Polyp Segmentation : CS 7643

Ashok Kamath Zahin Awosaf Quincy Nickens
Georgia Institute of Technology
akamath76@gatech.edu zawosaf3@gatech.edu qnickens3@gatech.edu

Abstract

Early detection of polyps is critical to prevention of colorectal cancer, the primary cause of death worldwide. For this project, we are focused on evaluating and improving deep learning segmentation methods for polyp detection, focused on generalization across multiple small benchmark datasets. Established architectures presented include U-Net, U-Net++, ResUNet++ and DUCK-Net. Also presented a custom model, HybridNet. The approach uses standard preprocessing, data augmentation, and architectural modifications to improve performance. Results show that while established models perform well on individual datasets, customized models achieve stronger generalization with ResUNet++ attaining the highest overall performance. The results presented highlight the importance of model simplicity and data training strategies for small medical imaging datasets.

1. Introduction/Background/Motivation

In 2022, approximately 1.9 million people around the world were diagnosed with colorectal cancer. By 2040, based on population aging and growth expectations, it is estimated that there will be 3.2 million cases annually. Colorectal cancer is the second most lethal cancer, after only lung cancer, as it killed 900,000 people in 2022 and comprised about 10 percent of all cancer deaths that year. Polyps are the forerunner of colorectal cancer, so their early detection can be lifesaving for an individual. Currently, there is a global health push for increased screening in the form of colonoscopies for early detection of polyps. However, without highly reliable machine learning methods for polyp detection, these efforts may go to waste.

Our objective is to examine and potentially improve current state-of-the-art deep learning methods for polyp image segmentation using small datasets. These medical images are derived from colonoscopies performed in humans. Polyps are tissue growth on the inner lining of the colon or rectum. Some polyps are tumors while others are benign. The goal of a deep learning model for this task is to take an RGB image as input and highlight, or segment, the portion of the image where a polyp is in a binary mask. We chose to focus on small polyp datasets since obtaining high performance on these is more challenging, yet more cost effective than using larger polyp datasets.

This seemingly simple responsibility is complicated by the polyps' frequently small size or flat shape, and their ability to blend in with the surrounding color and texture inside the colon. Furthermore, the model must avoid mistaking normal tissue for a polyp. A frequent limitation of today's models is that a model trained on one benchmark dataset, such as Kvasir, may not generalize well to another, such as CVC, since the camera hardware, illumination, and colon preparation may differ between datasets. This is ultimately the problem we aim to solve: achieving high generalized polyp image segmentation performance across benchmark datasets. Our standard of measurement is the Intersection over Union metric. Our objective is to find or design a deep learning model that is trained on all 4 datasets and then performs well on test sets from each of the datasets.

Current polyp image segmentation today is driven by the benchmark datasets such as CVC-ClinicDB and Kvasir-SEG. For each sample in the datasets, there is typically an image of the colon along with the ground truth label, which is a binary mask that highlights the polyp in the image. Highly trained radiologists or gastroenterologists are responsible for creating an accurate ground truth label. After gathering the data and annotating the ground truth, data preprocessing occurs. During this stage, images are resized to fixed dimensions, pixel values are normalized, and the images are sometimes augmented with flips, rotations, crops, or other tactics. Augmentation is executed to ensure the model generalizes well regardless of image orientation, preventing overfitting while balancing bias and variance.

Following data preprocessing, model training and testing begins. Standard models for medical image segmentation include U-Net, U-Net++, DeepLabV3+, and TransUNet. For loss functions, practitioners typically use Dice Loss, Binary Cross Entropy, or Intersection over Union.

One of the datasets we are using is Kvasir-SEG, which

consists of 1000 polyp images and their corresponding binary ground truth segmentations. This dataset was created from the original Kvasir dataset, which was created by the Simula Research Laboratory in Norway. The creators of Kvasir-SEG drew images from the polyp class of Kvasir and created the segmentations using Labelbox, which is a tool for creating regions of interest in an image frame. CVC-ClinicDB consists of 612 images taken from 31 colonoscopy video sequences of 23 patients at Hospital Clinic Barcelona. The images have a resolution of 384x288 pixels and were annotated by gastroenterologists in Barcelona. CVC-ColonDB contains 300 images taken from 13 video sequences from 13 patients. Each image has a resolution of 500x574 pixels. Finally, ETIS-LaribPolypDB contains 196 images taken from colonoscopy videos and was designed specifically to be challenging for segmentation models. Only images with diverse polyp shapes, sizes or textures were selected.

2. Architectures

In this section we provide small descriptions and the architectures about the models we used for our experimentations.

2.1. U-Net

Having a symmetric "U" shape, U-Net is a fully convolutional network designed for medical segmentation data [5]. Without a fully connected layer, U-Net uses skip connections coupled with a upsampling decoder to achieve accurate results with limit quantities of data. U-Net's architecture is displayed in Figure 4

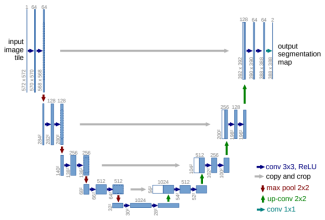

Figure 1. U-Net architecture. Blue boxes correspond to multi-channel feature maps, white boxes represent copied feature maps. [5].

2.2. U-Net++

U-Net++ seek to improve on U-Net by bridging the gap between encoder and decoder with dense, nested skip layers that allow low-level features in the encoder to be associated with high level features in the decoder [6]. These skip pathways contain dense convolutional blocks. Deep supervised learning allows for better optimization.

2.3. ResUNet++

ResUNet++ also seeks to improve on U-Net's architecture by adding additional features to assist with learning. Residual layers are added as a "shortcut" in each encoder/decoder to prevent vanishing gradients. Squeeze-and-Excitation blocks are added after residual layers to adaptively reweight features to emphasize the most important ones. Attention blocks are added in the decoder to highlight the most informative parts of the feature maps. Finally, an Atrous Spatial Pyramid Pooling layer is added between encoder and decoder to capture multi-scale information. The network is designed to work with scale variations and capture finer details [3].

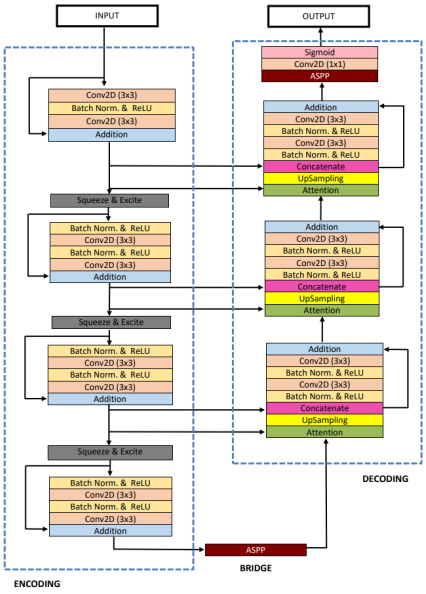

Figure 2. Diagram of ResUNet++ architecture. [3].

2.4. DUCK-Net

DUCK-Net (Deep Understanding Convolutional Kernel) is a CNN architecture designed for medical image segmentation with limited information and computation resource. Figure 3 shows the convolutional "DUCK" block used by the architecture. The block consists of a number of parallel layers, each designed to identify features at varying scale. The output of the block is summed, allowing the model to choose the best feature scale for each region of the image. The network is designed to generate a model quickly from scratch without sacrificing the detail of a larger model [2].

2.5. HybridNet

HybridNet is an efficient segmentation architecture building on ideas from ResUNet++, RAPUNet, and DeepLabV3+, it combines residual feature extraction,

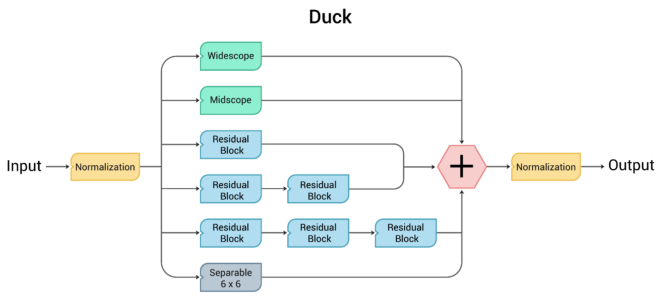

Figure 3. Diagram of DUCK-Block parallel layers structure [2].

attention-guided skip connections, and multi-scale contextual learning within a compact U-shaped encoder-decoder framework.

The encoder is composed of three stages, each implementing a Residual Double Convolution block featuring two successive 3×3 convolutions, batch normalization, ReLU activations, and a skip connection. Max-pooling layers downsample the spatial resolution after each block while increasing feature dimensionality, ensuring deeper, richer representations while maintaining stable gradient flow—following the principles introduced in ResUNet++.

At the bottleneck, HybridNet incorporates a Mini-ASPP module inspired by DeepLabV3+, applying parallel dilated convolutions with rates of 1, 6, 12, and 18 to capture multi-scale contextual information. The outputs are concatenated and compressed through a 1×1 convolution, enabling efficient feature aggregation without significantly increasing model complexity.

The decoder mirrors the encoder and includes three upsampling stages using transpose convolutions. Each encoder feature passed through a skip connection is refined by an Attention Gate, drawing inspiration from RAPUNet, which allows the model to emphasize polyp-relevant features while suppressing irrelevant background. These attended features are then concatenated with the decoder outputs and refined through additional Residual Double Convolution blocks, improving segmentation boundary sharpness and precision.

3. Approach

We started by training and testing standard medical image segmentation models such as the U-Net and U-Net++. Then, we experimented with some under-the-radar models such as DuckNet and ResUNet++. For data preprocessing, since the size of the images varied from dataset to dataset, we standardized the images to 256 by 256 pixels with 3 channels or to 352 by 352, depending on which model we are using. The RGB values were transformed and standardized to between 0 and 1 as well.

The original DUCK-Net implementation used tensor-

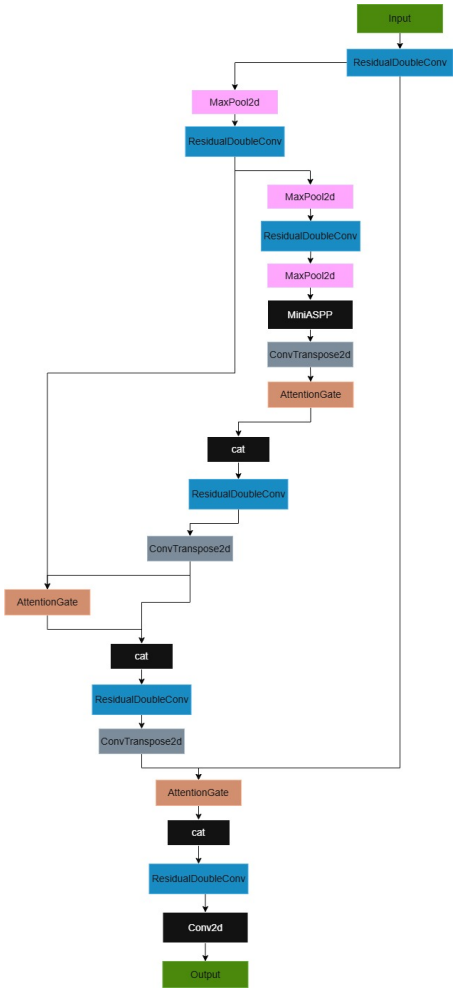

Figure 4. HybridNet Architecture

flow [1], however, for this project, an implementation using PyTorch was adapted for testing [4]. Adaptions were made to the code to provide consistent measurements and visuals for model comparison, training code to facilitate generalization comparison between datasets, data augmentation, and model adaptions to show the effect of adding or removing layers from the Duck-Blocks. The adapted code can be found at the following link: https://github.com/afr0thunder/Duck_Net-Amended

4. Experiments and Results

We measured success based on the performance of a model on test sets from each of the original 4 datasets. We chose to use Intersection over Union (IoU) and the Dice score as the main metrics of comparison since we wanted to measure the overlap between the predicted segmentation and the actual segmentation, which is something that metrics like accuracy fail to encompass.

Our first experiment was to train and validate each model

on an individual dataset. From there, the model was tested on a test set from the same dataset and a test set from each of the other 3 datasets. However, training on only one dataset detracted from the model's ability to generalize, which was visible in the model's poor results. For this reason, we moved on to experimenting with a combined training set that included 80 percent of each dataset, which totaled 1748 image samples, along with a validation and test set, each of 220 image samples. Then, the model was trained up to 100 epochs with early stopping using a patience counter ranging from 5 to 20. We expected this to improve performance since the model would see a greater variety of polyps in images with different sizes, shapes and textures during training.

4.1. U-Net Results

The first model we tested with this new approach was the standard U-Net from the PyTorch-Segmentation library, using pre-trained encoder weights from ImageNet, an EfficientNet backbone, and 3 in channels for the RGB layers. Table 2 in the appendix shows the test results. We chose to use pre-trained encoder weights since this allows the encoding portion of the mode to begin with a strong start using representations it has learned from ImageNet. The model stopped early at epoch 34 during training with a patience counter of 5, likely due to the pre-trained weights holding some representations already. On the test sets, the model scored an average of 75.9 percent IoU. The model scored at least 75 percent accuracy on 3 of the test sets, but on the ETIS-LaribPolypDB test set, the U-Net scored only 55 percent IoU, indicating this model has yet to overcome the challenging images that comprise this dataset.

4.2. U-Net++ Results

The next model was U-Net++. This model was also from the PyTorch-Segmentation library, used pre-trained encoder weights from ImageNet, and contained a ResNet34 backbone. For mIoU, this model scored only slightly better than the original U-Net on average while still performing poorly on the challenging ETIS test set. This suggests the model could benefit from training on manipulated or augmented images from the ETIS dataset that could be added to the training set along with the original ETIS training images, potentially improving the model's ability to generalize for these more challenging images.

4.3. ResUNet++ Results

Next, We conducted experiments using the ResUNet++ model. Before training, we applied data augmentation to increase the diversity of the training set, allowing the model to generalize better and become more robust to real-world variations. Specifically, we used the following augmentations: resizing to the desired image size, random horizontal flip, random vertical flip, random 90-degree rotation, color jittering, normalization, and conversion to tensor format. These were implemented using the Albumentations library with a 50

We performed experiments across different filter sizes—16, 17, 32, and 34—and two input resolutions: 256×256 and 352×352 pixels. Additionally, we created a combined training set by merging samples from four datasets to assess the model's generalization ability. Among all configurations, the model achieved the highest average Dice score (80.44 percent) and IoU (69.23 percent) using a filter size of 32 and an input resolution of 352×352 pixels. This configuration also showed strong precision, recall, and accuracy compared to other settings.

To further enhance performance, we generated two augmented images per training sample using the same augmentation pipeline. This substantially increased the variability of the training data, helping the model learn more robust features. As a result, we observed a significant improvement in performance, achieving an average Dice score of 86.07 percent and an average IoU score of 77.25 percent across the four datasets, still using a filter size of 32 and an input size of 352×352 pixels.

4.4. DUCK-Net

Similarly to previous architectures, DUCK-Net had difficulty generalizing between datasets when trained on only one given set. Table 6 summarizes DUCK-Net's performance when trained on all four dataset's tested against test sets from each dataset with an average dice score of 0.7888. Consistently, models generalize well to CVC-ClinicDB and Kvasir-SEG with higher metrics while struggling to generalize to CVC-ColonDB and ETIS-LaribPolyp. ETIS-LaribPolyp has significantly less data than any of the the other three datasets, so it's influence can be understood as lack of information relative to the broader dataset.

DUCK-Net's stated design advantage is the parallel layers from which the model can choose the best feature scale in a region of the image [2]. To better understand the influence of each of these parallel features on performance, each layer was systematically removed during model training while holding training hyperparamters constant. Additional layers were also added to observe impact on performance. Kvasir-SEG was used as the dataset for both training and test with no data augmentation.

During the removal of layers, with few exceptions, each reduction in the model translates to higher performance on all metrics. Table 6 displays the model metrics for altered DUCK-Block architectures. These results indicate that the DUCK-block might be too complex for small segmentation datasets and indicating the model complexity is causing overfitting, accumulating noise or repeated information rather than generating useful feature maps. To illustrate

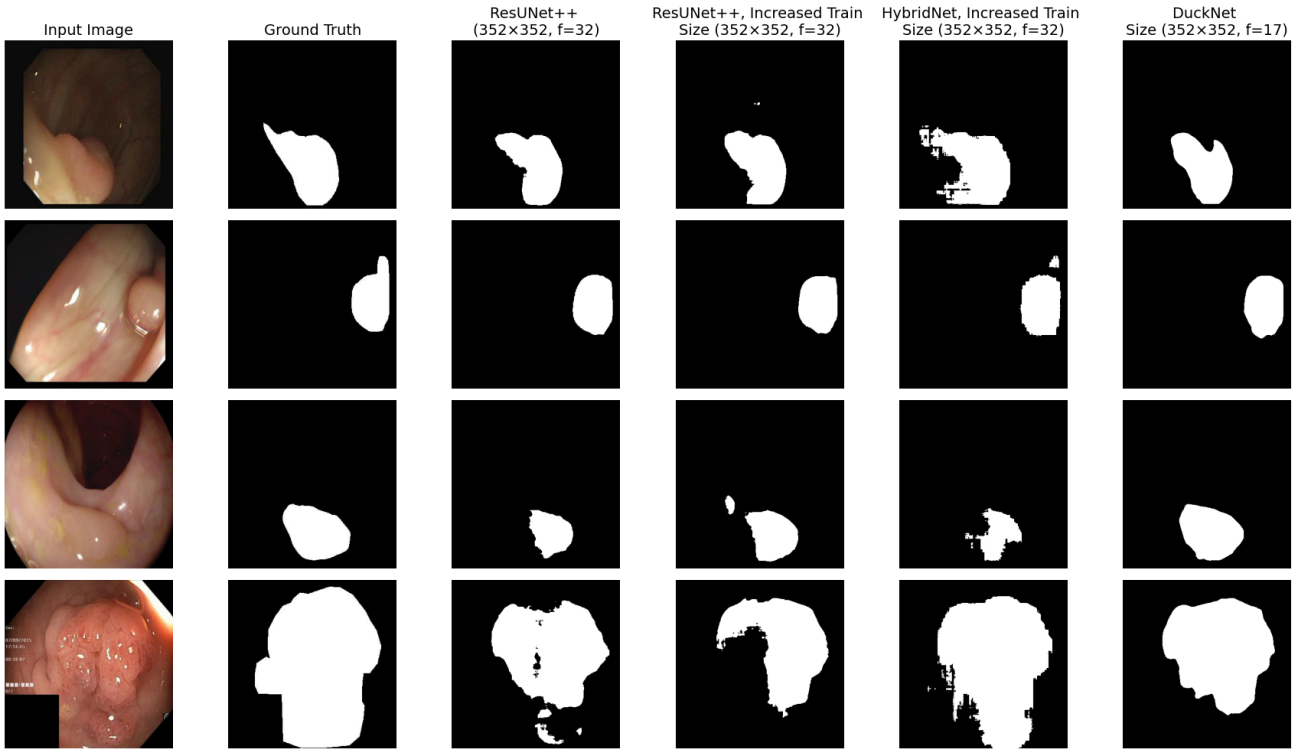

Figure 5. Comparison of prediction masks using different methods.

the issue, additional residual layers were added resulting in consistently poorer performance.

A reduced DUCK-block, containing only four parallel layers was developed. The block contains the Widescope, midscope, and two residual layers resulting in significantly higher performance and lower compute time. The result of the reduced block is also presented in Table 6. A simpler or targeted architecture is better for the smaller datasets presented in this experiment rather than presenting the model with too much information during training. Future work here might compare performance on datasets with wider range in scale.

4.5. HybridNet

As the next step, we conduct experiments with HybridNet, setting the filter size to 32 and the input image resolution to 352×352. The model was trained on a combined dataset while utilizing the simple augmentation strategy discussed earlier, where two images were generated per original sample.

When evaluated across multiple polyp segmentation benchmarks, HybridNet achieved an average Dice score of 79.74 percent and an average IoU of 68.32 percent. The model also attained a high precision of 89.11 percent and an overall accuracy of 97.26 percent. Specifically, HybridNet showed strong performance on datasets such as CVC-ClinicDB (Dice score of 84.82 percent) and Kvasir-SEG (86.83 percent Dice), while maintaining competitive results on the more challenging ETIS-LaribPolypDB (65.00 percent Dice).

Finally, we have shown the comparison of the average test results among the different models have used for our experimentation. In the table, n is how many new images were generated for each image sample, default was 0.

Table 1. Comparison of average test results among different models

Tested On	mDice	mIoU	mPrec	mRec
U-Net	0.8487	0.7660	0.9385	0.8111
U-Net++	0.8448	0.7425	0.9529	0.8135
ResUNet++	0.8044	0.6924	0.8845	0.7734
ResUNet++ (n=2)	0.8607	0.7725	0.9176	0.8248
DuckNet	0.7888	0.7149	0.8745	0.8055
HybridNet (n=2)	0.7974	0.6832	0.8911	0.7548

5. Conclusion

In this project, several deep learning models for polyp segmentation were evaluated with a focus on achieving strong generalization results across limited size medical

image datasets. U-Net and U-Net++ demonstrated high performance within datasets but generalization between datasets had lower metrics. ResUNet++ achieved the highest overall scores, highlighting the benefit of it's design coupled with data augmentation techniques. DUCK-Net also struggled with inter-dataset generalization, but performed better with a pruned DUCK-block infrastructure. HybridNet performance shows a competitive alternative to the established networks. Overall, for small medical image datasets, the results highlight the importance of model simplicity, diverse training data, and targeted augmentations to achieving robust segmentation performance.

6. Illustrations, graphs, and photographs

Table 2. U-Net with Efficient Net Backbone - Test Results Across Datasets (with Average)

Tested On	mDice	mIoU	mPrec	mRec
CVC-ClinicDB	0.9153	0.8461	0.9442	0.8927
CVC-ColonDB	0.9235	0.8581	0.9377	0.9097
ETIS-LaribPolypDB	0.6388	0.5118	0.9382	0.5403
Kvasir-SEG	0.9172	0.8481	0.9340	0.9018
Average	0.8487	0.7660	0.9385	0.8111

Table 3. U-Net++ Test Results Across Datasets (with Average)

Tested On	mDice	mIoU	mPrec	mRec
CVC-ClinicDB	0.9423	0.8543	0.9502	0.9352
CVC-ColonDB	0.9202	0.7653	0.9423	0.9015
ETIS-LaribPolypDB	0.6010	0.4862	0.9715	0.5288
Kvasir-SEG	0.9158	0.8642	0.9476	0.8883
Average	0.8448	0.7425	0.9529	0.8135

Table 4. ResUNet++ on Combined Training Set - Test Results Across Datasets (with Average)

Tested On	mDice	mIoU	mPrec	mRec
CVC-ClinicDB	0.8524	0.7469	0.8983	0.8164
CVC-ColonDB	0.8338	0.7181	0.8739	0.8005
ETIS-LaribPolypDB	0.6694	0.5446	0.8758	0.6361
Kvasir-SEG	0.8621	0.7598	0.8899	0.8405
Average	0.8044	0.6924	0.8845	0.7734

Table 5. ResUNet++ on Combined Training Set with generated 2 images per samples - Test Results Across Datasets (with Average)

Tested On	mDice	mIoU	mPrec	mRec
CVC-ClinicDB	0.8999	0.8222	0.9348	0.8747
CVC-ColonDB	0.8928	0.8085	0.9194	0.8687
ETIS-LaribPolypDB	0.7584	0.6532	0.8881	0.6959
Kvasir-SEG	0.8916	0.8061	0.9283	0.8597
Average	0.8607	0.7725	0.9176	0.8248

Table 6. DUCK-Net performance across all four datasets, averages are weighted to test set size.

Tested On	mDice	mIoU	mPrec	mRec
CVC-ClinicDB	0.8338	0.7594	0.8974	0.8473
CVC-ColonDB	0.6518	0.5881	0.8744	0.6400
ETIS-LaribPolypDB	0.5408	0.4853	0.8505	0.5128
Kvasir-SEG	0.8594	0.7785	0.8649	0.8973
Average	0.7888	0.7149	0.8745	0.8055

Table 7. Modular analysis of DUCK-Net performance with removed and added parallel layers using Kvasir-SEG.

Layer	mDice	mIoU	mPrec	mRec
No Adjustment	0.7412	0.6542	0.8586	0.7216
Removed: Widescope	0.8195	0.7336	0.8617	0.8338
Removed: Midscope	0.7979	0.7083	0.8283	0.8370
Removed: Separable	0.8465	0.7717	0.8791	0.8621
Removed: Residual x1	0.8583	0.7743	0.8688	0.8827
Removed: Residual x2	0.7873	0.6913	0.7976	0.8655
Removed: Residual x3	0.8824	0.8092	0.9106	0.8874
Added: Residual x4	0.6839	0.5757	0.7082	0.7965
Added: Residual x5	0.7822	0.6846	0.8983	0.7568
Reduced Block	0.8776	0.8051	0.9008	0.8855

Table 8. HybridNet on Combined Training Set with generated 2 images per samples - Test Results Across Datasets (with Average)

Tested On	mDice	mIoU	mPrec	mRec
CVC-ClinicDB	0.8482	0.7391	0.8839	0.8233
CVC-ColonDB	0.8230	0.7025	0.8806	0.7752
ETIS-LaribPolypDB	0.6501	0.5222	0.9117	0.5645
Kvasir-SEG	0.8683	0.7691	0.8883	0.8564
Average	0.7974	0.6832	0.8911	0.7548

7. Work Division

Student Name	Contributed Aspects	Details
Ashok Kamath	Wrote Section 1, Contributed to Sections 3 and 4	Researched importance of polyp segmentation, each of the datasets, ran training and testing for U-Net and U-Net++ with cross testing across datasets
Zahin Awosaf	Contributed to Section 2 and 4	Generated new image samples using data augmentation and trained and tested ResUNet++ and HybridNet on these data, with cross testing across datasets.
Quincy Nickens	Contributed to Abstract and Sections 2, 3, and 4	Trained DUCK-Net models and compiled results, performed alterations to DUCK-blocks, short descriptions of presented architectures: section 2 (aside from HybridNet)

Table 9. Contributions of team members.

References

[1] Razvan Duma. DUCK-Net: Dilated u-net with a convolutional kernel for polyp segmentation. GitHub repository, 2021. Accessed: [4/18/2025]. 3

[2] Razvan-Gabriel Dumitru, Darius Peteleaza, and Catalin Craciun. DUCK-Net: Dilated u-net with a convolutional kernel for polyp segmentation. *arXiv preprint*, arXiv:2311.02239, 2023. Accessed: [4/18/2025]. 2, 3, 4

[3] Debesh Jha, Pia H. Smedsrud, Michael A. Riegler, Dag Johansen, Thomas de Lange, Pål Halvorsen, and Håvard D. Johansen. Resunet++: An advanced architecture for medical image segmentation. In *IEEE International Symposium on Multimedia (ISM)*, pages 225–2255, 2019. 2

[4] Russell Kim. DUCK-Net-PyTorch: Pytorch implementation of dilated u-net with a convolutional kernel for medical image segmentation. GitHub repository, 2023. Accessed: [4/18/2025]. 3

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. 2

[6] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, volume 11045 of *LNCS*, pages 3–11. Springer, 2018. 2