Real-time Feedback for Colonoscopy in a Multi-center Clinical Trial

Wallapak Tavanapong
Department of Computer Science
Iowa State University
Ames, IA 50011, USA
tavanapo@iastate.edu

JungHwan Oh
Department of Computer Science and Engineering
University of North Texas
Denton, TX 76203
junghwan.oh@unt.edu

Gavin Kijkul
Department of Computer Science
Iowa State University
Ames, IA 50011, USA
gkijkul@iastate.edu

Jacob Pratt
Department of Computer Science
Iowa State University
Ames, IA 50011, USA
jrpratt@iastate.edu

Johnny Wong
Department of Computer Science
Iowa State University
Ames, IA 50011, USA
wong@iastate.edu

Piet C. deGroen
Division of Gastroenterology
Hepatology and Nutrition
University of Minnesota
Minnesota, MN 55455, USA
degroen@umn.edu

***Abstract* — We report the technical challenges, solutions, and lessons learned from deploying real-time feedback systems in three hospitals as part of a multi-center controlled clinical trial to improve quality of colonoscopy. Previous clinical trials were conducted in one center. The technical challenges for our multi-center clinical trial include 1) reducing additional work by the endoscopists to utilize real-time feedback, 2) handling different colonoscopy practices at different hospitals, and 3) training an effective CNN-based classification model with a large variety of patterns of data in day-to-day colonoscopy practice. We report performance of our real-time systems over a period of 20 weeks at each hospital. We conclude that CNN-based classification can achieve very good performance in real-world deployment when trained with high quality data.**

Keywords— Multi-center clinical trial, Real-time feedback of colonoscopy quality, Convolution Neural Network (CNN)

I. Introduction

Colorectal cancer (CRC), despite being a preventable cancer, is still expected to cause about 53,200 deaths in the U.S. in 2020 [1]. Colonoscopy allows for detailed examination of the entire colon and removal of all premalignant lesions during the procedure, the latter typically done during withdrawal of the endoscope after reaching maximum intubation. Colonoscopy is readily available in the U.S. and covered by insurance. Given these facts, why is CRC prevention by colonoscopy lower than expected? Currently, the dominant explanations for the continued incidence of CRC are endoscopist-related factors, such as choice of suboptimal equipment, not removing remaining debris, not reaching the cecum, fast withdrawal, no effort at inspection of areas behind folds and angulations, and an inadequate polyp removal technique. Recent research examining 15 years of data at a private ambulatory surgery center with 80 endoscopists practicing high quality examination of the colon shows a prevention of 67% of CRC compared to the SEER-18 population [2]. The high-quality examination protocol with an acronym “CLEAR” was used to have the endoscopists (1) **C**lean the colon, (2) **L**ook **E**verywhere, and (3) perform complete **A**bnormality **R**emoval.

Since 2003, we have been developing software (Endoscopic Multimedia Information System or EMIS) for automated measurements of quality and feedback during colonoscopy. In a single-center controlled clinical trial in 2012, the quality of colonoscopy performed by ten third-year GI trainees with feedback from EMIS improved significantly [3]. EMIS measured multiple intra-procedure quality metrics (e.g., clear withdrawal time without blurry frames, amount of stool during insertion and withdrawal, “spiral score”, and Boston Bowel Preparation Scale scores), but the provided feedback consisted only of the “spiral score”, or in simple terms how well the endoscopist tried to look everywhere. The objective measurements were then combined into a single automated quality score which was shown by domain experts to correlate with quality of colonoscopy [4]. Since then, there were two reports of real-time feedback systems for colonoscopy in single-center clinical trials [5], [6]. To the best of our knowledge, none have been reported for a multi-center clinical trial. We are nearing the end of the first phase of the deployment of EMIS (version 6) and EMIS-Deep (EMIS extension utilizing CNN) in a multi-center clinical trial at University of Washington Medical Center, University of Minnesota, and Johns Hopkins University. The deployment presented several technical challenges that we had to overcome.

Our contributions include 1) presenting the technical challenges encountered and their corresponding solutions in the deployment of real-time measurements and feedback for endoscopy in a multi-center clinical trial, 2) reporting the effectiveness of our real-time systems, EMIS (version 6) and EMIS-Deep, over a period of 20 weeks, and 3) showing that

This work was supported in part by the National Institutes of Health Grant No. #1R01DK106130-01A1. Findings, opinions, and conclusions expressed in this paper do not necessarily reflect the view of the funding agency.

TABLE I: Existing real-time detection and feedback for colonoscopy

Publication	Sub-system	Task	Feature	Classifier	Deep Learning Architecture	Training Data		Validation Data		Test Data		Image Resolution	# doctors in a clinical trial	# patients in a clinical trial	Average processing frame rate (fps)
Wang et al. 2019 [5]		Polyp Detection	Extracted from CNN trained from scratch	CNN	SegNet	5,545 images		27,725 images total		N/A		480x360	8 endoscopists: 2 senior (>20,000 procedures), 2 midlevel (3,000 - 10,000 procedures), 4 junior (100-500 procedures)	536 patients (control) 522 patients (feedback)	25
						3,646 PF	1,911 NPF	6,153 PF	21,572 NPF						
								138 unique polyps from 138 polyp videos (with all PF) totaling 40.61 min. (FR N/A)	54 full-length videos with all NPF totaling 714.99 min. (FR N/A)						
Su et al. 2019 [6]	Model B	Cecum Classification	Alex Net Features	2 FC layer NN	NN	2,825 images		933 images		949 images		64x64	4 endoscopists (5,000-8,000 procedures)	315 patients (control) 308 patients (feedback)	25
	Model E	Vitro/Vivo Classification	Pre-trained features (CNN)	1 FC layer NN	NN	4,086 images		1,118 images		1,280 images		32x32			
	Model BP	BBPS Score Classification	ZFNet features	2 FC layer NN	NN	6,402 images		918 images		1,000 images		64x64			
	Model PD	Polyp Detection	Learned features	CNN	Darknet-19 & YOLO V2	2,638 images		712 images		751 images		416x416			
	Model S	Blurry/Clear Classification	MobileNets features	1 FC layer NN	NN	1,654 images		492 images		N/A		32x32			
Byrne et al. 2019 [11]		Classification of polyp types: Adenoma, Hyperplastic, No-polyp, Unsuitable Classification	Extracted from CNN trained from scratch	CNN	Inception	60,089 images		40 videos total (length and FR N/A)		125 videos total (length and FR N/A)		N/A	No clinical trial	N/A	20
						49,273 PF (17,426 hyperplastic, 31,847 adenoma)	10,816 NPF	38 polyp videos	2 non-polyp videos	51 hyperplastic, 74 adenomas videos	0 non-polyp videos				
Urban et al. 2018 [10]		Localization of polyps	Fine-tuned features (pre-trained on ImageNet)	CNN	ResNet50 & YOLO	8,641 images total (7-fold cross validation)				46,277 images	~455,000 images from 11 videos	224x224, 480x480	No clinical trial	N/A	98
						4,088 PF		4,553 NPF		13,964 PF and 32,313 NPF					
Riegler et al. 2017 [9]	Detection sub-system	Abnormality Detection	Hand-crafted features	ranked list KNN	None	18,781 images (leave-one-out cross-validation)						1920x1080, 856x480, 712x480	No clinical trial	N/A	192
	Localization sub-system	Abnormality Localization	Hand-crafted shape features	PF detection and polyp localization	None										
Wang et al. 2015 [8]		Classification of polyp frames & localization of polyps	Hand-crafted edge cross-section features	Rule-based generalized from SVM classifier	None	~287,712 images (leave-one-out cross-validation)				~1,906,092 images		720x480	No clinical trial	N/A	10
Stanek et al. 2012 [7]		Detection of the start and end frames of an endoscopic procedure	Hand-crafted color and temporal features	Rule-based	None	N/A		N/A		> 265 million images (2464 h of video)		720x480	No clinical trial	N/A	125

NN: Neural Network; **PF:** Polyp Frame; **NPF:** Non-Polyp Frame; **FC** layer: Fully connected layer; **FR:** video frame rate; **KNN:** K-Nearest Neighbor classification

given representative training data, a CNN-based system can be very effective in real-world deployment.

The rest of this paper is as follows. Section II presents related work on real-time measurement and feedback for colonoscopy. Section III describes the challenges and our solutions. Section IV presents the performance evaluation of EMIS (version 6) and EMIS-Deep over a 20 week period, covering over 2,000 procedures. We provide the conclusion and discussion of the future work in Section V.

II. Related Work

Table I shows a summary of existing works on real-time systems for colonoscopy. Given that several methods are already capable of running in real-time, we omit other methods that were not reported to be capable of running in (near) real-time.

A. Real-time Feedback for Colonoscopy in Single-Center Controlled Clinical Trials

In the aforementioned clinical trials, audio and an extra monitor were used to provide feedback. The results of both trials show a higher adenoma detection rate on procedures with feedback. The system by Wang et al. provided a sound prompt when a polyp appearance was detected [5]. The endoscopist was asked to look at the main monitor until hearing the sound prompt, at which point the endoscopist would look at the second monitor for the detected polyp location displayed in a hollow blue tracing box. The feedback in the trial conducted by Su et al. [6] included audio prompts when continuous blurry or unstable frames were detected. The detected polyp location was displayed on a second monitor. Both systems used deep-learning models. Table I shows the technical differences between the two systems as well as other real-time capable systems for colonoscopy not in a clinical trial. Su et al. utilized five neural-network models, four of which used features extracted from existing pre-trained models as input to shallow fully-connected neural networks [6]. Model B classified cecum and non-cecum images to identify the beginning of the withdrawal phase of a colonoscopy. Model E classified in vitro and vivo frames to identify the end of the withdrawal phase (end of the procedure). Model BP output four probabilities for an image belonging to each Boston Bowel Preparation Scale score (BBPS). Model PD classified whether an image shows a polyp appearance. Finally, model S determined withdrawal stability by classifying blurry and non-blurry frames, as well as computing the similarity between two subsequent frames. The models by Su et al. were trained, validated, and tested on small datasets. Aside from model PD, none of the models' clinical trial performance was reported.

B. *Real-time Capable Polyp Detection Methods for Colonoscopy*

This category includes methods reported to being capable of running in (near) real-time but were not mentioned as part of any clinical trial. Earlier works in this category used hand-crafted features while recent studies used deep-learning based methods. These methods were shown to perform well on limited datasets, but performance on full-length colonoscopy procedures to simulate the application in routine colonoscopy screening were not reported.

Stanek et al. proposed real-time detection of the start and end of an endoscopic procedure using hand-crafted features [7]. Polyp-Alert by Wang et al. [8] used edge-cross-section features and a rule-based classifier to detect edges that make the contour of a polyp. It tracked the same polyp appearing in nearby frames and provided a visual marker on the detected polyp edge up to ten times a second. Tests on approximately 18 hours of videos showed that Polyp-Alert returned feedback with a recall of unique polyps of 97.7% and on average about 0.86 min. of false alert per 20 min. procedure video. There are no other publications which report results of test performances on millions of frames as in this work. Riegler et al. proposed an end-to-end multimedia system for automatic disease detection, and visualization in GI tract procedures [9]. Urban et al. trained different CNNs on two different datasets for polyp detection and localization [10]. Polyp localization used a variation of the "You Only Look Once" (YOLO) method, which predicts bounding boxes for multiple polyps within an image. The best performing model for both tasks was pre-trained using ResNet50 that achieved 96.1% accuracy. Byrne et al. proposed a CNN based on the inception network architecture to detect different types of polyps under Narrow Band Imaging [11]. They achieved 94% per frame accuracy in classification of adenoma and hyperplastic polyps. These techniques were tested on different datasets. Hence, we cannot compare their effectiveness directly.

III. EMIS: Real-time Feedback Systems in a Multi-center Clinical Trial

A. *Design Goals for Real-time Feedback in Routine Practice*

Our goals are different from real-time feedback systems used in previously reported single-center clinical trials [5], [6]. Our goals are as follows. 1) Minimum change of the current endoscopy practice to utilize real-time feedback. We do not introduce an extra monitor nor additional sound devices for feedback. All feedback is visual on the same monitor without overlapping with the colon mucosa area on the monitor. 2) Ability to handle differences in colonoscopy practice at different hospitals. 3) System switched on and ready at least 12 hours/day during out-patient practice hours and at least 5 days/week in all the sites and recognition of full-length endoscopy procedures performed by any number of participating endoscopists instead of by a fixed set of endoscopists as in existing trials. 4) No need for personnel to manually indicate the start or the end of a procedure.

B. *Extension of EMIS to EMIS-Deep*

EMIS runs on an off-the-shelf PC equipped with a video capturing card that takes as input a composite signal from an endoscope video processor, configured to always disable any patient- or physician-related information, i.e., endoscope text features are permanently disabled. We achieved the first design goal by using a hardware overlay that overlays the real-time feedback from EMIS ─non-black pixels only─ with the original signal from the endoscope processor, as developed in the previous trial [3]. Feedback consists of text in the lower right corner of the screen displaying EMIS INSIDE when a procedure is detected, meaning the endoscope is inside the patient, and EMIS ON when the endoscope is outside the patient. In our next phase of deployment, we will also display feedback for spiral score, the number of detected retroflexion frames, colon preparation, and the blurriness of images. Some feedback will only be displayed when the measured quality is below a preset threshold value.

EMIS is built on top of our middleware SAPPHIRE designed for modularity, scalability, configurability, collaboration among multiple developers, and ease of extension [12]. SAPPHIRE comes with several modules for basic tasks, for instance, video capture, reading and writing video files, and displaying frames overlaid with analysis results from other modules on screen. EMIS version 4 has been published in previous work and used in a previous clinical trial. It has the following capabilities: 1) Detection of the begin and end frames of an endoscopy; 2) Classification of informative frames and non-informative frames; 3) Measuring the amount of stool-colored pixels for each frame; 4) Measuring the spiral score; and 5) Calculating various objective measurements of colonoscopy quality.

The last two design goals proved difficult to achieve with EMIS version 4 due to the variety of endoscope use patterns among different hospitals. EMIS version 6 includes a revised automated cropping module for Olympus scopes. After fine-tuning threshold values for detection of an endoscopic procedure, EMIS version 6 has been running well in two hospitals. However, we did not find suitable threshold values for the remaining hospital since there are many more patterns of outside-the-patient images beyond what our hand-crafted features were designed for. To address this issue, we made a decision between 1) developing additional hand-crafted features to upgrade the classifier; or 2) using deep-learning for feature representation and classification. We chose the latter approach given fast development time and ease of adaptability. We had some reservations related to this choice because existing CNN-based medical analyses were tested only on limited datasets (i.e., fewer than 23K images as shown in Table I). In contrast, this clinical trial requires the processing of millions of images.

To address this challenge, we developed EMIS-Deep, an integration framework that enables shared memory communication between EMIS and deep-learning modules. Fig. 1 shows the overview of EMIS-Deep. The framework also provides the flexibility to switch to different deep-learning toolkits without long development time. The disadvantage is the cost of copying data into and from the shared memory. We minimize this cost as much as possible. EMIS-Deep is currently used for detecting the frame numbers of the start and the end of an endoscopic procedure. We can add other types of image

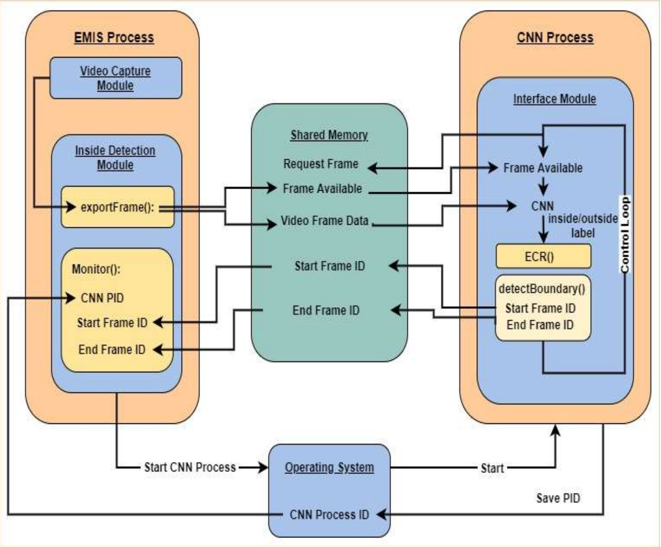

Fig. 1. Overview of EMIS-Deep for the start and end frame numbers of an endoscopy procedure; PID denotes the CNN process ID.

classification as needed. We implemented two new modules in C/C++ as dynamic linked libraries (DLL). Each module runs in its own thread. One module runs inside the EMIS process and the other module runs inside a CNN process for CNN-based classification (Fig. 1). In our current implementation, the CNN process runs an executable file generated from PyInstaller that wraps our Python program calling Google TensorFlow for image classification using CNN.

The new module in the EMIS process initializes the shared memory fields for communication with the CNN process. For every frame routed through the middleware from the video capture module, the new module checks the shared memory field whether the CNN process requests any frame. If so, the module crops and resizes the current frame and copies the resized frame data to the shared memory data field. It also updates the shared memory fields for availability of the frame data as well as the frame ID. The frame ID is unique for each frame and it is assigned by the video capture module which increments the frame ID by one for each frame it sends out through the middleware. The thread monitors the shared memory fields for the start frame ID (SID), the first inside-the-patient frame of a procedure, and the end frame ID (EID), the first outside-the-patient frame, respectively. Finally, the thread also monitors whether the CNN process is still active and forks the new CNN process if it becomes inactive to ensure the robustness of the program.

The CNN process initializes the CNN architecture, modeled after VGG16 [13], and loads the pre-trained model that classifies whether a frame is “inside-the-patient” (i.e., frame shows intestinal mucosa) or “outside-the-patient”(i.e., frame shows non-intestinal mucosa such as endoscopy room). It then loads the interface module implemented as a C DLL with functions that are callable from Python, and begins a control loop. The control loop runs for the duration of the process. In the control loop, for every *t* frames, it calls the Python-C interface through the interface module to update the shared memory field to request a frame from EMIS and waits with time-out for the frame data, which is then input to the trained CNN model. The control loop also periodically saves values of important state variables to the shared memory so that it can resume from those states when restarted by the EMIS process.

```
detectBoundary (predictedLabel, bndFrameID, enterThres, exitThres)
1:  if predictedLabel == previousPredictedLabel return
2:  if predictedLabel == 0:   // frame predicted as inside-the-patient
3:      if videoOutsidePatient is true:   // camera is outside-the-patient
4:           count ++
5:           if count  >= enterThres:   // find enough “inside” frames
6:               SID = bndFrameID    // find the start frame ID
7:               count = 0
8:               videoOutsidePatient = false
9:  else   // current frame predicted as outside
10:     if videoOutsidePatient is false: // camera is inside the patient
11:          count ++
12:          if count >= exitThres:  // find enough “outside” frames
13:              EID = bndFrameID  // find the end frame ID
14:              count =0
15:              videoOutsidePatient = true
```

(a) Function to find the boundary frame ID, which is either SID or EID

```
errorCorrection(meanRed, meanBlue, CNNPr, ECR_thres, $\mu$, $std$ )
label = 1;          // outside-the-patient frame
redOverBlue = smooth(meanRed, 15) / smooth(meanBlue, 15)
redBluePr = NormContDist(redOverBlue, $\mu$, $std$)
smthCNN = smooth(CNNPr,15)
jointPr = smthCNN * redBluePr
denPr = (1- smthCNN) * (1- redBluePr)
finalPr = jointPr / (jointPr + denPr)
if finalPr > ECR_thres:
    label = 0;       // inside-the-patient frame
return slidingfilter(label)
```

(b) Function to correct the CNN predicted results

Fig. 2. Key functions in the CNN process

Ideally, if the CNN always predicts the label of each frame correctly, we can simply rely on the CNN predicted results to find the values of SID and EID as outlined in Fig. 2(a). That is, if the previously and the newly predicted labels are the same, no further computation is needed (Line 1). However, if they are different, we potentially find the boundary frame separating inside- and outside-the-patient frames. We save the boundary frame ID in the *bndFrameID* variable and call the **detectBoundary** function to set the value of the global variable *videoOutsidePatient.* This global variable is initialized to true since the camera is initially outside the patient. The variable is set to false in Line 8 if its current value is true and a sufficient number of inside-the-patient frames (i.e., *count* >= *enterThres*) are seen. The value of *videoOutsidePatient* is set to true in Line 15 if its current value is false (i.e., the camera was inside the patient) and a sufficient number of outside-the-patient frames are seen (i.e., *count*>= *exitThres* value). For a perfect CNN, the values of *exitThres* and *enterThres* can simply be one; however, as classification is often not perfect, we use higher threshold values.

C. Dealing with Incorrect CNN Prediction Results

The correctness of CNN classification depends on whether the data used for training are representative of the data seen in deployment. If this is not the case, many inside-the-patient frames may be missing, a single procedure may be cut into multiple video files, or many outside-the-patient frames may be considered as part of the procedure. We optimize CNN classification by 1) using our error correction function outlined in Fig. 2(b) before using the corrected value as input to Fig. 2(a) and 2) development of a high quality, representative training dataset.

C.1 CNN Error Correction: Fig. 2(b) shows the error correction function based on an observation that inside-the-patient frames have a significant amount of red color compared to blue color which outside frames do not have. The **smooth** function in Fig. 2(b) implements a linear smoother to output an average of 15 past values including the current value. We apply the function to the mean normalized red value (*meanRed*), mean normalized blue value (*meanBlue*), and *CNNPr*---the probability of the current frame belonging to the inside-the-patient class determined by CNN. In Line 3, the output of **NormContDist**, a normal continuous distribution function with the mean μ and the standard deviation *std,* is used to correct the frame label in Lines 4-9. The label is passed to the **slidingfilter** function that adds the label to the sliding window. Only when the input label agrees with the two previous labels stored in the sliding window is the label output, otherwise, it is suppressed, and the previous output's label is output.

C.2 Training Datasets: CNN performance depends greatly on the quality of the training dataset. Initially, we thought that implementation of inside- and outside-the-patient frame classification using CNN would be easy, until we encountered large variations in appearance of outside-the-patient frames in routine colonoscopy screening. Gradually adding new mis-classified frames encountered to the training data and retraining the model created a significant class imbalance in the training dataset, resulting in poor performance. Synthetic data augmentation that generates minor variations on the dataset would not work well due to major differences in the patterns. To overcome all these challenges, we created a new, balanced training dataset that included various patterns seen in the sample population data for model training and validation. This resulted in the best performing model. Fig. 3 shows the variety of patterns in the training dataset that resulted in the best performance in our studies. In this figure, one image represents a pattern---a group of images in the same cluster in the feature space. There are 20 patterns (red color) for the inside-the-patient class and 28 patterns for the outside-the-patient class although the total number of images in each class is the same. A larger image in Fig. 3 represents a cluster with more images. The distances among the images reflect how far they are in the summary feature space with two dimensions, which are the first two components from Principal Component Analysis applied to the features extracted from the image using the original pre-trained VGG16 model. Using the model trained on this dataset, the CNN accuracy on the validation dataset of 30,496 images is 98.8%. The validation dataset is class balanced and does not overlap with the training dataset.

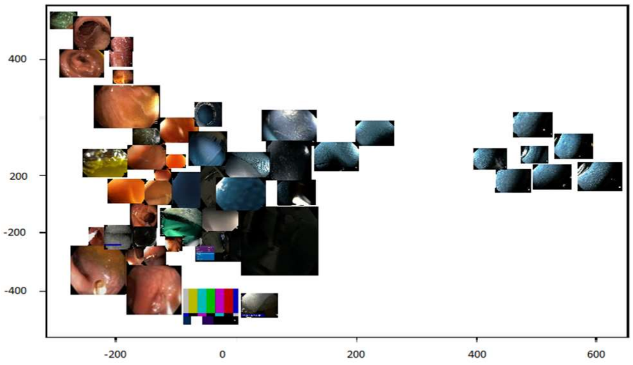

Fig. 3. Forty-eight different patterns in the training dataset of 38,804 images of the same pixel resolution; each image in this figure represents one pattern (cluster) obtained by hierarchical clustering. The image size roughly tells the size of the cluster; a larger image size in this figure means the cluster has more images.

Does this training dataset represent the population data? Currently, there are no quantitative measures designed to answer this question. Such a measure would be useful so that a representative training dataset can be found semi-automatically without time-consuming manual effort. Currently, one must examine the classification performance during the deployment and retrain the model until the desired performance in real-world deployment is reached.

TABLE II. CNN architecture configuration; the size-in and size-out are described by rows × cols × #nodes. The kernel configuration for the convolutional layers (conv1-5) is specified as rows × cols × #nodes × #filters, stride. For the max pooling layers (pool1-4), we list rows × cols × # nodes, stride. Relu activation function and Adam optimizer were used.

Layer	Size-in	Size-out	Kernel
Conv1	64x64x3	64x64x16	3x3x3x16,1
Pool1	64x64x16	32x32x16	2x2x16,2
Conv2	32x32x16	32x32x32	3x3x16x32,1
Pool2	32x32x32	16x16x32	2x2x32,2
Conv3	16x16x32	16x16x64	3x3x32x64,1
Pool3	16x16x64	8x8x64	2x2x64,2
Conv4	8x8x64	8x8x128	3x3x64x128,1
Pool4	8x8x128	4x4x128	2x2x128,2
Conv5	4x4x128	1x1x256	4x4x128x256,1
FC	256	256	
Softmax	256	2	

IV. Performance Evaluation

As described in Section III, all the collected videos do not contain any information about patients nor endoscopists. We received the formal approvals for the first phase of the deployment from all institutions to capture de-identifiable procedures, one video file per procedure, for evaluation and calibration of software parameters. Our CNN architecture is a smaller version of VGG16 with the architecture configuration listed in Table II.

Table III shows the performance in five rooms of the three hospitals. The software was setup to run from 6:00am to midnight daily. However, the Olympus endoscope video processor in each room was turned on at different times, typically from 7am-6pm during weekdays and was turned off after the last procedure of the day. The composite video resolution was 720x480 and the frame rate was 29.97 fps. In one room, our system analyzed over 11 hours' worth of video (about 1.18M frames) per day 5 days a week for more than 20 weeks. There may not be procedures every day and the endoscope processor may not be turned on as early as 6am. Nevertheless, to the best of our knowledge, this is the largest reported test of real-time systems in endoscopy and of CNN-based systems during routine colonoscopy. Our goal was to ensure that our software is sufficiently accurate and reliable in day-to-day routine endoscopy practice before we use it for real-time feedback of colonoscopy quality. This is to avoid irrelevant factors (e.g., significant amount of false predictions) from affecting the measured quality metrics. Rooms W1 and W2 denote two endoscopy rooms from the same hospital; M1 denotes one room in the second hospital. E denotes one room in a third hospital that is the only place using EMIS-Deep. E-2020 denotes that the system used the CNN model trained on the latest, most representative training dataset with 48 patterns discussed in Section III.C.2 and summarized in Fig. 3, the error correction algorithm outlined in Fig. 2(b), and the parameter values indicated in Table IV. In contrast, E-2019 denotes the system used prior CNN models without the error correction algorithm and without the best representative training dataset.

TABLE III. Performance in different rooms in all three hospitals

Rooms	Duration	#Actual procedures*	#Problem cases	Recall	Failed ratio
M1	25 weeks	493	71	0.92	0.14
W1	20 weeks	466	10	0.98	0.02
W2	20 weeks	410	18	0.95	0.04
E - 2019	18 weeks	715	86	0.87	0.12
E - 2020	7 weeks	269	11	0.97	0.03

*Full-length endoscopy procedures; failed ratio=the ratio of #problem cases to #actual procedures

The number of problem cases reported in Table III includes cases where procedures were split into multiple video files, video files of cases with more than 1 minute of outside-the-patient frames, cases missing a significant amount of procedure content, and multiple cases being incorrectly combined into one video file. Recall measures the number of correctly detected procedures (non-problem cases) to the number of actual procedures. Table III shows high recall values (0.95-0.98) and very small failed ratios (2-4%) for W1, W2, and E-2020 (EMIS-Deep). The results for E-2019 are not as good, as the CNN of E-2019 was trained on a worse quality training dataset. M1 used EMIS, just as W1 and W2, but had different parameter values. Most failed cases are videos containing non-procedure images.

TABLE IV. Parameters and values for EMIS-Deep for E-2020

Parameter for room E-2020	**Value**
t: frame interval in the CNN process	6
Resized image size (pixels x pixels)	64x64
N(μ,std): Normal distribution for red over blue ratio	N(1.15, 0.15)
enterThres (for detectBoundary)	5
exitThresh (for detectBoundary)	25
ECR_Thres (for errorCorrection)	0.95
Training hyper-parameters chosen empirically: learning rate (0.001), weight decay (0.01), batch size (512)	

V. CONCLUSION AND FUTURE WORK

We presented the challenges and solutions in deploying real-time measurements and feedback in a multi-center clinical trial. We reported the effectiveness of the real-time systems observed over a 20 week period. We show that the effectiveness of a CNN-based system is highly dependent on how representative the training data is. The software used in this paper is currently not publicly available. As future work, we will investigate dependability measures of the training data. Less dependable training data means more potential failures. We will utilize the dependability measures to develop effective semi-automated solutions to collect training datasets to train effective CNN models. The measures should be independent from deep-learning models used and should be comparable among different training datasets. Second, we show that EMIS-Deep, our loosely integrated framework, using shared memory works well for integrating EMIS and CNN code using TensorFlow. EMIS-Deep enables flexible integration of existing code and new code using state-of-the-art deep-learning toolkits. The framework is easily extensible to include deep-learning based detection of other objects of interest.

CONFLICT OF INTEREST AND AKNOWLEDGMENTS

Tavanapong, Wong, and Oh have equity interest and management roles in EndoMetric Corp. Dr. de Groen serves on the Scientific Advisory Board of EndoMetric Corp. We thank Jaber Salem for his help in collection of performance data.

REFERENCES

[1] American Cancer Society, “Colorectum Cancer Statistics,” 2020. [Online]. Available: https://cancerstatisticscenter.cancer.org/?_ga=2.95264916.902125337.1581945528-1873365005.1581945528#!/cancer-site/Colorectum.

[2] S. Xirasagar, Y. Wu, M.-H. Tsai, J. Zhang, S. Chiodini, and P. C. de Groen, “Colorectal cancer prevention by a CLEAR principles-based colonoscopy protocol: an observational study,” *Gastrointest Endosc*, pp. S0016-5107(19)32494–0, Dec. 2019.

[3] N. Srinivasan *et al.*, “Real-time Feedback Improves the Quality of Colonoscopy by Trainees: A Controlled Clinical Trial: ACG/AstraZeneca Fellow Award: 1492,” *American Journal of Gastroenterology*, vol. 107, p. S596, 2012.

[4] F. Enders, W. Tavanapong, M. J. Szewczynski, J. Oh, J. Wong, and P. De Groen, “Tu1018 Objective Evaluation of Colonoscopy: Development and Validation of an Automated Score,” *Gastroenterology*, vol. 146, no. 5, p. S-728, May 2014.

[5] P. Wang *et al.*, “Real-time automatic detection system increases colonoscopic polyp and adenoma detection rates: a prospective randomised controlled study.,” *Gut*, 2019.

[6] J.-R. Su *et al.*, “Impact of a real-time automatic quality control system on colorectal polyp and adenoma detection: a prospective randomized controlled study (with videos),” *Gastrointestinal Endoscopy*, 2019.

[7] S. R. Stanek, W. Tavanapong, J. Wong, J. H. Oh, and P. C. de Groen, “Automatic real-time detection of endoscopic procedures using temporal features,” *Computer Methods and Programs in Biomedicine*, vol. 108, no. 2, pp. 524–535, 2012.

[8] Y. Wang, W. Tavanapong, J. Wong, J. H. Oh, and P. C. de Groen, “Polyp-Alert: Near real-time feedback during colonoscopy,” *Computer Methods and Programs in Biomedicine*, vol. 120, no. 3, Jan. 2015.

[9] M. Riegler *et al.*, “From Annotation to Computer-Aided Diagnosis: Detailed Evaluation of a Medical Multimedia System,” *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, May 2017.

[10] G. Urban *et al.*, “Deep Learning Localizes and Identifies Polyps in Real Time With 96% Accuracy in Screening Colonoscopy.” Gastroenterology, 2018.

[11] M. F. Byrne *et al.*, “Real-time differentiation of adenomatous and hyperplastic diminutive colorectal polyps during analysis of unaltered videos of standard colonoscopy using a deep learning model.” Gut, 2019.

[12] S. R. Stanek *et al.*, “SAPPHIRE middleware and software development kit for medical video analysis,” in *2011 24th International Symposium on Computer-Based Medical Systems (CBMS)*, 2011, pp. 1–6, doi: 10.1109/CBMS.2011.5999145.

[13] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *Computer Science*, 2014.