SCRIBBLE-SUPERVISED SEMANTIC SEGMENTATION FOR HAUSTRAL FOLD DETECTION

Samuel Ehrenstein
Department of Computer Science
University of North Carolina
Chapel Hill, NC
ehrensam@cs.unc.edu

Sarah McGill
Department of Medicine
University of North Carolina
Chapel Hill, NC
mcgills@email.unc.edu

Julian Rosenman
Department of Medicine
University of North Carolina
Chapel Hill, NC
rosenmju@med.unc.edu

Stephen Pizer
Department of Computer Science
University of North Carolina
Chapel Hill, NC
smp@cs.unc.edu

ABSTRACT

Purpose

Colonoscopy is considered the "gold standard" screening procedure for colorectal cancers. But its effectiveness is limited by the fact that endoscopists sometimes fail to view all parts of the colon surface. We seek to build a system for detecting these missed areas during the procedure, alerting the endoscopist, and providing means to guide him or her back to view the missed areas.

Methods

As part of a system for detecting these missed areas and ameliorating these misses, it is useful to provide a means for the computer to comprehend the colon geometry viewed by a video frame. To that end, we introduce a new deep learning method for semantic segmentation of colonoscopy video frames to detect haustral folds. Our method is based on the DeepLabV3+ neural network architecture and takes as input frame colors and per-frame depth maps produced by a reconstruction method. Our method is trained using scribble supervision, a type of weakly-supervised learning.

Results

We show that our method achieves good results and outperforms the state of the art for haustral fold detection from video frames, with a pixel accuracy of 90% compared to 66% for the state of the art. In addition, we demonstrate that our method produces consistent segmentations over colonoscopic video sequences.

Conclusion

Our method of using scribble supervision to train a neural network for detecting haustral folds outperforms the state of the art for this task and achieves high accuracy and consistent results over consecutive frames. Thus our method can potentially be used for localization within the colon.

Keywords Colonoscopy · Haustral fold detection · Scribble supervision · Deep learning · Reconstruction

1 Introduction

Colonoscopy is a standard procedure for detecting and preventing colorectoral cancer. It is performed by a physician, who inserts an endoscope into the patient's colon and visually inspects it for cancerous or precancerous growths. However, colonoscopy's effectiveness is limited by the fact that some areas of the colon surface are unintentionally not surveyed. Methods have been proposed for detecting these missed areas, or blind spots, but once detected, a solution is needed to guide the endoscopist back to survey the blind spot. One approach is to use the ridges on the colon, known as haustral folds, as reference features to determine the location of the endoscope and blind spot.

Using haustral folds for navigation first requires being able to reliably detect them in the video feed from the endoscope. The most reliable method in literature for this detection is based on CycleGAN [18]. In this paper, we propose a more effective method, utilizing a training approach known as scribble supervision.

1.1 Cancer and its detection

Colorectal cancer (CRC) is the third-most common cause of cancer death in the United States [15]. Colonoscopy is the standard screening procedure for CRC and is recommended for all adults when they reach age 45 and every 10 years thereafter. In the year 2000, 20% of adults had undergone colonoscopy in the past 10 years. By 2018, the figure was 61%, and this increase is believed to be responsible for significant reductions in CRC morbidity and mortality over this time period [12, 4]. In colonsocopy the endoscopist uses an optical endoscope to visually examine the surface of the patient's colon. A wire loop inserted through a hollow channel in the endoscope is used to remove adenomatous colon polyps, the cause of almost all colorectal cancers [5, 1]. But colonoscopy still misses between 6 and 27 percent of polyps, many due to the endoscopist not surveying some regions of the colon surface [17].

1.2 Detection of blind spots

Ma et al. [7] developed a method of detecting these blind spots (missed regions of the colon surface) through real-time 3D reconstruction of the colon surface. Once a blind spot is detected, the endoscopist may wish to be guided back to it in order to survey the region. This requires localization of the endoscope tip within the colon, which is made difficult by the discontinuous nature of the 3D reconstructed segments, as well as the non-rigid nature of the colon. In our situation, the only external information available is the video feed from the endoscope itself, which consists of a sequence of frames. Our task is to use this information, and information derived from it in real time, to determine the location of the endoscope tip in the colon relative to detected blind spots. In particular, when the endoscope tip is within 10-15 cm of the blind spot, we need to be able to show its position in real time (along with the blind spot) on an onscreen visualization. This will allow the endoscopist to precisely maneuver back to inspect the blind spot. To do this, we need to be able to associate what is currently being observed with what was observed in the same region when the blind spot was missed. While a pose computed via SLAM will be available, it will not necessarily be known relative to the blind spot. Thus we need a direct localization method based on current observations.

Previous work on the problem of localization during colonoscopy is sparse. One approach was to use photometric features with place-recognition methods taken from the field of robotics [8]. However, since the visual appearance of the colon can change with deformation, an alternative is to consider geometric features of the colon. The most prominent of these are the haustral folds, which appear as ridges on the colon's surface and are thought to be created through contractions of smooth muscle surrounding the colon [3].

The first step in using haustral folds for navigation is to be able to reliably detect them in a video frame, which can be done by image segmentation. The most reliable method until now is known as FoldIt [9]. It is based on CycleGAN [18] and treats the problem of fold detection as one of domain translation. While FoldIt does not require images from optical colonoscopy to be labeled, its accuracy degrades for small folds, and its generative approach leads to strong priors which cause false positives. Instead of using a generative approach, we treat the problem as classification of pixels, i.e. semantic segmentation.

1.3 Our Contributions

In this work, we introduce an alternative deep learning-based method of haustral fold detection. Instead of training on unpaired data, we chose to use scribble annotations as labels for weakly-supervised learning [6]. Traditionally, semantic segmentation is learned from a dataset of images which have every pixel given a class label. The downside to this approach is that fully labeling an image is very time-consuming. In addition, many pixels are ambiguous and can be reasonably assigned to either "fold" or "not fold" classes. Thus, we chose to use scribbles to label only a selection of pixels whose classes are clearly evident.

As this method is designed to be used in the same product as [7], it is possible to take advantage of the geometry models therein. Specifically, one model, called ColDE [16], predicts the distance from the camera, or depth, at each pixel in every frame. We thus combine these depth maps with the color frames, creating a 4-channel, RGB-D input for our model, demonstrated in Fig. 1.

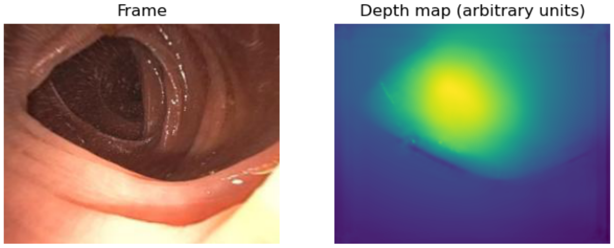


Figure 1: Left: A frame from colonoscopy video. Right: its depth map in arbitrary but consistent units, as predicted from [16].

The contributions of this paper can be summarized as follows:

- We propose and evaluate an approach which outperforms FoldIt at detecting haustral folds over a variety of metrics. In addition, our approach learns a one-to-one mapping, whereas FoldIt learns a many-to-many mapping. Thus our approach is more flexible in terms of training data, as continuous video sequences are not required.
- To our knowledge, we are the first to combine scribble supervision with monocular depth information, as well as the first to formulate the problem of haustral fold detection as one of discriminative semantic segmentation.

We expect to be able to release our code and data soon; these will be found at `https://github.com/qscgy/ridge-detector`.

2 Related Work

2.1 Scribble supervision

The goal of scribble-supervised semantic segmentation is to assign a class label to each pixel in an image, while only learning from a set of images with a small fraction of their pixels labeled. Specifically, the labels are "scribbles", that is, curves drawn on the image (see Fig. 3). Each scribble has one class, and the pixels under the scribble are assigned that class in the label. All other pixels are marked as "unknown" class. Although only a small fraction, usually less than 5%, of pixels are labeled, it has been found that scribble supervision can still produce results comparable to fully-supervised approaches [14, 13].

2.2 FoldIt

FoldIt [9], a method based on CycleGAN [18], is the best previous method of fold detection to our knowledge. It formulates the problem of fold detection as a problem of translating between image domains. The authors use three domains: domain A, images from optical colonoscopy; domain B, images from virtual colonoscopy, that is, grayscale images without texture taken inside 3D models of colons derived from CT scans; and domain C, images inside 3D colon models where folds have been marked in red using the mathematical algorithm from [10] [1]. Like CycleGAN, FoldIt uses unpaired data; there is no correspondence between specific images in separate domains. The authors of [9] released two pre-trained FoldIt models with the same architecture trained on different datasets: one trained on a subset of the same data we used, and another trained on a public dataset of frames from optical colonoscopy. While the FoldIt authors trained on the same data as this work for the figures in their paper and they claim that there is no significant difference with FoldIt trained on public data, we have found that the latter always outperforms the former on the metrics we used for evaluation in Sec. 6.2, at least for optical colonoscopy studies. We will refer to this version simply as FoldIt from now on.

[1] We did not consider this method in this paper because it is a geometric method operating on CT scans of the entire colon, which are not available during optical colonoscopy.

(a) Down the barrel.

(b) En face.

Figure 2: Examples of "down the barrel" and "en face" frames.

2.3 Finding normals and depths

In addition to the RGB color values, we also use the depth values produced by ColDE [16], which provide geometric information. Haustral folds are geometric features, but they express themselves in images as variations in the color, i.e. as photometric features. By using RGB with depths, we take advantage of both photometric and geometric features of haustral folds for better inference.

3 Preliminaries

One approach to scribble-supervised semantic segmentation is known as the "two-stage" approach. Scribbles are first used to generate pseudo-labels for every pixel, which are then used to train the network as in fully-supervised learning. However, we chose the alternative, "one stage" approach because it is less computationally intensive to train and has fewer hyperparameters to tune. Instead of generating pseudo-labels, we directly train the network on two loss functions: one supervised loss function over the annotated pixels, and another loss function which is the weighted sum over some set $\{\mathcal{L}_1...\mathcal{L}_R\}$ of regularization loss functions over all pixels.

For an image x of M by N pixels, the set of all pixels is Ω, and $|\Omega| = MN$. For an RGB-D image, $p \in \Omega$ is a vector $p \in \mathbb{R}^4$. Given a network input x, the output $S(x)$ is a set of $|\Omega|$ indicator vectors $S_p \in [0,1]^2$ corresponding to the softmax outputs for the two classes. The subset $\Omega_L \subset \Omega$ is the set of pixels with scribble labels, and the corresponding ground truth labels are $y_p \in \{0,1\}$ for all $p \in \Omega_L$.

We now define the total segmentation loss $\mathcal{L}(x, y)$ to be

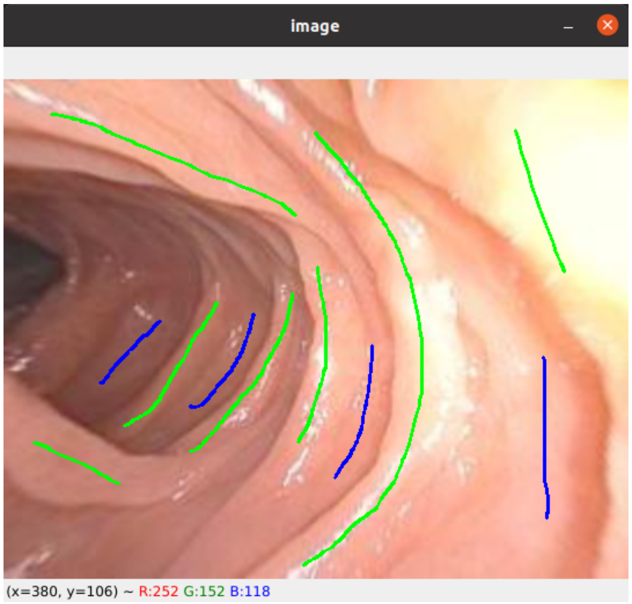
$$\mathcal{L}(x,y) = \sum_{p \in \Omega_L} -y_p \log S_p + \sum_{i=1}^{R} \lambda_i \sum_{p,q \in \Omega} \mathcal{L}_i(S_p, S_q; x) \tag{1}$$

The first term in this equation is the binary cross-entropy loss computed over all labeled pixels. The second term is the total regularization loss, a weighted sum of R regularization losses $\{\mathcal{L}_1...\mathcal{L}_R\}$. Each $\mathcal{L}_i(S_p, S_q; x)$ is defined as the affinity of the segmentation outputs at two pixels p, q, as well as of x, the input. We use the $(\cdot; x)$ notation to emphasize that the function $\mathcal{L}_i$ is parameterized by the entire input x. We explain in further detail in section 5.2.

4 Data

We prepared both training and testing datasets for training and evaluating our methods, respectively, by uniformly sampling our research group's library of colonoscopy video sequences. We then processed these sampled frames using ColDE [16] to predict a dense pixel depth map, which was concatenated to the image to form an RGB-D image. To these we applied scribble annotations (Sec. 4.2).

Most frames in both the training and test sets had a small angle between the camera and the longitudinal axis of the colon (known as "down the barrel" frames). While this is representative of the population of frames in video sequences that our group can presently successfully reconstruct for blind spot detection [7, 16], it does mean that there are relatively few frames in the training and test sets with a large angle between camera and colon ("en face" frames). Examples of both of these types of frames are shown in Fig. 2.



(x=380, y=106) ~ R:252 G:152 B:118

Figure 3: Example of scribble annotations drawn in our scribble annotation tool. Green scribbles are of category "fold", and blue scribbles are of category "not fold".

4.1 Training set

Our training dataset was obtained by first taking every fifth frame, sequentially, of each sequence in our research group's library of colonoscopy frames from approximately 78 patients, for a total of 2556 frames in the training set. Scribble annotations were then applied to 1828 of these frames; the remainder lacked sufficient detail to make annotations with high confidence. We then processed these 1828 frames using [16], as explained above.

4.2 Annotations

Scribble annotations were applied to images using a tool we wrote in Python; example annotations are shown in Fig. 3. The user has the option to draw color-coded scribbles for the two classes, "fold" and "not fold", and can also erase and change between frames using the keyboard. Scribbles were applied only to pixels where the expert annotator had high confidence that the class label was correct. For "fold" scribbles, the scribbles were to follow the curve of the fold, close to its crest but not including any pixels behind it. On each frame, the annotator's goal was to put a scribble in connected components with the proper class label, without making low-confidence annotations. Some of the selected frames were unable to be annotated and were excluded from the training set, due to the fact that they did not contain any regions that could be labeled with adequate confidence.

4.3 Data augmentation

Data augmentation was used in order to increase the diversity of the training set. Images were flipped horizontally at random with a 50% probability and randomly cropped in order to increase the range of physical scales represented in the data. In order to introduce more labeled pixels, we also exploited the fact that scribbles were drawn well away from the expected boundaries between "fold" and "not fold" regions and increased the stroke width of annotations. These thicker scribbles were manually examined on a sample of training images to confirm that no pixels were being mislabeled.

4.4 Test set

In addition to the training set, a test set was prepared and annotated. This set consisted of 64 video frames and their predicted depth maps. This set was drawn at random from the same library of video frames as the training set, but excluding the frames in the training set so that the test set is unseen by the network. Each of these frames was scribble-annotated, this time attempting to place scribbles on as many connected components of the full segmentation as possible in addition to the criteria used for annotating the training set. Across all 64 images, there were 203,170 labeled pixels, with 101,604 "fold" and 101,566 "not fold".

As explained above, the test set does not contain many "en face" frames. As we expect the depth map inputs to be most useful on these frames, this test set cannot fully evaluate the advantage of depth map inputs. We discuss this further in Sec. 6.3.

5 Methods

5.1 Network architecture

Our network architecture is similar to the DeepLabV3+ version used in [14], with two notable changes. First, instead of the ResNet-101 backbone, we use MobileNetV2 [11] as our backbone. This is a network designed to run on mobile devices, so it runs faster then ResNet-101 on the same hardware. Second, we modified the first layer of MobileNetV2 to a 4-channel input while retaining the pretrained weights via the method used in the `timm` Python package. The RGB channels initialize to their respective pretrained weights, while the depth channel initializes to the R channel weights. This was done in order to initialize the depth channel with pretrained weights instead of random. All weights in all other layers initialize to the pretrained MobileNetV2 weights, as they are unaffected by the change.

Diagrams of the full system at training and inference time can be found in appendix A.

5.2 Loss functions

While the cross-entropy loss term alone from (1) is enough to obtain decent performance, it has been found [13, 14] that adding regularization losses $\{\mathcal{L}_i\}$ can improve performance. In this work, we experiment with two regularization losses: the dense conditional random field loss $\mathcal{L}_{CRF}$ and the normalized cut loss $\mathcal{L}_{NC}$, both from [14] (defined below).

5.2.1 Dense CRF loss

The dense conditional random field regularization loss (dense CRF)[14] computes the affinity between every pair of pixels, and is low when pixels with high affinity for each other are labeled in the same class.

Let $S^k \in [0,1]^{|\Omega|}$ be the vector of the kth components of every S_p, and let the affinity matrix $\hat{W} \in \mathbb{R}^{|\Omega|\times|\Omega|}$ be a function of the input image x giving a weight to every pair of pixels $p, q \in \Omega$ based on their similarity to each other. We detail the affinity matrix below. We can write the dense CRF loss as

$$\mathcal{L}_{CRF} = \sum_{k\in\{0,1\}} (S^k)^T \hat{W}(\mathbf{1} - S^k) \tag{2}$$

5.2.2 Normalized cut loss

The normalized cut loss is an extension of $\mathcal{L}_{CRF}$.

$$\mathcal{L}_{NC} = \sum_{k\in\{0,1\}} \frac{(S^k)^T \hat{W}(\mathbf{1} - S^k)}{d^T S^k} \tag{3}$$

The degree vector d is equal to $\hat{W}\mathbf{1}$. This allows us to simplify the loss a bit by removing a constant term, so we have

$$\mathcal{L}_{NC} = -\sum_{k\in\{0,1\}} \frac{(S^k)^T \hat{W} S^k}{d^T S^k} \tag{4}$$

$\hat{W}$, assigns the weight every pixel applies to the loss value at a given pixel. Since this weight is very small for far-away pixels, we follow [14] and find $\hat{W}S^k$ by applying a bilateral Gaussian (RGB vs. XY) filter to the RGB channels of the input x, so that

$$\hat{W}_{ij}(x) = \frac{1}{W_p} \sum_{(k,l)\in\Omega} g(||I(k,l) - I(i,j)||_2; \sigma_r) g(||(k,l) - (i,j)||_2; \sigma_s) \tag{5}$$

where W_p is a normalization constant, $g(\cdot;\sigma)$ is the 1-dimensional Gaussian centered at 0 with standard deviation σ (for hyper-parameters σ_r, σ_s), and where $||\cdot||_2$ is the L2 vector norm. Thus $\hat{W}(x)$ is a function of the distances and differences in intensities (i.e. the distance in RGB space) between a pixel and all other pixels.

5.3 Implementation details

All training and testing was performed on a computer running Ubuntu 20.04 LTS with an NVIDIA GeForce Titan X GPU. Our code is built on that of [14], which provided an implementation of $\mathcal{L}_{CRF}$. We used their C++ bilateral filter implementation to implement $\mathcal{L}_{NC}$.

6 Results

We performed our experiments using the test set described in Sec. 4. Evaluation metrics were calculated only based on the labeled pixels. We conducted two types of experiments, one studying the performance on each class overall and the other studying the performance per-frame. In this section we will demonstrate the superior performance of our method against FoldIt, the consistency of our method in identifying folds across consecutive video frames, and the results of ablation studies.

6.1 Experiment descriptions

6.1.1 Per-class experiments

For the first set of experiments, presented in Sec. 6.2.1, metrics used are per-class precision[2], recall[3], and F1 score, as well as overall accuracy for each model. These were chosen because the objective of these experiments is to study overall classification performance, the results of which are one predicted class label per image pixel in each image in the test set. They are standard metrics in machine learning literature for evaluating classifier performance.

Defining TP, TN, FP, FN to be the total numbers of true positive, true negative, false positive, and false negative pixels respectively for a class, we have

$$\begin{aligned}
\text{Precision} &= \frac{\text{TP}}{\text{TP + FP}}, \text{ the fraction of positive classifications that are correct}\\
\text{Recall} &= \frac{\text{TP}}{\text{TP + FN}}, \text{ the fraction of positives that are classified as such}\\
\text{F1} &= \frac{\text{2TP}}{\text{2TP + FP + TN}}, \text{ the harmonic mean of precision and recall}\\
\text{Accuracy} &= \frac{\text{TP + TN}}{\text{TP + FP + TN + FN}}, \text{ the fraction of pixels that are classified correctly}
\end{aligned}$$

These are computed across all 203,170 labeled pixels in all 64 labeled images, so variance cannot be easily presented. The next section describes the second set of experiments we conducted, which attempt to understand the variation in performance between frames.

6.1.2 Per-frame experiments

For the second set of experiments, presented in Sec. 6.2.2, we computed the accuracy on labeled pixels of our two best configurations (no regularizations, with and without depth inputs) on each of our 64 test frames. We did the same for FoldIt for comparison, in order to study the variation of these methods on different types of frames. As these experiments evaluate accuracy over the population of frames, which is a continuous-valued quantity, it is not possible to use the metrics from section 6.1.1. Instead, we consider the distribution of accuracies over the population of frames, presented in Fig. 5.

6.2 Segmentation performance vs. FoldIt

Performance versus FoldIt is shown in Figs. 4 and 5 and Table 1. These results are from a network trained for 96 epochs on RGB-D data with no regularization loss terms. In section 6.3 we present and discuss the results of ablation studies on regularization terms and depth inputs.

6.2.1 Per-class performance

As seen in Fig. 4, in all samples our method produces better segmentations. It is clear in the first three columns that our method correctly segments folds that are entirely missed by FoldIt. In the fourth and sixth columns, FoldIt marks

[2] Also known as positive predictive value (PPV).

[3] Also known as sensitivity or true positive rate (TPR).

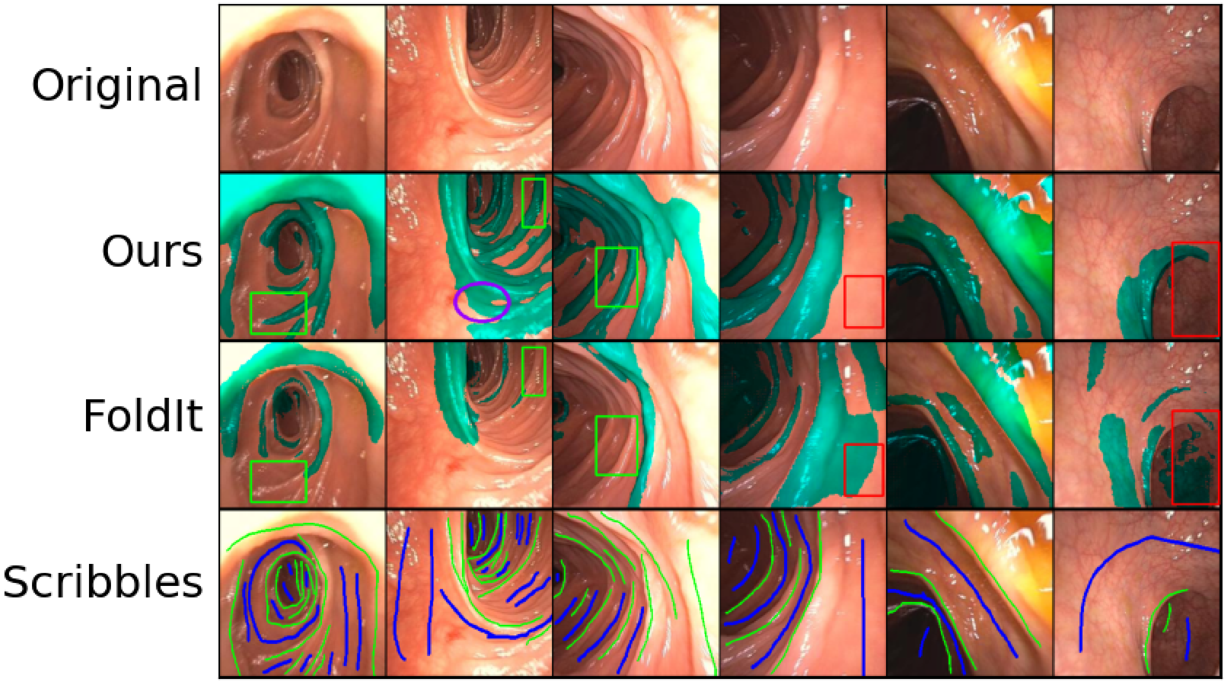


Figure 4: Sample images with the network outputs superimposed in green. Green boxes show regions where our method correctly marked folds which were missed by FoldIt. Red boxes show regions where FoldIt incorrectly marked folds that our method correctly marked as not-folds. The purple oval in the second row of the second column marks a region where our method gave an inconsistent result. The fourth row shows the ground-truth label scribbles.

Class	**Precision**	**Recall**	**F1 score**
Not fold	(**0.91**, 0.62)	(**0.88**, 0.86)	(**0.90**, 0.72)
Fold	(**0.88**, 0.76)	(**0.92**, 0.47)	(**0.90**, 0.58)
Accuracy			(**0.90**, 0.66)

Table 1: Comparison of our method's performance versus FoldIt on the test set of 64 annotated images. Each result is presented as an ordered pair of the form (our result, FoldIt result).

more area as "fold" than it should (red boxes). In the fifth column, FoldIt marks the entire dark area in the lower-left corner as "fold" even though it is not. This can also be seen in the first and fourth columns. Our method does not run into this problem, as it is trained on numerous examples where this part of the image, corresponding to the farthest visible part of the colon wall, is labeled as "not fold". This is in agreement with the results in Table 1, where we found that our method outperforms FoldIt on every metric used for evaluation.

Additionally, 4 videos showing the superior performance of our method versus FoldIt are included in the supplementary material[4]. These sequences are included as "OC Video Sequence 1" through "OC Video Sequence 4" in the supplementary video of [9].

In both Fig. 4 and the video, FoldIt has a tendency to mark as "fold" an approximately circular region near the center of the frame in "down the barrel" views. The colon is never perfectly straight, meaning that even "down the barrel" camera views will be facing the colon wall at a non-infinite distance.

However, our method still has room for improvement. We can see in the second column of Fig. 4 that although it picks up more folds than FoldIt, our method is inconsistent in how it classifies pixels on the sides of folds (circled in purple). The circled region should be either all marked as "fold", or have two parallel but disconnected "fold" regions. In addition, our method sometimes misclassifies any pixels close to maximum intensity as "fold", and these are common in colonoscopy as a result of specular reflection off of wet surfaces.

[4] Also linked here: `https://drive.google.com/drive/folders/1m1mgsnbj4MBaPA21sM3-TgEa73ov4ZLg?usp=drive_link`

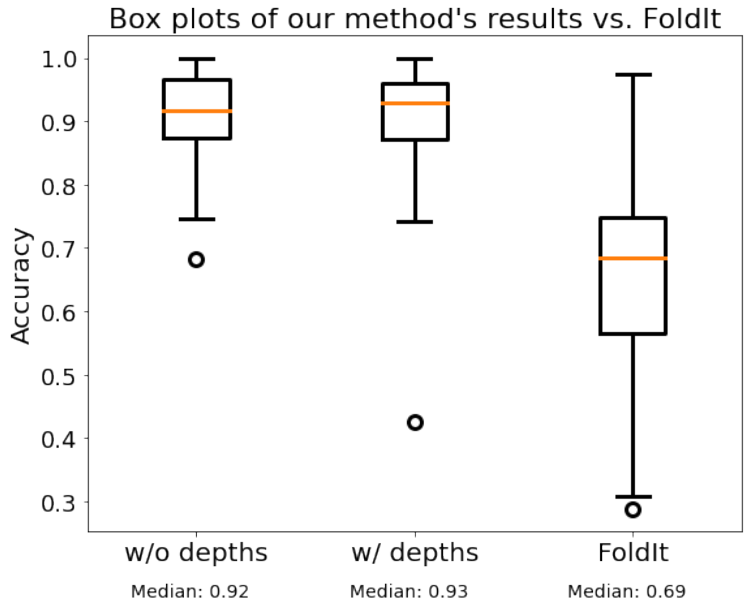


Figure 5: Box plots of accuracy over the population of frames.

6.2.2 Per-frame performance

We see clearly in Fig. 5 that our method, trained with or without depth inputs, produces a higher median accuracy on the frames of the test set with lower variance than FoldIt. This is significant because if this method is to be used for localization, it needs to perform consistently in all parts of the colon.

6.2.3 Timing

As our neural network is fully feedforward and deterministic, the time to process each frame is asymptotically constant, but the first frame processed takes much longer due to the way PyTorch allocates memory. In clinical use, the model will be instantiated once and then used for the duration of the procedure, so the first frame can be ignored in calculating time per frame. We report 0.017 ± 0.0025 sec/frame over 63 of the 64 frames in the test set, corresponding to 59.9 frames per second.

6.3 Ablation studies

As part of this work, we conducted ablation studies on the two regularization losses (normalized cut and dense CRF) and on the use of predicted depth maps. The results, shown by the validation accuracies during training, are in Fig. 6. We found that the best results occurred with no regularization losses, and there was no statistically significant difference with or without depth input, although in Fig. 5 we see that the model trained with depths has a slightly higher median accuracy. The lack of significant difference with or without depths is likely due to the test set not having many "en face" frames (Sec. 4.4). In en face frames, we expect depths to provide more of an advantage because the height of the folds would be clearly seen. By contrast, "down the barrel" frames present each fold and its adjacent pockets as being at a similar distance to the camera, so they do not provide much information to detect folds. For "down the barrel" frames, we expect the surface normals and curvatures, the derivatives of the depths, to provide a significant advantage over RGB-only input, as these features vary more between folds and the rest of the colon. Evidence for this can be found in [10], in which a proposed method for detecting folds uses information derived from curvature.

The reduced performance with regularization losses compared to supervised loss only is likely a result of the regularizations being designed for general-purpose video segmentation. The regularization losses, taken from [14] and [13], were tested on the PASCAL VOC 2012 dataset [2]. This dataset consists of images taken by the general public, meaning that they tend to have much greater diversity in colors, shapes, and textures than colonoscopic video. As a result, these regularization losses are not calibrated to the domain of colon video, so they do not provide an improvement. A regularization tailored to the colon (e.g., using the fact that fold regions must have a convex principal curvature) would be expected to provide an improvement over the supervised loss alone.

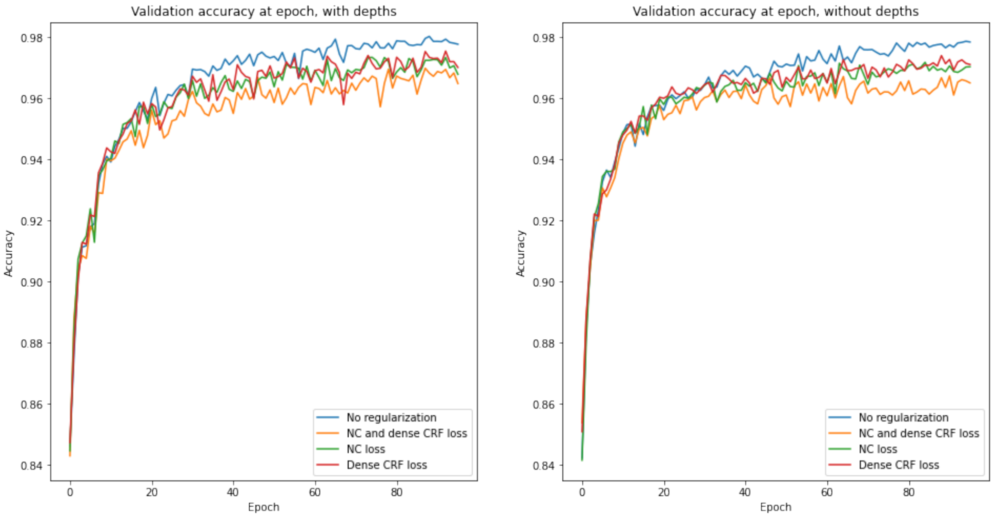


Figure 6: Validation accuracy at training epoch with (left) and without (right) depth inputs. We can see that the configuration with no regularization losses performs best, then the configurations with either NC or dense CRF losses, then both losses together perform worst.

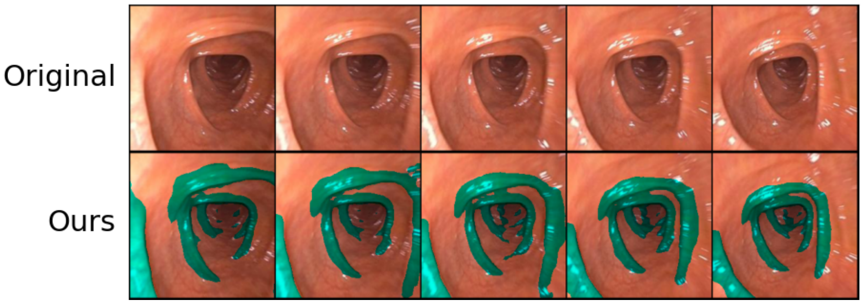


Figure 7: Results on a sequence of 5 consecutive video frames. These are frames 41-45 of Video 1 in the supplement, also linked here: `https://drive.google.com/drive/folders/1m1mgsnbj4MBaPA21sM3-TgEa73ov4ZLg?usp=drive_link`.

6.4 Feature consistency

We see qualitatively in Fig. 7 that our method is able to detect the same folds consistently and produce consistent segmentations over a sequence of 5 consecutive video frames. These are frames 41-45 of the 204 frames in the sequence linked in section 6.2.1 and in the caption of Fig. 7.

7 Limitations

One limitation in our method is that it does not distinguish between adjacent folds. This is not "incorrect" behavior per se, as the network is trained to mark "fold" pixels, but when the haustrum (pocket) between two folds is not visible, the network will mark a single "fold" region. In the future, we intend to address this problem by modifying this work to perform panoptic or instance segmentation, that is, marking each "fold" pixel as belonging to a specific fold.

In addition, the fact that folds often have strong specular reflections has led to a problem where areas of specular reflection are sometimes marked as folds.

8 Future Work

In addition to per-pixel depths, ColDE [16] also produces estimates of the surface normal vector at each pixel. In the future, we plan to incorporate these features into our method, as mentioned in Sec. 6.3. In addition, we plan to train and evaluate our method compared to FoldIt on a set of only "en face" frames, in order to better study the effect of depth input. Some of these frames already exist in the training set, but we also plan to add more "en face" frames to the training set and retrain to avoid the possibility of underfitting to "down the barrel" frames only.

Disclosures

This work was supported by Olympus Corporation, Tokyo, Japan. The authors have no relevant financial or non-financial interests to disclose.

References

[1] Ahn SB, Han DS, Bae JH, Byun TJ, Kim JP, Eun CS (2012) The miss rate for colorectal adenoma determined by quality-adjusted, back-to-back colonoscopies. Gut and liver 6(1):64

[2] Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2012) The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html

[3] Huizinga JD, Pervez M, Nirmalathasan S, Chen JH (2021) Characterization of haustral activity in the human colon. Am J Physiology-Gastrointestinal and Liver Physiology 320(6):G1067–G1080. https://doi.org/10.1152/ajpgi.00063.2021, pMID: 33909507

[4] Levin TR, Corley DA, Jensen CD, Schottinger JE, Quinn VP, Zauber AG, Lee JK, Zhao WK, Udaltsova N, Ghai NR, et al (2018) Effects of organized colorectal cancer screening on cancer incidence and mortality in a large community-based population. Gastroenterology 155(5):1383–1391

[5] Levine JS, Ahnen DJ (2006) Adenomatous polyps of the colon. N Engl J Med 355(24):2551–2557. https://doi.org/10.1056/NEJMcp063038, pMID: 17167138

[6] Lin D, Dai J, Jia J, He K, Sun J (2016) Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3159–3167

[7] Ma R, Wang R, Pizer S, Rosenman J, McGill SK, Frahm JM (2019) Real-time 3d reconstruction of colonoscopic surfaces for determining missing regions. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, pp 573–582

[8] Ma R, McGill SK, Wang R, Rosenman JG, Frahm JM, Zhang Y, Pizer SM (2021) Colon10k: A benchmark for place recognition in colonoscopy. 2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI) pp 1279–1283

[9] Mathew S, Nadeem S, Kaufman A (2021) Foldit: Haustral folds detection and segmentation in colonoscopy videos. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, pp 221–230

[10] Nadeem S, Marino J, Gu X, Kaufman AE (2017) Corresponding supine and prone colon visualization using eigenfunction analysis and fold modeling. IEEE Transactions on Visualization and Computer Graphics 23:751–760

[11] Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC (2018) Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4510–4520

[12] Siegel RL, Miller KD, Goding Sauer A, Fedewa SA, Butterly LF, Anderson JC, Cercek A, Smith RA, Jemal A (2020) Colorectal cancer statistics, 2020. CA: A Cancer Journal for Clinicians 70(3):145–164. https://doi.org/https://doi.org/10.3322/caac.21601

[13] Tang M, Djelouah A, Perazzi F, Boykov Y, Schroers C (2018) Normalized cut loss for weakly-supervised cnn segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1818–1827

[14] Tang M, Perazzi F, Djelouah A, Ben Ayed I, Schroers C, Boykov Y (2018) On regularized losses for weakly-supervised cnn segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp

507–522

[15] US Preventive Services Task Force (2021) Screening for Colorectal Cancer: US Preventive Services Task Force Recommendation Statement. JAMA 325(19):1965–1977. https://doi.org/10.1001/jama.2021.6238

[16] Zhang Y, Frahm JM, Ehrenstein S, McGill SK, Rosenman JG, Wang S, Pizer SM (2021) ColDE: A depth estimation framework for colonoscopy reconstruction. arXiv preprint https://arxiv.org/abs/arXiv:2111.10371 [eess.IV]

[17] Zhao S, Wang S, Pan P, Xia T, Chang X, Yang X, Guo L, Meng Q, Yang F, Qian W, et al (2019) Magnitude, risk factors, and factors associated with adenoma miss rate of tandem colonoscopy: a systematic review and meta-analysis. Gastroenterology 156(6):1661–1674

[18] Zhu JY, Park T, Isola P, Efros AA (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE international conference on computer vision, pp 2223–2232

Appendices

A System Diagrams

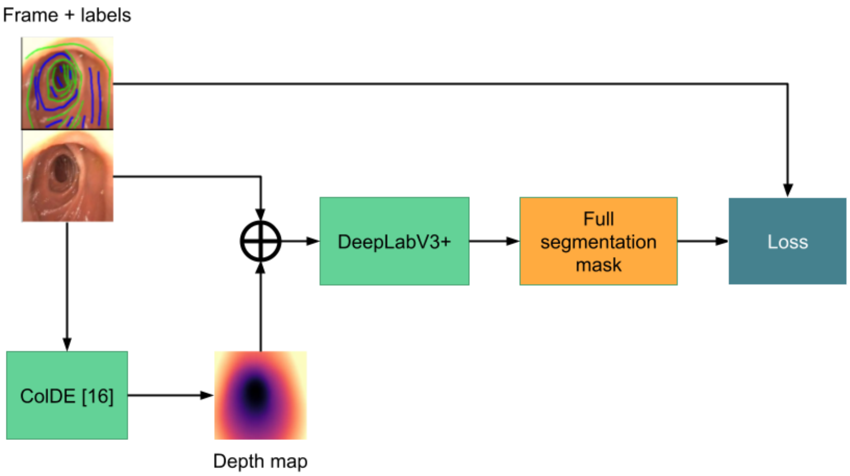


Figure 8: Diagram of the full system in training. Colonoscopy video frames and their computed depth maps are inputs to the network, and the loss is computed according to Eq. 1 using the frame, network output, and scribble labels.

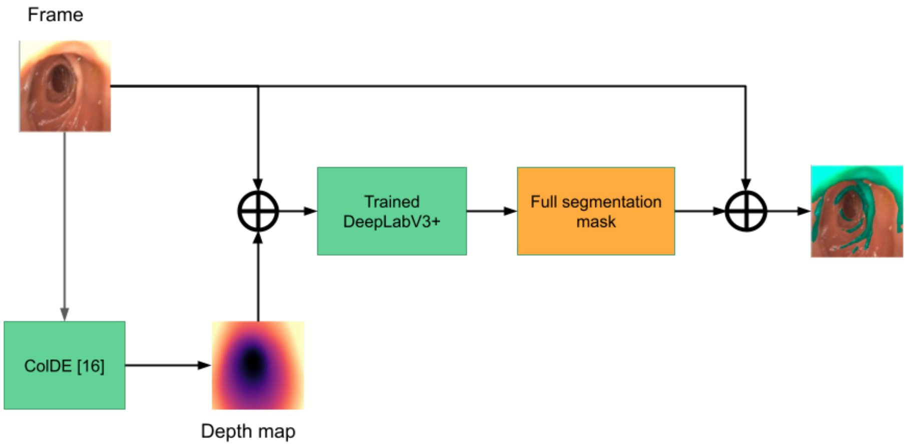


Figure 9: Diagram of the full system at inference time. There are no ground-truth scribbles; the system computes a segmentation mask for all frame pixels from the frame and its computed depth map.