Journal Name

Research Paper

Real-Time Endoscopic Image Semantic Segmentation for Efficient Polyp Identification

M. Al-Asli

Computer Engineering Department, College of Computer Science and Engineering, Taibah University, Medina, Saudi Arabia

Article history
Received: 1 April 2025
Revised: 17 June 2025
Accepted: 28 June 2025

*Corresponding Author:
Mohammed Al-Asli,
Taibah University, Medina,
Saudi Arabia
Email: masali@taibahu.edu.sa

Abstract: The advancement of endoscopic procedures has significantly enhanced diagnostic capabilities in polyp identification and removal, which is critical for colorectal cancer prevention. However, the success of these procedures hinges on the accuracy and speed of real-time image analysis. Traditional image processing methods often fail to provide timely and precise segmentation of endoscopic images. This paper presents the implementation of endoscopic image semantic segmentation using U-Net-like convolutional neural networks (CNNs) and depicts how it can be implemented efficiently on field-programmable gate arrays (FPGAs). The proposed methodology involves a structured pipeline for model development, data preparation, and deployment on FPGA hardware to ensure optimal resource utilization and real-time performance. Experimental results demonstrate that our CNN model achieves a Dice coefficient of 0.92 and an Intersection over Union (IoU) score of 0.85. Furthermore, the real-time FPGA implementation achieved an average inference time of 0.007 seconds per image and a speedup of approximately 2.5× compared to NVIDIA RTX 3070 Ti GPU implementation. Additionally, the FPGA solution consumes approximately 31× less power than the GPU.

Keywords: Real-Time Segmentation; Endoscopic Image Analysis; Polyp Detection; Convolutional Neural Networks; FPGA Acceleration

Introduction

Endoscopic imaging systems are sophisticated medical devices used to visualize the interior of the gastrointestinal tract and other hollow organs (Lee & Kim, 2019; Kumar & Mohan, 2020; Klein & Gokhale, 2018). Clinicians face several challenges during real-time procedures, including limited visibility due to bowel contents, variable lighting, and motion artifacts. One critical issue is the delay in image analysis, which can lead to missed polyps, especially small or flat ones that are easily overlooked during fast-paced examinations. These delays may reduce diagnostic accuracy and increase the likelihood of interval cancers, where polyps go undetected and progress between screenings. Real-time, automated polyp detection and segmentation can assist clinicians by highlighting suspicious regions instantly, improving detection rates, reducing cognitive load, and ultimately enhancing patient outcomes.

Endoscopic imaging systems consist of a flexible tube equipped with a light source and a miniature camera, allowing healthcare professionals to capture live images of internal structures. The primary purpose of endoscopy is to diagnose and treat various conditions, such as identifying and removing polyps, which are precursors to colorectal cancer. During an endoscopic procedure, the physician inserts the endoscope through natural openings and obtains direct visualization of the esophagus, stomach, intestines, and other parts. The high-resolution images obtained can reveal abnormalities such as inflammation, tumors, or bleeding, facilitating timely interventions. Traditionally, segmentation has been performed using conventional image processing techniques, which often struggle to provide the necessary speed and precision required in real-time examinations (Schoenfeld & McCarthy, 2021; Zhang & Xu, 2020).

Recently, machine learning (ML) has been increasingly utilized in endoscopic imaging to enhance diagnostic accuracy and efficiency. Convolutional neural networks (CNNs) have shown promising results in semantic segmentation in various medical applications and are expected to enable more precise identification of anatomical structures and abnormalities within endoscopic images. Real-time analysis in endoscopic imaging is important for several reasons. First, it allows clinicians to identify abnormalities and perform immediate interventions such as polyp removal or biopsies, which can significantly improve patient outcomes (Esteva et al., 2017; Liu & Zhang, 2020). Second, real-time segmentation enhances accuracy by providing immediate visual feedback. Third, quicker analysis leads to shorter procedure times, which improves patient comfort and reduces anxiety. Real-time capabilities also ensure adaptability to dynamic environments, providing clinicians with the most current information during procedures. Finally, real-time segmentation facilitates more accurate polyp classification in endoscopic images. Classification tasks benefit from this approach because the segmentation process outputs binary masks, which allow machine learning algorithms to focus on specific features without background interference (Kumar & Gupta, 2021; Zhang & Wang, 2020; Gao & Zhang, 2020).

Field-programmable gate arrays (FPGAs) represent a powerful hardware solution for implementing these advanced machine learning models. FPGAs are integrated circuits that can be configured in the field after manufacturing. They can provide efficient parallel processing tailored to specific applications. This programming flexibility makes them particularly well-suited for real-time image processing tasks where speed and efficiency are paramount. Utilizing FPGAs for polyp image semantic segmentation offers several advantages. Their parallel processing capabilities can significantly reduce inference times compared to traditional CPU or GPU implementations. This is essential in medical environments where rapid analysis can lead to timely clinical decisions. Additionally, FPGAs have no operating systems or specialized software requirements, and they typically consume less power than GPUs, making them ideal for portable and embedded medical devices that require continuous operation without frequent recharging.

However, programming FPGAs is not a straightforward task and requires experience in hardware design. Recently, great progress has been made in transforming code written in high-level languages to hardware designs. One such advancement is the development of the PYNQ framework (Python Productivity for Zynq), in which FPGAs like the PYNQ-Z2 can be used within Jupyter Notebook to run CNN models on their programmable logic fabric. This research aims to enhance this capability further and targets endoscopic imaging systems by integrating a U-Net-like CNN model with FPGAs for semantic segmentation tasks. The combination of the segmentation model with the computational efficiency of FPGAs not only improves diagnostic accuracy but also optimizes resource utilization. While real-time image analysis is critical during endoscopic procedures, its successful implementation is heavily dependent on the choice of hardware. Compared to traditional platforms such as GPUs, FPGAs offer several distinct advantages.

- Unlike GPUs, which can consume upwards of 50–250W during inference, FPGAs such as the PYNQ-Z2 used in this study consume as little as 1.6W.
- FPGAs operate using parallel hardware logic without reliance on thread scheduling or dynamic memory access, resulting in predictable and consistently low latency.
- FPGAs allow for application-specific pipeline design, where only necessary operations are implemented in hardware.
- FPGA SoCs such as PYNQ-Z2 offer tight integration of compute, control, and communication, enabling real-time feedback within clinical tools without external hardware.

The contributions of this work are as follows:

- Developing an ML framework for polyp image segmentation on FPGAs
- Developing a UNET-like CNN model suitable for deployment on FPGAs
- Utilizing Tensil AI compiler to convert the polyp segmentation model to a Hardware design
- Analyzing the performance of the deployed polyp segmentation model and reporting the inference time, power consumption and resource utilization.

Related Works

The field of polyp detection and segmentation using deep learning on embedded and FPGA-based platforms has seen significant advancements. Various studies have explored different architectures, implementation techniques, and computational optimizations to improve accuracy and efficiency in real-time medical imaging. In this section, we summarize key contributions from existing literature and compare them to our FPGA-based approach.

Deep Learning for Polyp Detection and Segmentation

Wireless Capsule Endoscopy (WCE) is a diagnostic tool used to visualize the gastrointestinal tract, particularly the small intestine, which is often difficult to access with traditional endoscopic methods. This non-invasive technique involves the ingestion of a small, pill-sized camera that captures high-resolution images of the gastrointestinal tract as it travels through. One notable study proposed a lightweight deep neural network specifically designed for WCE. This approach focuses on local processing within the capsule, ensuring on-board detection without the need for external processing. This method targets low-power miniaturized devices, whereas our FPGA-based segmentation system is designed for real-time endoscopic procedures with greater computational capability (Lee et al., 2021; Gomes et al., 2020; Jha et al., 2021; Varam et al., 2023).

Additionally, Wang et al. (2022) present a deep learning model, GastroNet, for feature detection and classification of gastrointestinal abnormalities. While this study focuses on multi-class classification, our research emphasizes real-time segmentation on FPGAs for more fine-grained detection and localization of polyps. Yang & Yu (2021) provided a comprehensive review of artificial convolutional neural networks in the context of object detection and semantic segmentation for medical imaging. Their work outlines the evolution of CNN architectures and emphasizes their utility in various diagnostic tasks, including gastrointestinal image analysis. Drawing from the foundational insights presented in their study, our work addresses one of the key limitations highlighted: the lack of real-time, power-efficient deployment strategies suitable for clinical use. By proposing an FPGA-based implementation of a U-Net-like segmentation model, we advance this area through a lightweight and hardware-accelerated framework tailored for embedded medical systems.

Jafar et al. (2024) explore the development of advanced machine learning techniques aimed at enhancing the detection and classification of polyps during endoscopic procedures. This study focuses on creating a semantic segmentation network that differentiates between polyps and surgical instruments within the complex visual environment of endoscopic images. They achieved a Dice score of 0.9621 using the Computer Vision Center Clinic Database (CVC-ClinicDB). However, their scope differs from that of our work.

Selvaraj & Jayanthy (2023) evaluated different convolutional neural network (CNN) architectures and optimization methods for segmenting polyps in wireless capsule endoscopy images. Their findings show that certain CNN architectures, particularly U-Net and DeepLab, significantly outperformed others in terms of segmentation accuracy and processing speed, achieving high Dice coefficients that indicate precise delineation of polyps from surrounding tissues. Furthermore, the paper shows that the integration of optimization techniques improved both convergence rates and model accuracy. The results indicate that semantic segmentation of polyps using CNNs can facilitate earlier detection and intervention in colorectal cancer.

Noor et al. (2023) focused on developing an explainable AI framework to enhance the localization and classification of various gastrointestinal disorders based on endoscopic images. Their results demonstrated that the proposed model achieved high accuracy rates in identifying conditions such as ulcers, polyps, and tumors. The use of explainable AI techniques allowed clinicians to interpret model predictions effectively, providing insights into the decision-making process behind classifications. The study reported high precision, recall, and F1 scores, indicating the model's reliability.

Hardware Implementations for Medical Imaging

Asha et al. (2024) presented a novel approach to implementing deep learning models on field-programmable gate arrays (FPGAs) for the analysis of colorectal tumor images. Their results indicated that the FPGA-based implementation significantly enhanced processing speed compared to traditional CPU and GPU methods, achieving real-time analysis capabilities. The study reported high accuracy rates in tumor detection and classification, demonstrating the effectiveness of the deep learning models employed. Additionally, the authors showcased the benefits of reduced power consumption and resource utilization in FPGA implementations, making it a viable option for clinical settings.

A CNN-based polyp detection model implemented on an embedded device highlights hardware efficiency and real-time processing. However, it is limited to detection, whereas our FPGA-based approach enables segmentation, providing more detailed localization of polyps within endoscopic images (Liu et al., 2020). Similarly, another relevant paper discusses integrating Hough Transform processing into a wireless capsule endoscopy system for efficient real-time detection of abnormalities. This work focuses on classical computer vision techniques, while our approach leverages deep learning for segmentation (Zhang et al., 2021).

The study on embedded detection of polyps in WCE images emphasizes early-stage detection using compact and power-efficient hardware. Our work differs by targeting real-time FPGA segmentation for live endoscopic procedures, where accurate boundary delineation is crucial for clinical decision-making (Inoue et al., 2021). DLA-E presents a deep learning accelerator designed for the classification of endoscopic images.

While this study highlights hardware acceleration techniques for deep learning models on embedded platforms, our work focuses on segmentation, which is essential for surgical planning and AI-assisted diagnosis (Chen et al., 2021).

A different approach employs YOLOv8 for polyp detection and introduces YOLO-Score Metrics to evaluate the suitability of detected polyps (Chen et al., 2022). This study emphasizes real-time inference and accuracy enhancement. Compared to our work, which focuses on segmentation using FPGA-based acceleration, this study prioritizes detection performance on standard GPUs. Our approach enables hardware-efficient real-time segmentation, making it more suitable for embedded clinical applications.

FPGA-Based Approaches

Implementation on customizable hardware for colorectal tumor classification with endoscopic video also present valuable insights. This research targets real-time video-based diagnostics, where customizable digital signal processing (DSP) cores are beneficial for low-power embedded systems. Compared to our FPGA-based segmentation approach, this study focuses on classification rather than segmentation. Additionally, FPGAs offer greater parallel processing capabilities than DSP-based architectures for deep learning applications (Zhang et al., 2021).

A comparative analysis of the Narrow U-Net architecture across GPU, CPU, and FPGA platforms evaluates accuracy, speed, and power efficiency. While this work provides a comparative analysis and targets implementing an ASIC for the ML model, our study specifically optimizes an FPGA implementation for real-time segmentation, showcasing its power efficiency and inference speed compared to GPUs (Alzubaidi et al., 2020). Furthermore, an FPGA implementation of support vector machine (SVM) classification for colorectal endoscopic images emphasizes traditional machine learning rather than deep learning. While SVMs are computationally less expensive, our work leverages deep neural networks for superior segmentation accuracy, making it more applicable to complex real-time endoscopic imaging tasks (Nasir et al., 2021).

Our Contribution

While previous studies such as Alzubaidi et al. (2020) and Nasir et al. (2021) have explored FPGA implementations of CNN models for medical imaging, our work contributes a distinct advancement by focusing on a fully deployable, low-cost FPGA solution (PYNQ-Z2) optimized for real-time polyp segmentation. Unlike prior work that often targets classification or detection, we implement a complete semantic segmentation pipeline tailored for FPGA constraints. Furthermore, we utilize the Tensil AI compiler framework which enables efficient model compression and quantization for edge deployment, facilitating the integration of our CNN model onto FPGA hardware.

Additionally, our system demonstrates low power consumption, operating at approximately 1.6W, which is significantly lower than the power requirements of GPU-based solutions. This integration, combined with our focus on power efficiency and edge deployment feasibility, distinguishes our contribution within the growing field of FPGA-based medical imaging.

Materials

Hardware Platform

The PYNQ-Z2 board is a System on Chip (SoC), an integrated circuit that combines all the essential components of a computer system into a single chip. It mainly consists of a Processing System (PS) and Programmable Logic (PL), also called FPGA fabric. The PYNQ-Z2 board's PS consists of an ARM Cortex-A9 dual-core CPU running Linux, facilitating communication between software and the FPGA fabric. The PL consists of programmable FPGA logic elements that execute custom user logic. In our case, we use the PL to implement the CNN model.

An advantage of the SoC architecture is the direct connection between the PS and PL, allowing users to extend the PS's functionality into the PL. For example, the user might use standard Python libraries on the PS and then leverage the PL to implement a custom image processing function, such as edge detection, to accelerate it. This custom function can be loaded from the PS to the PL dynamically, similar to a standard software library.

To configure the system, an image containing the PYNQ framework is flashed to an SD card for the PS to boot from. The user then accesses the PYNQ Linux environment and loads needed software packages. For the PL, a bitstream file (hardware configuration binary) along with the CNN model are required for programming. The PYNQ framework image is obtained from the official PYNQ.io website, and the FPGA fabric files are developed using the Tensil AI compiler and the Xilinx Vivado toolchain.

Figure 1 illustrates the basic components of the PYNQ-Z2 board. During inference, the PS acts as the control unit, managing data flow and interfacing with external systems such as storage, Ethernet, and Jupyter Notebook. The PL is configured to execute the CNN model in hardware. When an endoscopic image is loaded into memory via the PS, it is transferred to the PL for acceleration. The PL processes the image and returns the resulting segmentation mask to the PS, which then performs postprocessing and visualization tasks, such as thresholding or displaying the output mask in real time.

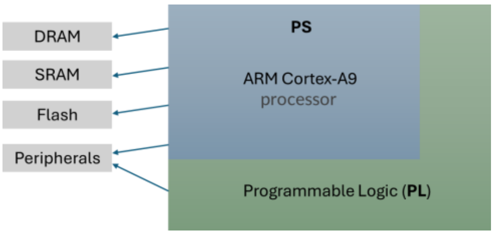


Fig. 1. Basic components of the PYNQ-Z2 board

Methods

Our proposed Framework for real-time endoscopic image segmentation consists of essential phases as shown in Fig. 2. It begins with model development phase using Python, where Python libraries like TensorFlow are used to develop the CNN model. This phase mainly consists of data preparation and model training steps. The data preparation involves reading the dataset, image normalization, which uniformly scales pixel values to ensure numerical stability during the training phase. Additionally, it involves augmentation techniques such as flipping, rotation, contrast adjustments, and noise addition. The model training phases involves defining the model structure and layers and tuning its hyperparameters, performing the actual model training, and evaluating the performance of the model using unseen images.

Once the model is trained and evaluated, it is saved as either TensorFlow Frozen Graph (pb) or Open Neural Network Exchange (onnx) file. The saved model is then used as an input to the Tensil AI compiler to prepare the PYNQ-Z2 FPGA HW files. The Tensil AI compiler basically performs two main tasks; the first task is converting the CNN model to a format representing the CNN model which can be loaded to the PYNQ-Z2 FPGA and the second task is generating the PYNQ-Z2 Verilog RTL (Register Transfer Level) file which generates using Xilinx Vivado software the bitstream (.bit) file to program the FPGA. The bitstream file generated basically creates an overlay in which the files generated in the first task can be loaded using Jupyter notebook to the FPGA using an ethernet cable. As illustrated in Fig. 2, the RTL is used to prepare the PYNQ-Z2 FPGA and then the model files (.tdata, .tmodel and .tprog) are used to deploy the CNN model. The tmodel (Tensil Model file) stores the ML model structure, the tprog (Tensil Program file) contains execution instructions for the model and the tdata (Tensil Data file) holds the weights and parameters of the trained model. To enable efficient deployment of the CNN model on the PYNQ-Z2 FPGA, we applied several hardware-specific optimizations, with a particular focus on quantization. Using the Tensil AI compiler, the trained floating-point model was quantized to 8-bit fixed-point precision, which is more suitable for FPGA logic and significantly reduces memory and computational demands. Quantization was applied in the post-training where weights, activations, and biases were converted from 32-bit floating point to 8-bit integer representations. This optimization dramatically lowered the model's bitstream size, allowed for faster inference, and reduced BRAM and DSP usage.

Besides programming the PYNQ-Z2 FPGA logic (the PL part) using the RTL file, further preparations are needed, as mentioned earlier, for the PYNQ-Z2 PS part. First the PYNQ-Z2 image which is a Linux OS is burned into an SD card and is inserted to the PYNQ-z2 board to boot from it. Second, necessary packages such as Core PYNQ, Python Essentials, Model Deployment, and Jupyter Notebook packages are installed.

After preparing the PYNQ-Z2 board and ensuring it is running as expected, the model then can be deployed (loaded) to the FPGA fabric. First, the bitstream file containing basic HW components for receiving the CNN model file and communicating with the PS is loaded using Jupyter notebook. Second, the CNN model files(.tdata, .tmodel and .tprog) are loaded to the FPGA fabric.

Finally, after the FPGA fabric is prepared and the CNN model is loaded and is up running inside the PYNQ-Z2 board, the inference step can be accomplished. First an input endoscopic image stored on the PS RAM using Jupyter notebook is fed to the FPGA fabric. Second the FPGA receives the input image and run the inference (prediction). Third, the PS receives the output activations and performs the necessary postprocessing to create the output binary mask. Finally, the outputted mask is stored and shown on the Jupyter notebook for further inspection and testing.

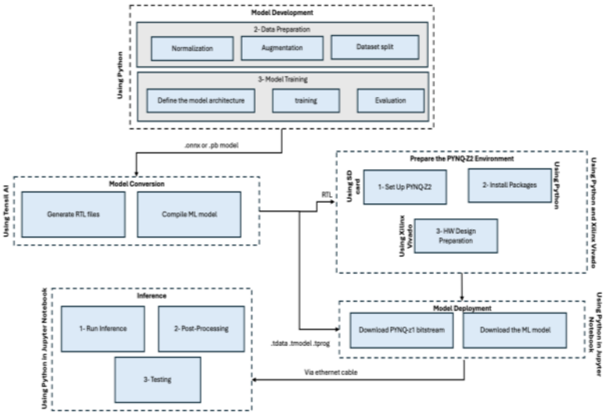


Fig. 2. Our proposed PYNQ-Z2 based accelerator framework

Segmentation CNN Model

The segmentation CNN model used for real-time endoscopic image follows a UNET like structure which is characterized by a feature extraction phase consisting of convolutional layers followed by ReLU activations and max pooling operations. It is worthwhile to mention that a CNN model needs to be developed for this segmentation task as currently models like UNET are not fully implemented using the PYNQ framework. The model incorporates fully connected layers for final segmentation and outputs categorical predictions based on the extracted features. This CNN-based segmentation model resembles a U-Net architecture but without the skip connections. The model adopts a bottleneck structure which consists of an encoder that employs convolutional layers, ReLU activations, and max pooling for feature extraction followed by a decoder that utilizes transpose convolutions to restore the original image size. Despite the absence of skip connections, this architecture maintains key similarities to U-Net including the use of convolution and max pooling layers in the encoder, a bottleneck layer at the deepest part of the network, and a decoder that employs transpose convolutions for up sampling. Hence, it is effective for segmenting the input images. The complete model consists of 7,065,250 parameters, with 7,064,290 being trainable. The architecture is summarized in Fig. 3.

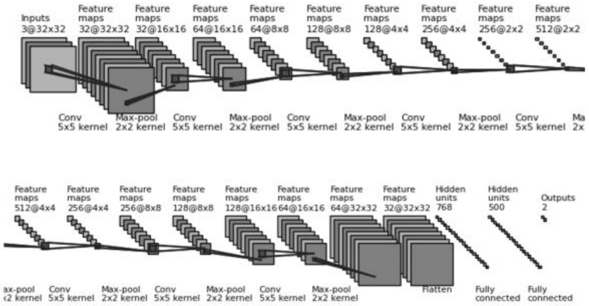


Fig. 3. Our developed CNN UNET like model structure

After preparing the CNN model, it can be saved in either the onnx or pb format. Both onnx and pb files can be compiled using Tensil AI to generate the necessary Hardware design files to deploy the model in the PYNQ FPGA. The process in which Tensil AI generates these files is shown in fig. 4. The first step in the Tensil AI workflow involves selecting an appropriate architecture defined by a tarch file. This file specifies the FPGA's hardware resources and the configuration required for tensor operations (PYNQ Z2 in our case). Once the architecture is determined, the RTL files are generated using the Tensil tool, which translates high-level tensor operations into low-level hardware descriptions. The RTL files (.v Verilog file) are then synthesized using Xilinx Vivado (shown in Fig. 5) to produce the necessary FPGA configuration files: the bit file, which programs the FPGA, and the hwh file, which describes the hardware interfaces required for communication with software applications.

Following the RTL generation, the ML model must be compiled to run efficiently on the FPGA. The compilation process converts the trained model into hardware-executable components, including the tmodel, tprog, and tdata files. These files enable the model to be executed on the FPGA while leveraging hardware acceleration for improved performance.

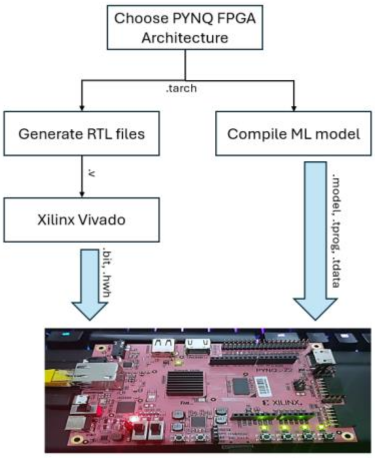


Fig. 4. The PYNQ-Z2 files generation and programming flow using Tensil AI

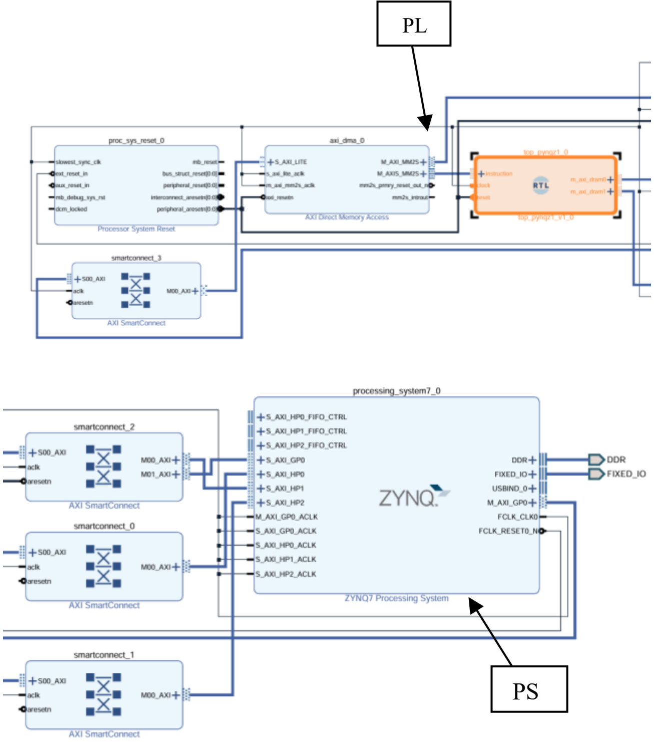


Fig. 5. The HW design of our Framework

The integration of CNN model inference with FPGA-based platforms like the PYNQ-Z2 offers significant advantages. It improves computational efficiency, reduces power consumption, and adds real-time processing capabilities. The use of Jupyter Notebook further simplifies model deployment, making it accessible to researchers and developers without requiring extensive knowledge of FPGA design. This approach demonstrates the potential of FPGA-accelerated ML inference for applications in edge computing, embedded systems, and real-time AI-driven solutions.

Results and Discussion

The performance of the real-time endoscopic image segmentation system was evaluated using both software (SW) and hardware (HW) implementations. The dataset used on this study consists of 612 images and mask. To ensure model robustness and clinical applicability, we carefully examined the dataset's diversity and representativeness, summarized in Table 1. The images vary in resolution and cover different types of polyps, ranging from small sessile to larger pedunculated types. Furthermore, the dataset includes frames with varying bowel preparation quality, lighting conditions (e.g., shadowing, specular highlights), and perspectives within the gastrointestinal tract, simulating real-world variability. The experimental setup included a deep learning model trained on a high-performance desktop PC, followed by deployment on an FPGA for real-time inference. Key metrics such as inference time, resource utilization, and segmentation accuracy were analyzed to assess the system's effectiveness in polyp identification.

Table 1. Summary of Polyp Dataset Characteristics

Attribute	Description
Total Number of Images	612
Image Type	RGB colonoscopy frames
Mask Availability	Yes – Binary segmentation masks for polyp regions
Image Resolution	Varies (common sizes: 384×288, 576×768, resized to 256×256 for training)
Polyp Types	Includes various shapes/sizes: sessile, pedunculated, flat
Scene Variability	Diverse lighting, specular highlights, shadows, and bowel preparation levels
Device or Scope Info	Not explicitly provided
Patient Demographics	Not available

Software-Based Inference Performance

The CNN model was trained on a desktop workstation with specifications as shown in Table 2. The model performance is evaluated using the Dice coefficient and Intersection over Union (IoU) metrics. The Dice coefficient measures the overlap between predicted and ground truth segmentation masks, defined as in Eq. (1), where A and B are the predicted and actual mask regions, respectively. The IoU, also known as the Jaccard index, quantifies the ratio of intersection to union between the predicted and ground truth masks, formulated as in Eq. (2). The U-Net model operates on input images of 32 × 32 × 3 pixels. The model is optimized using Adam (learning rate = 0.001) and trained with Sparse Categorical Crossentropy (SCCE) loss. The dataset is split into 70% for training and 30% for validation and testing, using stratified random sampling to ensure balanced representation of different polyp sizes and shapes. This split helps evaluate the model's generalization ability while preventing overfitting. The validation set is used to monitor performance during training, with early stopping and model checkpointing based on validation loss minimization to select the best-performing model. The model is trained for 210 epochs with a batch size of 8. The number of epochs is chosen to ensure sufficient learning while preventing overfitting, and early stopping is used to halt training if validation loss stops improving. The batch size of 8 balances memory efficiency and stable gradient updates during optimization.

$$Dice = \frac{(2 \times |A \cap B|)}{(|A| + |B|)} \quad (1)$$

$$IoU = \frac{|A \cap B|}{|A \cup B|} \quad (2)$$

Table 2. Desktop workstation specifications used in the study

Processor	Intel Core i9-11980K CPU @ 3.30GH
Memory	32GB DDR4 RAM
Graphics Card	NVIDIA GeForce RTX 3070 Ti
Software Environment	CUDA 11.8.522 cuDNN 11.2.0 Python 3.9.0 TensorFlow 2.1.0 Keras 2.3.1 Windows 11 (64-bit)

The Albumentations library was used to apply various augmentations, which can improve model robustness and generalization. The pipeline includes a horizontal flip with an 80% probability, allowing for symmetrical variations that help the model learn invariant features. Similarly, a vertical flip is incorporated, also with an 80% chance, further increasing data diversity. The rotation transformation allows images to be rotated by up to 45 degrees, introducing angular variations essential for tasks where object orientation may vary. Additionally, adjustments to brightness and contrast are applied with a 20% probability, ensuring that the model can handle different lighting conditions. Lastly, Gaussian blur is included, also with a 20% chance, to simulate focus variations and enhance the model's ability to recognize objects in less-than-perfect conditions. The dataset is systematically split into training (70%), validation (20%), and test (10%) sets, optimizing the learning process, hyperparameter tuning, and final performance assessment. We have tuned our model and studied the effect of its parameter to gain the most dice and IoU possible. Please refer to Table 2 for the details of the effect of each tuned component. The software-based inference served as a baseline for evaluating the FPGA implementation.

The training performance of the model is illustrated in Fig. 6, which presents the loss and accuracy curves over the training epochs. The left plot compares the training loss and validation loss. Both curves exhibit a downward trend which indicates effective learning. The validation loss closely follows the training loss which suggests minimal overfitting. The right plot shows the accuracy comparison. Both training and validation accuracy steadily increase and converges to about 98%. Please do note that the loss and accuracy are not a perfect measure for the segmentation task and they do not reflect the actual dice and IoU, although they can give meaningful patterns for knowing the learning process of the model. The close alignment between training and validation accuracy further confirms the model's generalization capability.

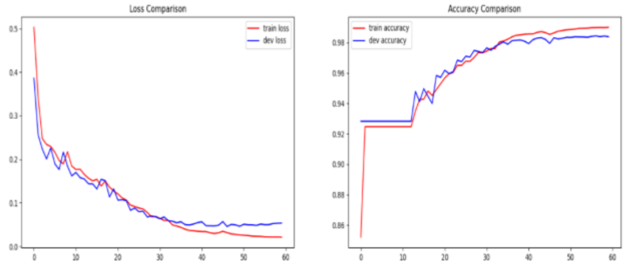
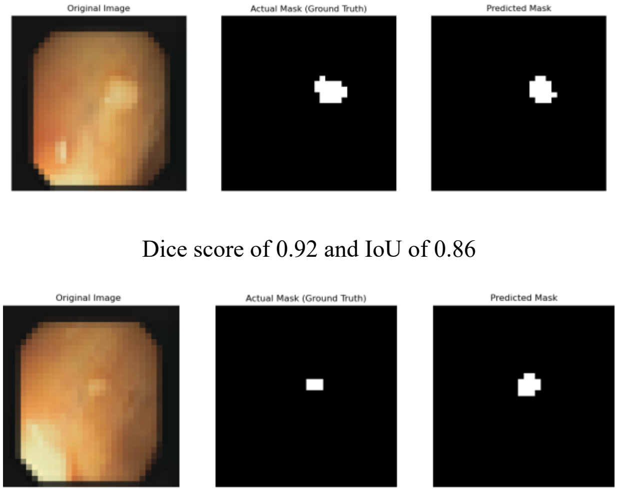


Fig. 6. Loss and accuracy for training and validation sets

Fig. 7 presents a sample of two images from the test dataset. The figure shows the images (32 x 32) along with its corresponding actual mask and predicted mask. The predicted mask is generally close to the actual mask especially the location of the white colors. Though it is not fully equal to the actual mask, it can be benefitable for many cases such as detecting the polyp from the predicted mask. The classification can be done easily as if there is no white color that indicates non-existence of the polyp in the image.



Dice score of 0.92 and IoU of 0.86

Dice score of 0.89 and IoU of 0.83

Fig. 7. Samples of the polyp image segmentations along with their actual and predicted masks, Dice score and IoU.

To assess the robustness of each model variant, we repeated each experiment five times using different random seeds and report the average Dice and IoU scores along with standard deviations. These variations confirm the consistency of improvements introduced at each stage. Table 3 summarizes the results of the ablation study, showing the impact of each model enhancement on segmentation performance in terms of Dice score and IoU, with results reported as mean ± standard deviation over multiple runs.

Table 3. Ablation study of our model

Experiment No.	Model Variation	Dice Score (%)	IoU (%)
1	Baseline model	82.00 ± 1.2	75.00 ± 1.3
2	+ Data Augmentation	85.10 ± 1.1	78.20 ± 1.2
3	+ Additional Conv Layer	87.20 ± 1.0	80.00 ± 0.9
4	+ Batch Normalization	88.40 ± 0.8	81.00 ± 1.1
5	+ Optimized Learning Rate	89.30 ± 0.7	82.00 ± 0.8
6	Full Model (All Enhancements)	91.16 ± 0.6	84.50 ± 0.7

Hardware-Based Inference Performance

For real-time execution, the trained model was deployed on an FPGA-based hardware system. Our testbed used for this is shown in Fig. 8 and a snippet of the Jupyter notebook used for the FPGA-based inference is shown in Fig. 9. The FPGA implementation was tested on 26 endoscopic images, and the total inference time recorded was 0.163 seconds, resulting in an average inference time of 0.007 seconds per image. This demonstrates the efficiency of FPGA acceleration in handling real-time segmentation tasks.

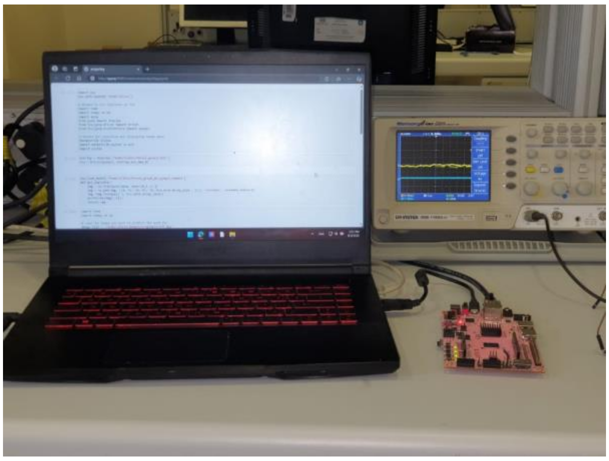


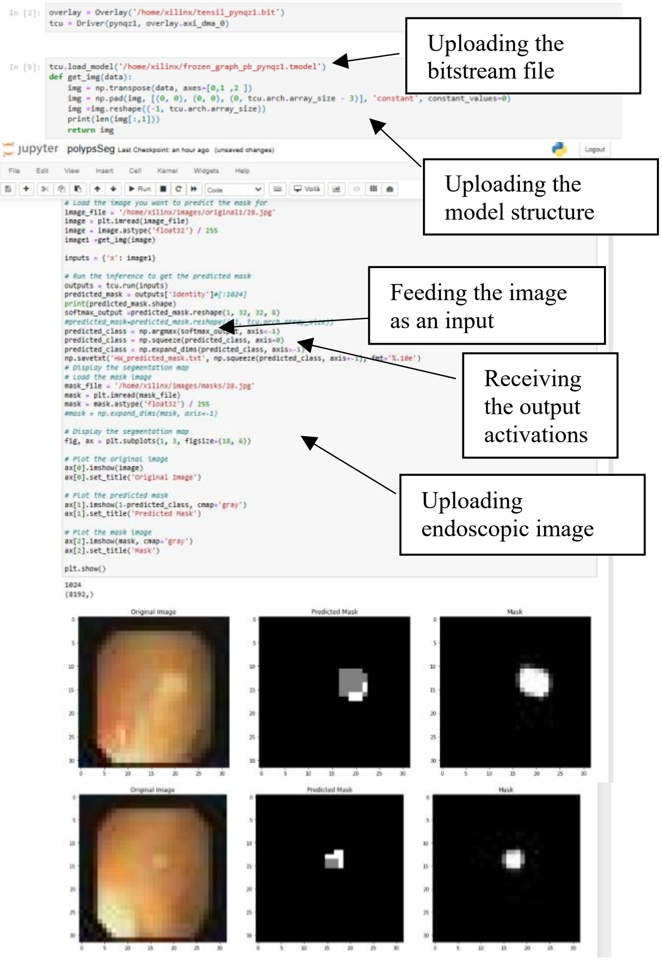
Fig. 8. Testbed used for endoscopic polyp image segmentation

The low inference latency achieved on the FPGA underscores its suitability for real-time medical applications, where rapid and accurate segmentation is crucial for clinical decision-making. The real-time processing speed ensures that endoscopic frames can be segmented and analyzed without perceptible delays, enhancing usability in live procedures.

Resource Utilization and Power Consumption

The FPGA-based inference was analyzed for hardware resource consumption, including logic utilization, Digital Signal Processing (DSP) usage, and memory allocation. The utilization details, such as Look-Up Table (LUT) usage, DSP block utilization, and on-chip memory consumption (shown in Fig. 10), suggest that only at maximum 33% of the FPGA fabric resources (DSP blocks) were consumed by our design. This suggests that our design is suitable for low cost and constrained FPGAs.

Power efficiency is a critical factor in real-time medical image processing, particularly for embedded and portable applications. Fig. 11 depicts the details of power consumption report generated for our design using Xilinx Vivado toolchain. The FPGA-based implementation of the endoscopic image segmentation model demonstrates exceptional power efficiency, consuming only 1.616W in total, with 1.472W attributed to dynamic power usage. Most of the power consumption stems from the PS (86%), while logic, DSP blocks, and BRAM contribute minimally. This low-power consumption makes FPGAs an attractive choice for real-time medical applications, especially where continuous operation is required in resource-constrained environments. In contrast, GPU-based implementations, such as the NVIDIA RTX 3070 Ti used in this study, typically consume up to 290W under full load. However, during our specific software inference experiment, the measured workstation power draw was 49.6W. Our FPGA implementation (1.616W) thus consumed approximately 31x less power than our GPU-based test setup.



Uploading the bitstream file
Uploading the model structure
Feeding the image as an input
Receiving the output activations
Uploading endoscopic image
jupyter polypsSeg Last Checkpoint: an hour ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Run
Code
Voilà
Original Image
Predicted Mask
Mask
0
5
10
15
20
25
30

```
In [2]: overlay = Overlay('/home/xilinx/tensil_pynqz1.bit')
        tcu = Driver(pynqz1, overlay.axi_dma_0)

In [9]: tcu.load_model('/home/xilinx/frozen_graph_pb_pynqz1.tmodel')
        def get_img(data):
            img = np.transpose(data, axes=[0,1 ,2 ])
            img = np.pad(img, [(0, 0), (0, 0), (0, tcu.arch.array_size - 3)], 'constant', constant_values=0)
            img =img.reshape((-1, tcu.arch.array_size))
            print(len(img[:,1]))
            return img
```

```
# Load the image you want to predict the mask for
image_file = '/home/xilinx/images/original1/28.jpg'
image = plt.imread(image_file)
image = image.astype('float32') / 255
image1 =get_img(image)

inputs = {'x': image1}

# Run the inference to get the predicted mask
outputs = tcu.run(inputs)
predicted_mask = outputs['Identity']#[:1024]
print(predicted_mask.shape)
softmax_output =predicted_mask.reshape(1, 32, 32, 8)
#predicted_mask=predicted_mask.reshape((-1, tcu.arch.array_size))
predicted_class = np.argmax(softmax_output, axis=-1)
predicted_class = np.squeeze(predicted_class, axis=0)
predicted_class = np.expand_dims(predicted_class, axis=-1)
np.savetxt('HW_predicted_mask.txt', np.squeeze(predicted_class, axis=-1), fmt='%.10e')
# Display the segmentation map
# Load the mask image
mask_file = '/home/xilinx/images/masks/28.jpg'
mask = plt.imread(mask_file)
mask = mask.astype('float32') / 255
#mask = np.expand_dims(mask, axis=-1)

# Display the segmentation map
fig, ax = plt.subplots(1, 3, figsize=(18, 6))

# Plot the original image
ax[0].imshow(image)
ax[0].set_title('Original Image')

# Plot the predicted mask
ax[1].imshow(1-predicted_class, cmap='gray')
ax[1].set_title('Predicted Mask')

# Plot the mask image
ax[2].imshow(mask, cmap='gray')
ax[2].set_title('Mask')

plt.show()
```

```
1024
(8192,)
```

Fig. 9. An illustration of the Jupyter notebook script used for inference on the PYNQ-Z2 FPGA for polyp segmentation

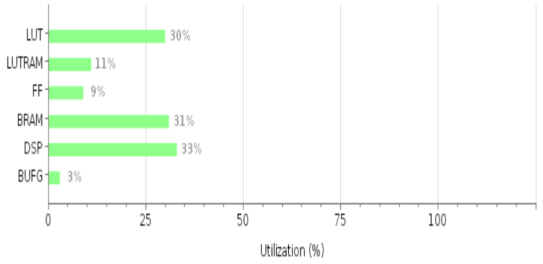
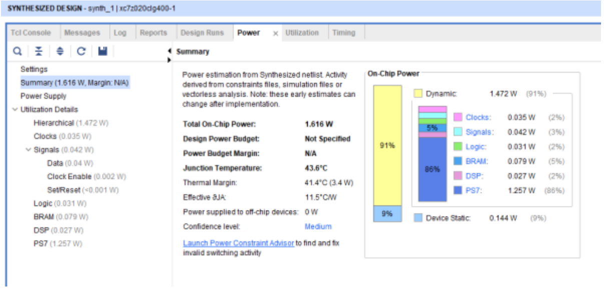


Fig. 10. Resource utilization of the model on the PYNQ-Z2 board



Power estimation from Synthesized netlist. Activity derived from constraints files, simulation files or vectorless analysis. Note: these early estimates can change after implementation.

Total On-Chip Power:	**1.616 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**43.6°C**
Thermal Margin:	41.4°C (3.4 W)
Effective θJA:	11.5°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Medium

Launch Power Constraint Advisor to find and fix invalid switching activity

Dynamic:	1.472 W	(91%)
Clocks:	0.035 W	(2%)
Signals:	0.042 W	(3%)
Logic:	0.031 W	(2%)
BRAM:	0.079 W	(5%)
DSP:	0.027 W	(2%)
PS7:	1.257 W	(86%)
Device Static:	0.144 W	(9%)

Fig. 11. Power consumption details of our model design obtained from Xilinx Vivado toolchain

Comparative Analysis of Software and Hardware Implementations

The results summarized in Table 4 indicate a significant improvement in inference speed for the FPGA-based implementation. While the software implementation leverages high-end GPU acceleration for deep learning computations, the FPGA implementation benefits from its parallel processing architecture which allows for real-time segmentation with minimal latency. Additionally, the FPGA-based approach achieves a slightly lower dice and IoU score compared to the software-based approach. However, it has a dramatic lower inference speed and power consumption which makes it ideal for real-time predictions in constrained devices.

For comparison study, several efficient software-based methods have been proposed by Liao et al., (2021), Wei et al. (2021), Shah et al. (2020) and Chen et al. (2021). While these papers present models with promising segmentation results, their techniques are not currently suitable for FPGA deployment. This suggests that further work should be directed in the future towards supporting more efficient segmentation model such as UNET and pretrained models targeting HW acceleration.

Table 4. Comparison of the SW model using TensorFlow versus the HW model deployed on the PYNQ-Z2 FPGA.

	SW version	**HW version**
Inference time	**0.372**	**0.163**
Average run time per image(s)	**0.016**	**0.007**
Dice	**91.16%**	**90.07%**
IoU	**84.50%**	**83.12%**
Power consumption	**49.6W**	**1.616W**

Table 5 highlights key differences between our Tensil AI-based implementation and the Xilinx Vitis AI approach used in (Alzubaidi et al., 2020). These comparisons span workflow efficiency, technical accessibility, and clinical deployment viability. Notably, our solution demonstrates superior Python integration and quantization automation while achieving significantly better power efficiency which is ~2× lower and faster inference times (~1.7× faster) on more cost-effective hardware. This combination of technical advantages positions our framework as particularly suitable for clinical environments where ease of deployment, energy efficiency, and real-time performance are critical requirements. The trade-off between accessibility (Tensil AI's minimal FPGA expertise requirements) and customization flexibility (Vitis AI's manual configuration options) reflects a distinct design approach for medical computing applications.

Table 5. Comparison of our framework against other frameworks

Feature	**Tensil AI (used in our work)**	**Xilinx Vitis AI (used by Alzubaidi et al. (2020))**
Python Compatibility	Full (Jupyter notebooks)	Limited (C++/HLS focus)
Quantization Automation	8-bit fixed-point (auto)	Manual configuration
FPGA Expertise Needed	Minimal	Significant
Clinical Readiness	Plug-and-play deployment	Custom RTL often required
Power consumption	1.616W	3.2W
Inference time	0.007s	0.012s
FPGA Platform	Low-cost PYNQ-Z2	Mid-range Zynq-7000

Future Work

This study demonstrates the potential of FPGA-based real-time segmentation for endoscopic image analysis, but several avenues remain for future research. First, incorporating larger and more diverse datasets, including external validation from available sources would improve model robustness and generalizability across varied clinical conditions. Second, quantization-aware training (QAT) and pruning techniques could be explored to further optimize model efficiency without compromising accuracy, especially for ultra-low-power FPGA deployments.

While stratified random sampling was used to ensure a balanced split across training, validation, and test sets (70%, 20%, and 10% respectively), we acknowledge the importance of further validating the generalization capacity of the model. In future work, we plan to incorporate k-fold cross-validation (e.g., 5-fold or 10-

fold), which systematically rotates training and validation data partitions to assess model performance more robustly across different data splits. At present, our focus is to develop a lightweight CNN model that can be accelerated and deployed on FPGAs. Currently, the dataset used does not contain metadata for external validation on independent datasets captured from different clinical settings.

Additionally, integrating skip connections and attention mechanisms into the CNN architecture may enhance segmentation precision, particularly for small or flat polyps. From a hardware perspective, extending the implementation to higher-end FPGAs or hybrid SoCs (e.g., Zynq Ultrascale) would allow for real-time processing of higher-resolution frames and more complex models. Finally, collaboration with healthcare providers is essential to enable pilot clinical deployments, where the framework can be tested in live endoscopic procedures to evaluate usability, reliability, and diagnostic impact.

Conclusion

The experimental evaluation confirms that FPGA-based real-time segmentation of endoscopic images provides a significant advantage in terms of inference speed while maintaining high segmentation accuracy. The proposed framework successfully demonstrates the feasibility of using hardware acceleration for deep learning-based medical imaging applications. The results of this study suggest that the framework can effectively distinguish polyp regions from background tissue. The real-time execution capability of FPGA further strengthens its suitability for integration into clinical workflows to provide healthcare professionals with immediate feedback during endoscopic procedures. Future work may focus on further optimizing hardware resource utilization and exploring advanced quantization techniques to enhance efficiency without compromising accuracy. This work demonstrates the feasibility of FPGA-based real-time segmentation for medical imaging, and future efforts should focus on clinical validation. Potential deployments include integration into endoscopic tower systems for live polyp highlighting, portable diagnostic units for low-resource settings, and capsule endoscopy devices, where on-device FPGA processing can reduce transmission and enable faster triage.

Ethics

The authors affirm that no ethical issues are anticipated following the publication of this work. However, the study has certain limitations affecting generalizability. First, the dataset consists of 612 annotated images; while diverse, it may not encompass the full variability of large-scale clinical settings. Second, the hardware implementation was tested on a low-cost FPGA (PYNQ-Z2), which restricted model size, precision, and computational throughput.

Funding Information

This research received no external funding.

References

Alzubaidi, A. R. M., et al. (2020). Comparative analysis of pure convolution narrow U-Net for colorectal polyp segmentation on GPU, CPU, and FPGA. Medical Image Analysis, 65, 101866. https://doi.org/10.1016/j.media.2020.101866

Asha, S., Chandru, R., & Rohit, S. (2024). High-efficiency FPGA implementation of deep learning models for colorectal tumor image analysis. In Proceedings of the 2024 Global Conference on Communications and Information Technologies (GCCIT) (pp. 1–6). IEEE. https://doi.org/10.1109/GCCIT63234.2024.10862443

Chen, J., et al. (2022). Real-time polyp detection from endoscopic images using YOLOv8 with YOLO-Score metrics for enhanced suitability assessment. International Journal of Computer Assisted Radiology and Surgery, 17(12), 2191–2202. https://doi.org/10.1007/s11548-022-02615-3

Chen, Y., et al. (2021). DLA-E: A deep learning accelerator for endoscopic image classification. IEEE Transactions on Neural Networks and Learning Systems, 32(8), 3352–3363. https://doi.org/10.1109/TNNLS.2021.3058999

Chen, Z., Guo, L., & Wang, Z. (2021). Polyp-PVT: Polyp segmentation with pyramid vision transformers. IEEE Access, 9, 12148–12156. https://doi.org/10.1109/ACCESS.2021.3051613

Esteva, A., et al. (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature, 542(7639), 115–118. https://doi.org/10.1038/nature21056

Gao, Y., & Zhang, H. (2020). A survey on the application of deep learning in medical image segmentation. Journal of Medical Systems, 44(8), 140. https://doi.org/10.1007/s10916-020-01668-5

Gomes, D. S. M., et al. (2020). Accelerating FPGA implementations for mobile medical devices with high-level AI libraries: An object detection model for colorectal polyp images. IEEE Transactions on Biomedical Engineering, 67(10), 2853–2864. https://doi.org/10.1109/TBME.2020.2967740

Inoue, T., et al. (2021). Toward embedded detection of polyps in WCE images for early diagnosis of colorectal cancer. BMC Bioinformatics, 22, 303.

https://doi.org/10.1186/s12859-021-04303-3

Jafar, A., et al. (2024). Unmasking colorectal cancer: A high-performance semantic network for polyp and surgical instrument segmentation. Engineering Applications of Artificial Intelligence, 138, 109292. https://doi.org/10.1016/j.engappai.2024.109292

Jha, D., et al. (2021). A comprehensive analysis of classification methods in gastrointestinal endoscopy imaging. Medical Image Analysis, 70, 102007. https://doi.org/10.1016/j.media.2021.102007

Klein, S., & Gokhale, R. (2018). Real-time image processing for endoscopic applications. Medical Image Analysis, 44, 162–175. https://doi.org/10.1016/j.media.2018.07.003

Kumar, S., & Mohan, P. (2020). Role of endoscopy in gastrointestinal disorders. Journal of Clinical Gastroenterology, 54(5), 385–392. https://doi.org/10.1097/MCG.0000000000001199

Kumar, V., & Gupta, R. (2021). Machine learning applications in endoscopy: A review. Journal of Clinical Gastroenterology, 55(5), 401–409. https://doi.org/10.1097/MCG.0000000000001346

Lee, J. H., & Kim, J. H. (2019). Advances in endoscopic imaging: Techniques and applications. World Journal of Gastroenterology, 25(12), 1502–1511. https://doi.org/10.3748/wjg.v25.i12.1502

Lee, K. H., et al. (2021). A locally-processed lightweight deep neural network for detecting colorectal polyps in wireless capsule endoscopes. Journal of Medical Imaging, 8(4), 044002. https://doi.org/10.1117/1.JMI.8.4.044002

Liao, J., Xu, Y., & Zuo, Z. (2021). Lightweight deep learning model for real-time colorectal polyp segmentation. IEEE Access, 9, 103732–103740. https://doi.org/10.1109/ACCESS.2021.3098854

Liu, J., et al. (2020). Implementation of a convolutional neural network into an embedded device for polyps detection. Journal of Real-Time Image Processing, 17(6), 1945–1957. https://doi.org/10.1007/s11554-019-00935-7

Liu, Y., & Zhang, Y. (2020). Deep learning in medical image analysis: A survey. Medical Image Analysis, 66, 101857. https://doi.org/10.1016/j.media.2020.101857

Nasir, M. S. B. M. H. M., et al. (2021). An FPGA implementation of SVM for type identification with colorectal endoscopic images. Journal of Healthcare Engineering, 2021, 8899180. https://doi.org/10.1155/2021/8899180

Noor, M. N., et al. (2023). Localization and classification of gastrointestinal tract disorders using explainable AI from endoscopic images. Applied Sciences, 13(15), 9031. https://doi.org/10.3390/app13159031

Schoenfeld, A. J., & McCarthy, S. (2021). Endoscopic techniques for colorectal cancer screening and prevention. Gastrointestinal Endoscopy Clinics of North America, 31(1), 1–12. https://doi.org/10.1016/j.giec.2020.08.001

Selvaraj, J., & Jayanthy, A. K. (2023). Automatic polyp semantic segmentation using wireless capsule endoscopy images with various convolutional neural network and optimization techniques: A comparison and performance evaluation. Biomedical Engineering: Applications, Basis and Communications, 35(6), 2350026. https://doi.org/10.4015/S1016237223500266

Shah, P., Patel, V., & Canny, J. (2020). A comprehensive study on colorectal polyp segmentation with ResUNet++, conditional random field, and test-time augmentation. IEEE Access, 8, 193667–193682. https://doi.org/10.1109/ACCESS.2020.3032236

Varam, D., et al. (2023). Wireless capsule endoscopy image classification: An explainable AI approach. IEEE Access, 11, 105262–105280. https://doi.org/10.1109/ACCESS.2023.3280923

Wang, S., et al. (2022). GastroNet: Gastrointestinal polyp and abnormal feature detection and classification with deep learning approach. Computerized Medical Imaging and Graphics, 96, 102017. https://doi.org/10.1016/j.compmedimag.2022.102017

Wei, W., Zhang, Y., & Xie, Y. (2021). Polyp segmentation of colonoscopy images by exploring the uncertain areas. Electronics, 12(9), 1962. https://doi.org/10.3390/electronics12091962

Yang, R., & Yu, Y. (2021). Artificial convolutional neural network in object detection and semantic segmentation for medical imaging analysis. Frontiers in Oncology, 11, 638182. https://doi.org/10.3389/fonc.2021.638182

Zhang, H., et al. (2021). A low-power and real-time architecture for Hough transform processing integration in a full HD-wireless capsule endoscopy. Sensors, 21(19), 6378. https://doi.org/10.3390/s21196378

Zhang, Y., & Xu, Y. (2020). Machine learning in endoscopic image analysis: A review. Journal of Medical Systems, 44(8), 136. https://doi.org/10.1007/s10916-020-01669-4

Zhang, Y., et al. (2021). Implementation on customizable embedded DSP core for colorectal tumor classification with endoscopic video toward real-time computer-aided diagnosis. IEEE Access, 9, 91042–91052. https://doi.org/10.1109/ACCESS.2021.3089832

Zhang, Z., & Wang, S. (2020). Real-time polyp detection in endoscopic images using deep learning. IEEE Transactions on Medical Imaging, 39(5), 1429–1440. https://doi.org/10.1109/TMI.2019.2930782