed by a lightweight rescaling back to FP32. Section 2.4 describes the Pi 5 microarchitectural improvements in more detail and explains how they benefit

low-precision inference workloads. Mathematically, an 8-bit weight q is obtained from its floating-point counterpart w via:

$$q = \text{clip} \left(\text{round} \left(\frac{w}{s} \right) + z, -128, 127 \right), s = \frac{\max|w|}{127}$$

where s is the per-tensor scale and z is the zero-point¹⁵. In our setup, the weight tensors in the FC layers are quantized once at conversion time, with (s_w, z_w) computed from the trained FP32 weights and stored permanently in INT8 form. During inference, the activations X entering a quantized FC layer remain in FP32 until runtime, when a per-batch scale s_X (and optionally zero-point z_X) is derived from the current activation range; X is then quantized, multiplied with the INT8 weights, accumulated in INT32, and finally de-quantized back to FP32. This procedure yields an approximate $4 \times$ reduction in weight storage for the FC layers and enables efficient low-precision execution without modifying the original network architecture or retraining the model.

Standard confusion matrix was used to measure the performance using True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN). The definitions for accuracy, precision, recall and F1-score are listed below:

$$\begin{aligned} \text{Accuracy} &= \frac{TP + TN}{TP + TN + FP + FN} \\ \text{Precision} &= \frac{TP}{TP + FP} \\ \text{Recall} &= \frac{TP}{TP + FN} \\ \text{F1 - Score} &= 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \end{aligned}$$

2.4 Model Inference on Raspberry Pi 5

The Raspberry Pi 5 serves as the deployment platform for our quantized HSI classification model, offering a practical balance between computational capability, cost, and portability. As the latest generation of the Pi family, the Pi 5 introduces substantial performance upgrades over the Raspberry Pi 4, most notably a 64-bit quad-core ARM Cortex-A76 CPU running at 2.4 GHz and support for up to 8 GB of LPDDR4X SDRAM. These new additions provide the necessary headroom for executing lightweight deep learning workloads without relying on external accelerators or cloud resources.

For this project, the Pi 5 enables standalone on-device inference of the quantized INT-8 ResNet-18 model. Its CPU architecture includes ARMv8-A NEON SIMD extensions, which accelerate integer operations and contribute to the observed reduction in inference latency following quantization. Combined with the onboard I/O capabilities including Bluetooth 5/6, Wi-Fi, and high-speed USB interfaces, the Pi 5 can support a compact imaging pipeline where image acquisition, preprocessing, model inference, and visualization can be integrated within a single device.

While the current framework performs inference on pre-exported NumPy test samples, the setup validates the hardware foundation for a fully embedded HSI device. We exported two TorchScript models: FP32 and INT-8. These were deployed without additional dependencies, and inference was performed entirely on the Pi 5 CPU. Each NumPy cube ($16 \times 128 \times 128$) was streamed through the model, with per-hypercube inference latency, peak RAM usage, throughput, and accuracy being recorded.

RESULTS

3.1 Model Size and Memory Consumption

INT8 quantization slashes the parameter footprint from 44.9 MB to 11.2 MB while keeping the ROM size fixed at 45.1 MB, aligning with the expected $4 \times$ savings from reducing weights to 8 bits. On-device RAM demand drops even more sharply: peak runtime memory fell from 25.51 MB to 2.62 MB on Pi 5, confirming that low-bit activations

substantially ease embedded deployment. These reductions are consistent with prior findings that INT-8 quantization can yield significant memory savings in deep CNNs without sacrificing accuracy^{10,15}.

3.2 Latency and Throughput

The deployment of deep neural networks on resource-constrained edge devices requires a critical optimization of the trade-off between predictive accuracy and computational efficiency. In real-time medical applications, minimizing inference latency is crucial to ensuring that the system provides immediate, actionable feedback during clinical procedures. To address this, we evaluated the impact of low-precision quantization on our model's operational speed such as the inference and throughput¹⁶. Inference latency, which measures the time taken to process a single hyperspectral cube, directly affects system responsiveness in real-time applications. Throughput, defined as the number of inputs processed per second, is often inversely related to latency in sequential pipelines such as our work. The INT-8 quantized model demonstrated a modest yet consistent improvement in mean inference latency, decreasing from 151.87 ms in the FP-32 model to 149.42 ms with a reduction of approximately 2.45 ms or 1.6%. In conjunction with this reduction in latency, model throughput increased slightly from 6.28 to 6.37 cubes per second, reflecting a 1.4% improvement.

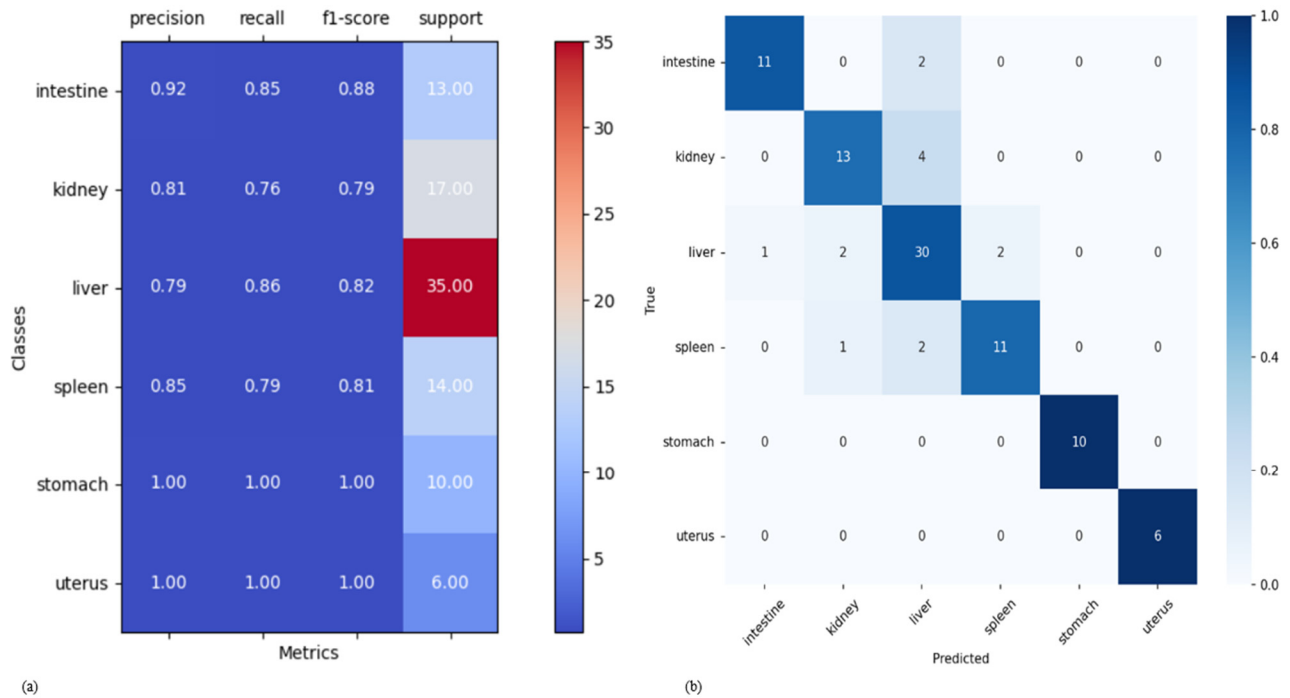


Figure 5. Performance metrics of the quantized model: (a) Classification report. (b) Confusion matrix.

3.3 Classification Accuracy

The assessment of deep learning model performance is fundamentally reliant upon its evaluation against an independent and representative test dataset, a prerequisite for establishing its capacity for generalization beyond the training environment¹⁷. A critical aspect of model optimization, specifically quantization, involves verifying that the reduced precision format does not introduce a statistically significant degradation in predictive power. Both FP-32 and INT-8 models maintained identical top-1 accuracy of 85.26% on the test dataset. Figure 5 shows the classification matrix and classification report of the quantized INT-8 model which were used to assess the model's performance¹⁷. The model correctly predicted all uterus samples (6/6) and stomach (10/10) samples. However, the model made a few misclassifications between anatomically similar organs such as kidney and liver which show non-zero off-diagonal entries.

DISCUSSIONS AND CONCLUSIONS

Our results demonstrate that standard convolutional backbones, when paired with dynamic INT-8 quantization, can enable real-time HSI analytics on hardware costing less than \$200 effectively, removing a long-standing barrier to POC deployment. Although the improvement in latency may appear modest, it ensures that inference consistently remains below the 200 ms upper bound typically required for surgical or other time-critical imaging workflows. This responsiveness, combined with a slight increase in throughput, highlights the computational efficiency gained through quantization. As a future step, we aim to develop a custom Python-based imaging application that will help clinicians to capture, save, and analyze hyperspectral images directly within the application itself. This tool would serve as a complete end-to-end interface, combining acquisition, preprocessing, classification, and visualization in a single application for clinical use.

This work delivers a deployable, low-cost HSI pipeline that preserves the classification performance while meeting real-time constraints, directly advancing embedded ML for biomedical applications. Within anesthesiology, the same edge-inference capability may support guidance by providing rapid tissue or structure differentiation during orotracheal intubation¹⁸, reducing reliance on external compute and improving workflow integration in the operating room. Likewise, endoscopic systems can benefit from on-scope HSI analytics for tissue characterization without offloading data to a server, enhancing decision support and maintaining data privacy.

Future work will focus on integrating the quantized model into fully handheld medical devices or single board-mounted medical imaging devices. This will port the inference to other microcontrollers and single board computers, alongside exploring more compact and hardware-friendly architectures, such as MobileNet or Vision transformers¹⁰ through quantization-aware training¹⁹ and knowledge distillation^{20,21}. In addition to Raspberry Pi class devices, higher-end edge AI modules such as the NVIDIA Jetson Orin Nano²², which provides a 6-core Arm Cortex-A78AE CPU, an Ampere GPU with Tensor Cores, and up to tens of INT8 TOPS for dense inference, offer a promising platform for running richer HSI models and small-scale multimodal pipelines at the point of care. Recent advances in small-scale vision-language models such as SmolVLM²³, PaliGemma²⁴, and Qwen3-VL²⁵ also offer exciting new opportunities to extend hyperspectral imaging with multi-modal reasoning capabilities in constrained environments. Looking forward, broader adoption of HSI analytics in embedded systems will depend on the availability of next-generation microcontroller platforms and single board computers that natively support ML inference with acceptable latency and throughput. While current low-power chips such as the ESP32 and Raspberry Pi Pico are limited in their ability to handle even quantized convolutional models, recent trends in hardware design such as the integration of dedicated neural accelerators as in EdgeTPU and increased toolchain maturity like TVM, CMSIS-NN and TensorFlow Lite Micro suggest a growing feasibility of deploying compact HSI classifiers at the extreme edge^{26–29}. As these hardware and software ecosystems mature, future work can shift toward developing tiny spectral models optimized for constrained deployment that can reach real-time systems.

ACKNOWLEDGEMENTS

Research reported in this publication was supported in part by the National Cancer Institute of the National Institutes of Health under Award Number R01CA288379 and by the Cancer Prevention and Research Institute of Texas (CPRIT) under Award Number RP240289 and RP240542. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

REFERENCES

- [1] Lu, G. and Fei, B., “Medical hyperspectral imaging: a review,” *J Biomed Opt* **19**(1), 010901 (2014).
- [2] Pruitt, K., DeAtley, W., Rathgeb, A., Johnson, B., Gahan, J. and Fei, B., “A high-speed hyperspectral imaging system and large-scale hyperspectral dataset for abdominal surgical applications,” *Medical Imaging 2025: Image-Guided Procedures, Robotic Interventions, and Modeling* **13408**, 69–76, SPIE (2025).
- [3] Ayala, L., Wirkert, S., Vemuri, A., Adler, T., Seidlitz, S., Pirmann, S., Engels, C., Teber, D. and Maier-Hein, L., “Video-rate multispectral imaging in laparoscopic surgery: First-in-human application,” *Sci. Adv.* **9**(10), eadd6778 (2023).
- [4] Pechlivani, E. M., Papadimitriou, A., Pemas, S., Giakoumoglou, N. and Tzovaras, D., “Low-Cost Hyperspectral Imaging Device for Portable Remote Sensing,” *Instruments* **7**(4), 32 (2023).
- [5] Leon, R., Fabelo, H., Ortega, S., Cruz-Guerrero, I. A., Campos-Delgado, D. U., Szolna, A., Piñeiro, J. F., Espino, C., O’Shanahan, A. J., Hernandez, M., Carrera, D., Bisshopp, S., Sosa, C., Balea-Fernandez, F. J., Morera, J., Clavo, B. and

- Callico, G. M., "Hyperspectral imaging benchmark based on machine learning for intraoperative brain tumour detection," *NPJ Precis Oncol* **7**(1), 119 (2023).
- [6] Studier-Fischer, A., Seidlitz, S., Sellner, J., Özdemir, B., Wiesenfarth, M., Ayala, L., Odenthal, J., Knödler, S., Kowalewski, K. F., Haney, C. M., Camplisson, I., Dietrich, M., Schmidt, K., Salg, G. A., Kenngott, H. G., Adler, T. J., Schreck, N., Kopp-Schneider, A., Maier-Hein, K., et al., "Spectral organ fingerprints for machine learning-based intraoperative tissue classification with hyperspectral imaging in a porcine model," *Sci Rep* **12**(1), 11028 (2022).
- [7] Hizem, M., Bousbia, L., Ben Dhiab, Y., Aoueilyne, M. O.-E. and Bouallegue, R., "Reliable ECG Anomaly Detection on Edge Devices for Internet of Medical Things Applications," *Sensors* **25**(8), 2496 (2025).
- [8] Raspberry Pi Team., "Raspberry Pi 5," January 2025, <https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf>.
- [9] He, K., Zhang, X., Ren, S. and Sun, J., "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778, IEEE, Las Vegas, NV, USA (2016).
- [10] Wang, X. and Jia, W., "Optimizing Edge AI: A Comprehensive Survey on Data, Model, and System Strategies," *arXiv:2501.03265* (2025).
- [11] Kim, H. E., Cosa-Linan, A., Santhanam, N., Jannesari, M., Maros, M. E. and Ganslandt, T., "Transfer learning for medical image classification: a literature review," *BMC Med Imaging* **22**(1), 69 (2022).
- [12] PyTorch Team., "ResNet-18," 2017, <https://docs.pytorch.org/vision/main/models/generated/torchvision.models.resnet18.html>.
- [13] Loshchilov, I. and Hutter, F., "Decoupled Weight Decay Regularization," *arXiv:1711.05101* (2019).
- [14] Krishnamoorthi, R., Reed, J., Ni, M., Gottbrath, C. and Weidman, S., "Introduction to Quantization on PyTorch – PyTorch."
- [15] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H. and Kalenichenko, D., "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2704–2713, IEEE, Salt Lake City, UT (2018).
- [16] Hanhirova, J., Kämäräinen, T., Seppälä, S., Siekkinen, M., Hirvisalo, V. and Ylä-Jääski, A., "Latency and throughput characterization of convolutional neural networks for mobile computer vision," *Proceedings of the 9th ACM Multimedia Systems Conference*, 204–215, Association for Computing Machinery, New York, NY, USA (2018).
- [17] Rainio, O., Teuhio, J. and Klén, R., "Evaluation metrics and statistical tests for machine learning," *Sci Rep* **14**(1), 6086 (2024).
- [18] Langeron, O., Bourgain, J.-L., Francon, D., Amour, J., Baillard, C., Bouroche, G., Chollet Rivier, M., Lenfant, F., Plaud, B., Schoettker, P., Fletcher, D., Velly, L. and Nouette-Gaulain, K., "Difficult intubation and extubation in adult anaesthesia," *Anaesthesia Critical Care & Pain Medicine* **37**(6), 639–651 (2018).
- [19] An, S., Shin, J. and Kim, J., "Quantization-Aware Training With Dynamic and Static Pruning," *IEEE Access* **13**, 57476–57484 (2025).
- [20] Hinton, G., Vinyals, O. and Dean, J., "Distilling the Knowledge in a Neural Network," *arXiv:1503.02531* (2015).
- [21] Mansourian, A. M., Ahmadi, R., Ghafouri, M., Babaei, A. M., Golezani, E. B., Ghamchi, Z. Y., Ramezani, V., Taherian, A., Dinashi, K., Miri, A. and Kasaei, S., "A Comprehensive Survey on Knowledge Distillation," *arXiv:2503.12067* (2025).
- [22] Wang, W., Iii, R. E. H., Fan, J., Miller, D. A. and Wax, A., "Real-time processing of high-throughput quantitative phase microscopy data using a Jetson Orin Nano," *BIOS* **3**(1), 012902 (2025).
- [23] Marafioti, A., Zohar, O., Farré, M., Noyan, M., Bakouch, E., Cuenca, P., Zakka, C., Allal, L. B., Lozhkov, A., Tazi, N., Srivastav, V., Lochner, J., Larcher, H., Morlon, M., Tunstall, L., Werra, L. von and Wolf, T., "SmolVLM: Redefining small and efficient multimodal models," *arXiv:2504.05299* (2025).
- [24] Steiner, A., Pinto, A. S., Tschannen, M., Keysers, D., Wang, X., Bitton, Y., Gritsenko, A., Minderer, M., Sherbondy, A., Long, S., Qin, S., Ingle, R., Bugliarello, E., Kazemzadeh, S., Mesnard, T., Alabdulmohsin, I., Beyer, L. and Zhai, X., "PaliGemma 2: A Family of Versatile VLMs for Transfer," *arXiv:2412.03555* (2024).
- [25] Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., et al., "Qwen3 Technical Report," *arXiv:2505.09388* (2025).
- [26] Lai, L., Suda, N. and Chandra, V., "CMSIS-NN: Efficient Neural Network Kernels for Arm Cortex-M CPUs," *arXiv:1801.06601* (2018).
- [27] Chen, T., Moreau, T., Zheng, L., Cowan, M., Ceze, L., Guestrin, C. and Krishnamurthy, A., "TVM: An Automated End-to-End Optimizing Compiler for Deep Learning."
- [28] David, R., Duke, J., Jain, A., Reddi, V. J., Jeffries, N., Li, J., Kreeger, N., Nappier, I., Natraj, M., Regev, S., Rhodes, R., Wang, T. and Warden, P., "TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems."
- [29] Seshadri, K., Akin, B., Laudon, J., Narayanaswami, R. and Yazdanbakhsh, A., "An Evaluation of Edge TPU Accelerators for Convolutional Neural Networks," *arXiv:2102.10423* (2022).