Indian Institute of Information Technology Guwahati

Department of Computer Science and Engineering

Endoscopy: A Deep Learning Approach

Author:
Amartya Dutta
1701005

Supervisor:
Dr. Ferdous Ahmed Barbhuiya

I would like to dedicate this thesis to my wonderful parents. I would also like to dedicate this to all the doctors and nurses who have been working relentlessly to save our lives.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Amartya Dutta

April 2021

Acknowledgement

I would like to thank my guide Dr. Ferdous Ahmed Barbhuiya for allowing me to choose this problem. It is because of his guidance and support that I have been able to perform properly and efficiently during this work. I would also like to thank Rajat Kanti Bhattacharjee for his contribution by suggesting ideas that were helpful for this work.

Abstract

Endoscopy is a very important procedure in the medical field. It is used to detect almost any diseases associated with the gastrointestinal (GI) tract. The current work attempts to use Deep Learning methods to ensure that such medical procedures can be automated and be used in real-life situations. One of the major concerns that need to be kept in mind is that such methods should not just be able to give results but also ensure that it occurs in real-time. The time duration required for the diagnosis should be as minimal as possible. Keeping the fastness of the desired method in mind, the current work implements the Tiny Darknet model. The Tiny Darknet Model is a reduced version of the Darknet Reference Model which has been proved earlier to have great performance when it comes to speed. Therefore, an attempt to efficiently classify the various medical conditions using the Tiny Darknet has been made. The aim is to be able to achieve similar such speed with the Tiny Darknet while also trying to maximise the accuracy score. Eventually, the Tiny Darknet succeeds in having a high classification speed, achieving up to a maximum speed of about *60fps* while scoring up to *0.76* according to the *MCC*. However, the method lacks in terms of being an accurate model. Therefore, the work investigates further approaches into improving the model's performance using methods such as Ben's Preprocessing. Besides this, the current work also investigates possibilities in exploiting the large amount of unlabeled data using Semi-supervised GAN such that it can be used to improve further performance. As an additional task, we have also generated binary masks for *Polyps*, a very important class of endoscopic images.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Machine learning and its applications in the field of healthcare has come to play an integral role in recent times. The main motivation behind using a machine learning model is to automate the process thereby trying to ensure that human errors can be taken care of while making the process faster. One such line of work involves detecting medical conditions and other lesions in the GI tract. Endoscopy is a very important process that is used to detect almost any disorder related to the GI tract. However, detection of these conditions depends highly upon the experience and the skill of the doctor performing it. Therefore, it may often result in high miss rates, averaging at about 20% [4].
Several innovations have been made in the past that focuses on detecting the various conditions of the GI tract using existing Machine learning methods. Georgakopoulos et al. [6] was one such work that had proposed a novel weakly-supervised learning method based on a Convolutional Neural Network to be used for detecting lesions in the GI tract. However, besides just being able to accurately detect the known medical conditions from the images of the GI tract, the speed at which these detections are made is also of utmost importance. Most of the existing methods are neither accurate enough to be implemented in real-life situations nor are they fast enough to be implemented in real-time. Most of the times, a speed improvement is compromised by a fall in the detection accuracy. It is following the same motivation, that in this work attempts have been made to implement such models to not just accurately but also efficiently classify the images of the GI tract into the known medical conditions.

Chapter 2

Related works

Pogorelov et al. [12] introduced the KVASIR, which was a Multi-Class Image Dataset that was to be used for Computer Aided Gastrointestinal Disease Detection. The motivation for this work was the fact that though computer-aided diagnosis can be of immense help, the lack of proper dataset created a major hindrance. Again, Borgli et al. [4] introduced the Hyper-KVASIR dataset which is a high quality and the largest yet dataset of the human GI tract. This was an extended version of the KVASIR dataset. With the dataset, they provided a baseline where they trained several states of the art CNN models on the labeled dataset and achieved a maximum Matthews Correlation Coefficient (MCC) score of 0.902. However, according to their work, there is still a lot of improvement potential, in terms of accuracy and especially in terms of the speed of classification. Tong et al. [18] used semi-supervised learning so that they could also use unlabeled data for training their model. However, that method could not be applied since the unlabeled data also includes classes of data that are not of interest to the task and will lead to poor models if not handled properly. Bria et al. [5] addressed the problem of class imbalance. However, the class imbalance existing there was in terms of the number of pixels of the object and the background. Kang et al.[9] attempted to use image processing methods such that the defects in the endocrine system can be detected in real-time. The main aim of their work was not to classify the conditions at real-time, rather make it properly visible to the physician so that it doesn't go unnoticed.

Chapter 3

Dataset

One of the major disadvantages researchers faces while working on medical data is the lack of properly labeled data, especially when it comes to images related to the GI tract. Most of the datasets available have very few such images, ones that are not sufficient enough to be exploited to the best possible extent. Hence, we were provided with the Hyper-Kvasir dataset. This is the largest dataset of the human GI tract available to date. It consists of $\approx 10k$ labeled images and $\approx 99k$ unlabeled images and 373 labeled videos. The labeled images are classified across 23 such medical conditions along with an existing class imbalance. Fig. 3.1 shows the distribution across the 23 classes while Fig. 3.2 shows the class names.

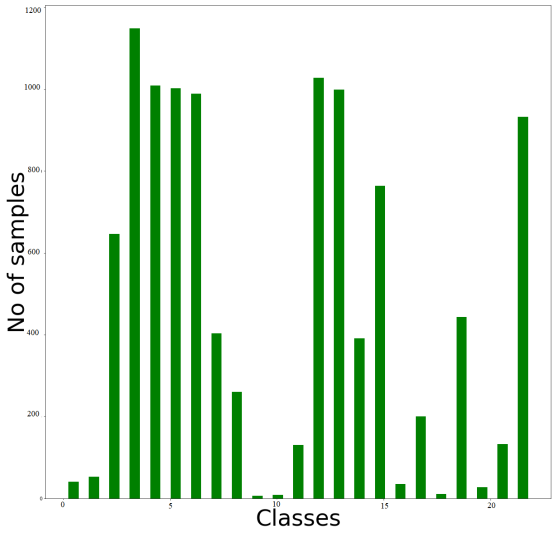

Figure 3.1: Class Distributions

Finding	Class Label
barretts	0
barretts-short-segment	1
bbps-0-1	2
bbps-2-3	3
cecum	4
dyed-lifted-polyps	5
dyed-resection-margins	6
esophagitis-a	7
esophagitis-b-d	8
hemorrhoids	9
ileum	10
impacted-stool	11
polyps	12
pylorus	13
retroflex-rectum	14
retroflex-stomach	15
ulcerative-colitis-grade-0-1	16
ulcerative-colitis-grade-1	17
ulcerative-colitis-grade-1-2	18
ulcerative-colitis-grade-2	19
ulcerative-colitis-grade-2-3	20
ulcerative-colitis-grade-3	21
z-line	22

Figure 3.2: Class Labels

Chapter 4

Classification of Lesions

4.1 Data Augmentation

The Hyper-Kvasir dataset contains $\approx 10k$ labeled images and $\approx 99k$ unlabeled images, which if exploited and labeled properly can provide an additional large amount of labeled data. However, classes that are not of interest to task could exist among the unlabeled images [4]. Hence, the current work avoids using unlabeled images.

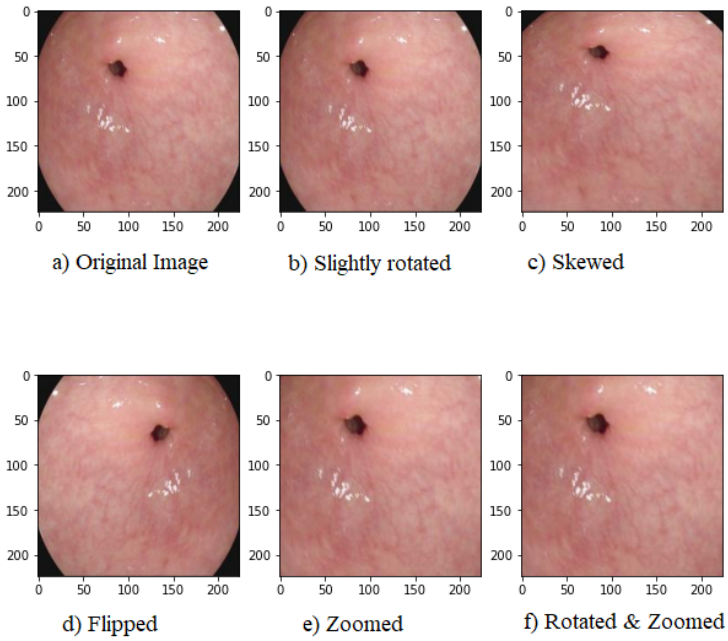

Figure 4.1: Augmented Images

The dataset contained only about 10k labeled images. A test set was created separately from the existing 10k images. This was done so that the model could finally predict on images it had never seen. Furthermore, the test set was created such that it contained at least *1* sample from each class. Asperti et al. [2] proved in their work that image augmentations had a positive impact on improving the efficiency of lesion detection in the GI tract. Hence, the images that were to be used for training were increased in number by augmenting them. Augmentations for each image involves images that were zoomed in, skewed, flipped, rotated or a combination of one of these, which was done using the Augmentor proposed by Bloice et al. [3], which was specifically meant for biomedical images. Each image was augmented an equal number of times to not disturb the class distribution. The images in the test set were left undisturbed. Once the images were augmented, the labeled images dataset was expanded to about $\approx 70k$ images. Fig. 4.1 shows the original image along with its augmentations.

4.2 Tiny Darknet Model

One important thing to keep in mind while classifying medical images is that besides being accurate, one also needs to be fast such that it can be implemented in real-time. Thus, in this work, we attempted to use the *Tiny Darknet*, model. Redmon et al. [14] proposed the Darknet reference model which has a smaller model size than most other architectures and reduces the total number of floating-point operations, thus reducing the inference time of the model. The Tiny Darknet is a reduced version of the Darknet model which is not just about 7 times smaller the size of the Darknet model but also has good accuracy scores on popular image classification benchmark datasets. The Tiny Darknet model is only a *23 layered models having only 913k trainable parameter*. Fig. 4.2 shows the number of layers in the model while Table 4.1 shows the number of parameters the model has. The primary motivation behind using this model was the reduction in inference time and smaller size which would allow it to be used on embedded devices as well.

Total parameters	913,879
Trainable parameters	913,879
Non-trainable parameters	0

Table 4.1: Number of Parameters in Tiny Darknet Model

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 224, 224, 3)]	0
conv2d_16 (Conv2D)	(None, 224, 224, 16)	448
max_pooling2d_4 (MaxPooling2	(None, 112, 112, 16)	0
conv2d_17 (Conv2D)	(None, 112, 112, 32)	4640
max_pooling2d_5 (MaxPooling2	(None, 56, 56, 32)	0
conv2d_18 (Conv2D)	(None, 56, 56, 16)	528
conv2d_19 (Conv2D)	(None, 56, 56, 128)	18560
conv2d_20 (Conv2D)	(None, 56, 56, 16)	2064
conv2d_21 (Conv2D)	(None, 56, 56, 128)	18560
max_pooling2d_6 (MaxPooling2	(None, 28, 28, 128)	0
conv2d_22 (Conv2D)	(None, 28, 28, 32)	4128
conv2d_23 (Conv2D)	(None, 28, 28, 256)	73984
conv2d_24 (Conv2D)	(None, 28, 28, 32)	8224
conv2d_25 (Conv2D)	(None, 28, 28, 256)	73984
max_pooling2d_7 (MaxPooling2	(None, 14, 14, 256)	0
conv2d_26 (Conv2D)	(None, 14, 14, 64)	16448
conv2d_27 (Conv2D)	(None, 14, 14, 512)	295424
conv2d_28 (Conv2D)	(None, 14, 14, 64)	32832
conv2d_29 (Conv2D)	(None, 14, 14, 512)	295424
conv2d_30 (Conv2D)	(None, 14, 14, 128)	65664
conv2d_31 (Conv2D)	(None, 14, 14, 23)	2967
global_average_pooling2d_1 (	(None, 23)	0
softmax_1 (Softmax)	(None, 23)	0

Figure 4.2: Layers in Tiny Darknet Model

4.3 Training Phase

The model used for this work was trained from scratch. Initially, $1/6^{th}$ images from each class were randomly selected for the generation of the initial seed. Initial seeds were generated by comparing the results of both *Adam optimizer* and *SGD optimizer*. The *F1 score* was monitored since it is directly proportional to the MCC, which was the final evaluation metric. SGD gave better results and hence the weights obtained using it were chosen. Fig. 4.3 shows the initial seed generation. The validation F1 score was monitored. Finally, the weights obtained for an F1 score of 0.48 on the validation data was used for initialising the new model. A relatively lower F1 score was chosen to ensure that an overfit

model is not used. SGD along with *weighted categorical cross entropy* was eventually used to train the model. After this, the new dataset that included the image augmentations was split accordingly into train and validation data. Both the training and the validation sets included the augmented images. The validation and training set were split according to a *1:10* ratio and was run for about *8k epochs*. The final training for the model was carried out by comparing the following two approaches.

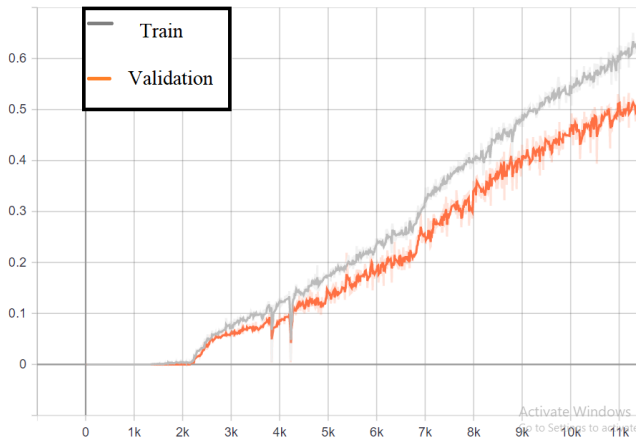

Figure 4.3: F1 score of train vs validation during initial seed generation

4.3.1 SGD + Momentum

The first method followed was SGD optimizer along with a *momentum value of 0.002*. It was observed that this model, in particular, showed positive learning every time it was preceded by a spike in loss or a dip in the F1 score. This was especially observed while training the model from scratch and can be useful for anyone intending to reproduce the same results. Fig. 4.4 shows the F1 score monitored for a few thousand epochs.

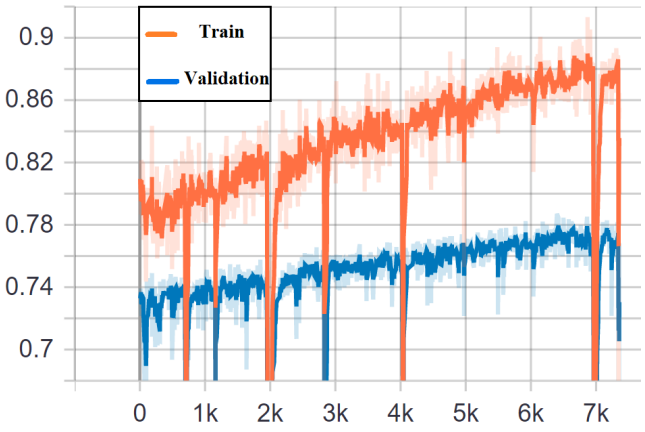

Figure 4.4: F1 score of Train vs Validation in SGD + Momentum

4.3.2 CyclicLR

Smith [17] introduced the concept of cyclic learning rates where the learning could be varied within a base and maximum value. This work exploits the concept of CyclicLR because it varies the *learning rate* after a set of steps such that the model can leave saddle points and can reach the optimal point at some point of time. Few dips, that are representative that the model is about to have positive growth in terms of learning was also observed in this. Fig. 4.5 shows the F1 score monitored for a few thousand epochs.

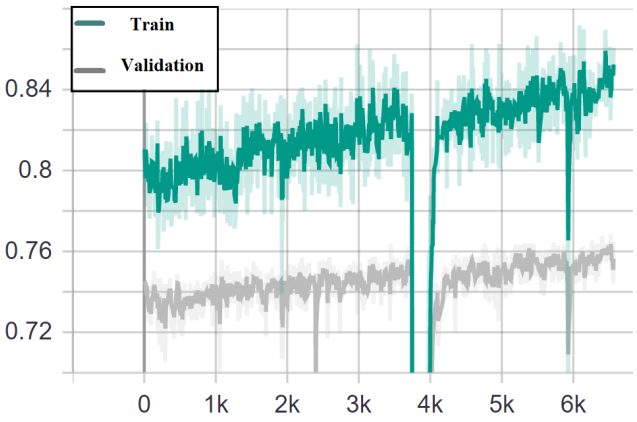

Figure 4.5: F1 score of Train vs Validation in CyclicLR

4.4 Results

The training on the Tiny Darknet model using *SGD + Momentum* was found to give relatively better results than *CyclicLR* and hence the weights obtained from it were chosen for the final predictions. The final model was used to predict the classes of images one at a time and using the time recorded, the speed of the model was recorded in terms of *fps*. The results obtained using the Tiny Darknet model on the Test data are shown in Table 4.2. The *F1 score, Precision and Recall* in both the cases were greater in case of *Micro* than in *Macro*. This is expected because Macro does not take into consideration the class imbalance whereas Micro does. The Tiny Darknet model achieved a good score on the Test data, especially for the *efficient detection task*, achieving up to *60fps*. However, there is a noticeable difference between the classification results between the Test and the Benchmark results. One plausible reason could be that the model is under-parameterized for the task. Therefore, chances are that it could overfit easily and thus fail on the evaluation data. Added the fact that the model was trained from scratch, further training for longer duration may be required for bringing in better performance. Fig. 4.6 shows the Histogram for the ground truth class distribution while Fig. 4.7 shows the histogram for the class distribution obtained after prediction.

Metric	Macro Average	Micro Average	Benchmark
Precision	0.712	0.74	0.910
Recall	0.708	0.74	0.910
F1	0.708	0.74	0.910
MCC	0.758	0.758	0.902
Average FPS	51.354	51.354	25
Maximum FPS	60.695	60.695	30

Table 4.2: Classification results on Test Data

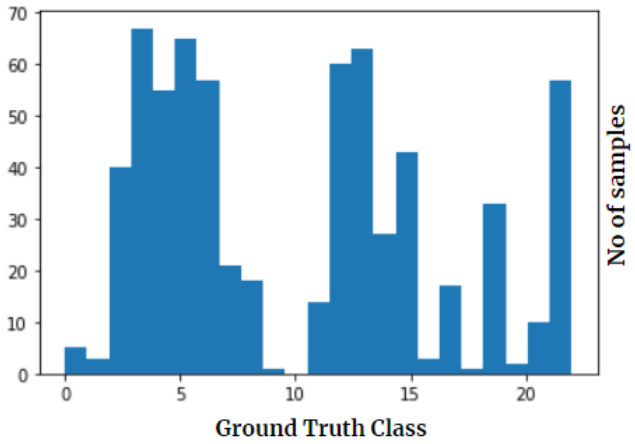

Figure 4.6: Histogram of Ground Truth Class Distribution

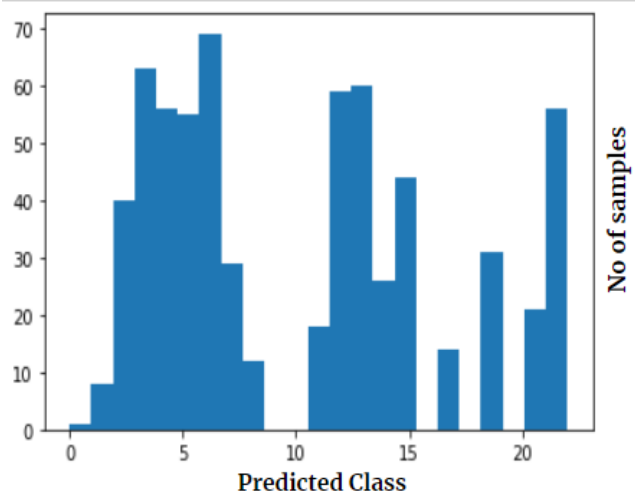

Figure 4.7: Histogram of the Class Distribution Obtained after Prediction

Chapter 5

Upsampling to Handle Class Imbalance

In the previous experiment of classifying the samples as in Chapter 4, we followed an approach where all the samples were augmented an equal number of times. This led to an increase in each class of images by a certain factor. As a result, that did not help the class imbalance problem. Therefore, in this section we discuss yet another approach, where the less frequent classes have been upsampled to remove a bit of the existing class imbalance problem. Accordingly we train our model using the same process as before and test on the same test set and get our results. Fig. 5.4 shows the training process. While Table 5.1 shows the results obtained. Only the micro average values have been shown as it gives more accurate results. As can be seen, the results achieved earlier were in fact better than the results achieved this time. Therefore, for our further experiments we stick by the same strategy as in Chapter 4.

Metric	Benchmark	Equal Aug. Micro Avg.	Unequal Aug. Micro Avg.
Precision	0.910	0.74	0.702
Recall	0.910	0.74	0.702
F1	0.910	0.74	0.702
MCC	0.902	0.758	0.724

Table 5.1: Results after revised augmentation

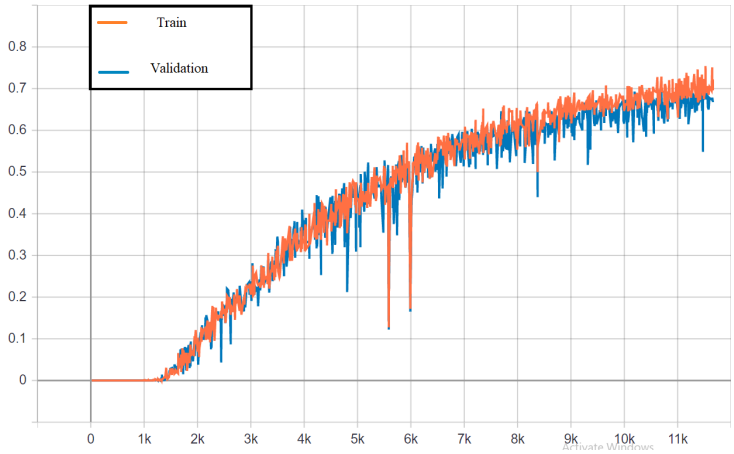

Figure 5.1: Training with lr = 1e-5

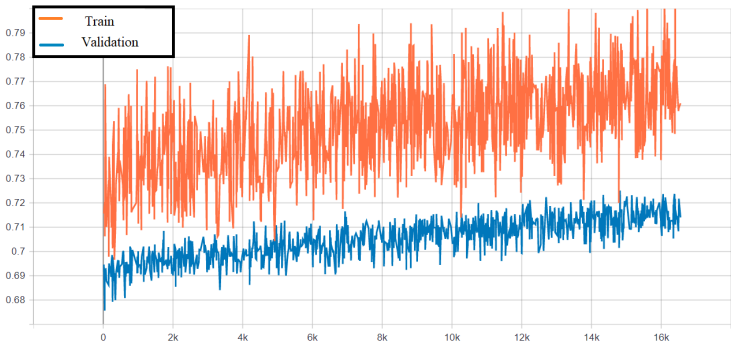

Figure 5.2: First half of training with lr = 1e-6

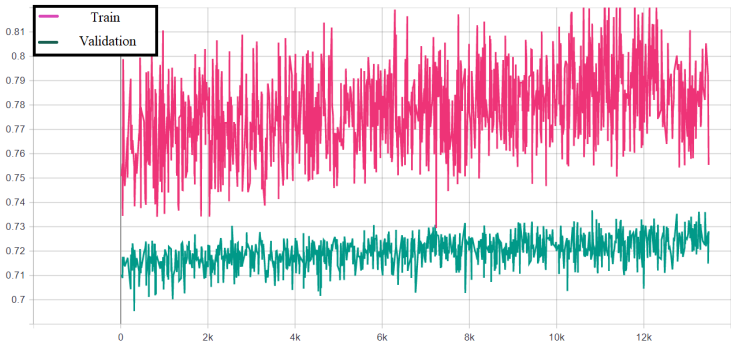

Figure 5.3: Second half of training with lr = 1e-6

Figure 5.4: F1 score vs epochs during the training process

Chapter 6

Ben's Preprocessing

6.1 Preprocessing and Augmentations

The Tiny Darknet Model is a small model and hence is relatively weaker than heavier models such as Resnet-152 [8] at extracting features from images [4]. Therefore, we decided to try some preprocessing method so as to help the model focus on the important areas in the images. Thus, for this task we chose the Ben's Preprocessing[1] method. This is a method that was implemented by Ben Graham the winner of one of the previous diabetic retinopathy competitions. Ever since then, the method has become rather popular in terms of preprocessing especially medical images. The main reasons behind using this method are:

1. Enhances finer details
2. Tackles different illumination problem

The method implements Gaussian Blur into the below equation

$$img = a.img1 + b.img2 + y \tag{6.1}$$

where a, b and y are hyper parameters tuned accordingly to give finer details. While **img1** is the original image itself in RGB, **img 2** is a Gaussian Blurred image. The Gaussian Blur parameter SigmaX, which is the standard deviation in the X direction, is varied to change the degree of preprocessing. Fig. 6.7 shows an image and its preprocessed images by varying the SigmaX value from 10 to 50. Accordingly, it was observed that on increasing the SigmaX the cavity region comes more into focus compared to its background. Hence, choosing SigmaX = 50, all the images were preprocessed before pipeling them. Accordingly, the images were then augmented. From Chapter 6, we observed that uniform augmentation proved to be more effective and hence all the images were augmented equal number of times as shown in Fig. 6.8. Once augmentations were

[1] `https://www.kaggle.com/banzaibanzer/applying-ben-s-preprocessing/`

done, we trained our same Tiny Darknet model and SGD + Momentum process for about 40k epochs on these preprocessed images and their augmentations. Fig. 6.13 shows the training process. The aim of this experiment was to check whether or not the preprocessing actually helped.

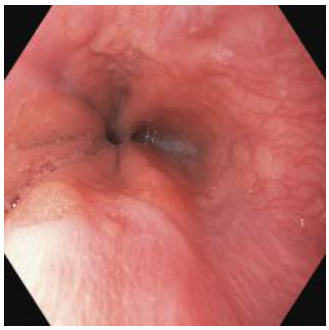

Figure 6.1: Original Image

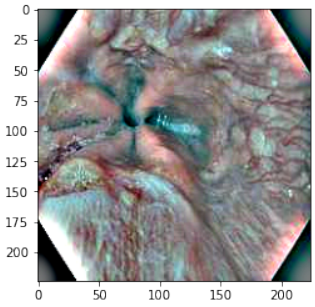

Figure 6.2: Sigmax = 10

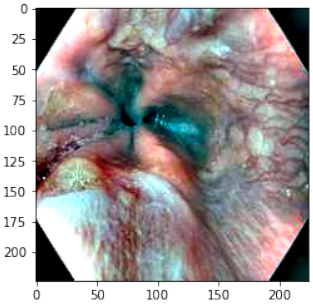

Figure 6.3: Sigmax = 20

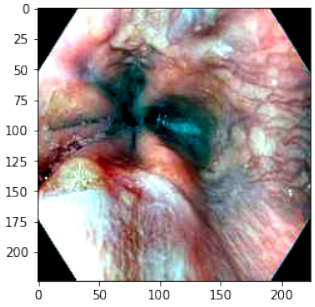

Figure 6.4: Sigmax = 30

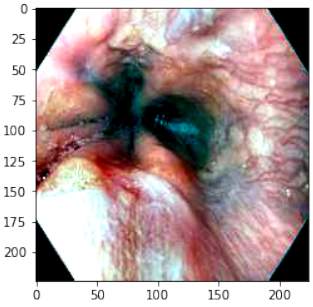

Figure 6.5: Sigmax = 40

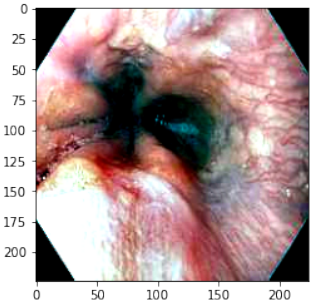

Figure 6.6: Sigmax = 50

Figure 6.7: Original Image and its preprocessed images

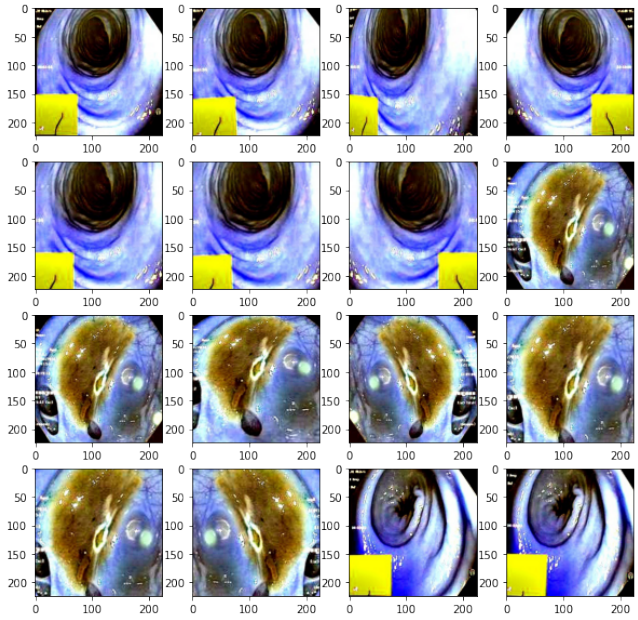

Figure 6.8: Augmentations of the Preprocessed images

6.2 Results

Once the model was trained and validated on the images, it was tested out on the same test set as the above experiments. The test set consisted of 719 images while the train and validation combined consisted of approximately 69k images. Table 6.1 shows the results obtained. As can be seen, preprocessing did help in improving the model's performance thus proving our intuition right that it helps the model to focus on the regions of importance.

Metric	Benchmark	Unpreprocessed Micro Avg.	Ben's Preprocessed Micro Avg.
Precision	0.910	0.74	0.768
Recall	0.910	0.74	0.768
F1	0.910	0.74	0.768
MCC	0.902	0.758	0.785

Table 6.1: Results after Ben's preprocessing

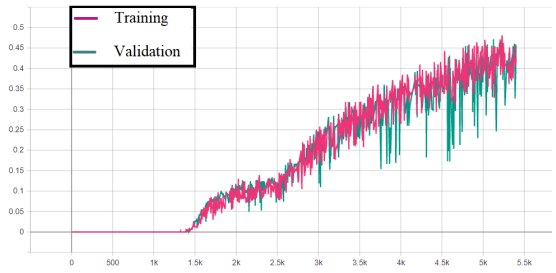

Figure 6.9: First half of Training with lr = 1e-5

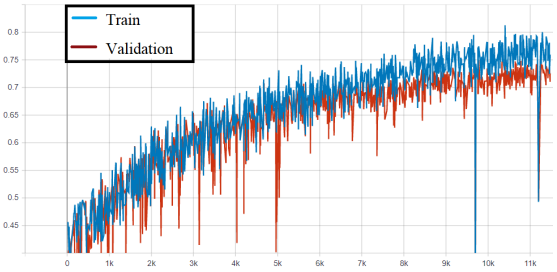

Figure 6.10: Second half of Training with lr = 1e-5

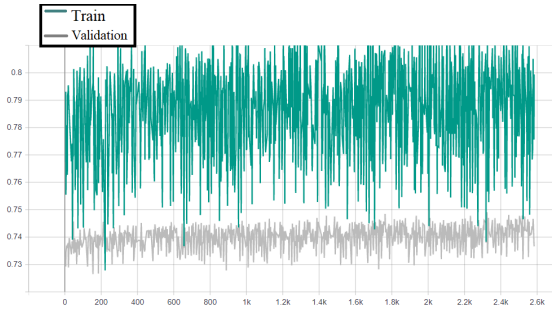

Figure 6.11: First half of training with lr = 1e-6

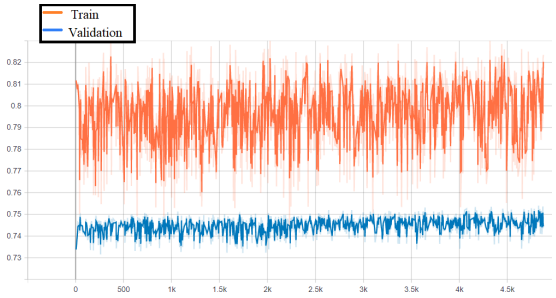

Figure 6.12: Second half of training with lr = 1e-6

Figure 6.13: F1 score vs epochs during the training process

Chapter 7

Selective Semi-Supervised Learning using GAN

7.1 What is Semi-Supervised Learning?

Supervised learning is an approach to train a machine learning model where a model is spoon fed information. However, the amount of labeled data is very limited in number compared to unlabeled data. Therefore, an approach known as Semi-Supervised Learning to utilise the unlabeled data using the limited amount of labeled data was developed. Semi-Supervised Learning is a technique where we only use a small set of labeled data from a large amount of unlabeled data to train our model. Zhu et al. [19] introduced semi-supervised learning as the natural method by which humans learn in the presence of both labeled and unlabeled data. Kingma et al. [10] further proposed a method where they used generative models to display efficient generalisation of models on datasets having even a small amount of labeled data. The Hyper-kvasir [4] dataset that has been used for this work consists of 10k labeled images but also consists of around 99k unlabeled images. However, the unlabeled images could consist of classes of images beyond the 23 class of images present in the dataset. Therefore, GANs [7] were used for this purpose so that we could exploit their generalisation ability to **selectively label our data**.

7.2 What is a GAN?

Generative Adversarial Networks, or GANs for short, are an approach to generative modeling using deep learning methods. Goodfellow et al. [7] proposed a new generative model which was trained via an adversarial process, in which the training occurs simultaneously in two models: a generative model and a discriminative model. The generative model captures the data distribution, and the discriminative model estimates the probability that a sample is real

or generated. Several works have been done on Generative Adversarial Network. Radford et al. [13] proposed a class of CNNs called deep convolutional generative adversarial networks popularly known as DCGAN. The aim of this was to bridge the gap between usage of CNN for supervised and unsupervised learning. There was also Arjovsky et al. [1], where they introduced a new algorithm named WGAN. This was proposed as an alternative to traditional GAN training where they showed methods to improve the stability of GAN training.

7.3 What is a Semi-Supervised GAN?

The Semi-Supervised GAN or SGAN is an extension of the Generative Adversarial Network architecture for addressing semi-supervised learning problems. Odena et al. [11] in their work introduced the concept of a Semi-Supervised GAN through a GAN-trained classifier and showed that it is able to perform as well as or better than a standalone CNN model on the MNIST handwritten digit recognition task when trained with 25, 50, 100, and 1,000 labeled examples. Further Salimans et al. [16] from OpenAI achieved at the time state-of-the-art results on a number of image classification tasks using a semi-supervised GAN, including MNIST.
The discriminator in a traditional GAN is trained to predict whether a given image is real or fake, allowing it to learn features from unlabeled images. In a SGAN, the discriminator model is updated to predict K+1 classes, where K is the number of classes in the prediction problem and the additional class label is added for a new *fake class*. It involves directly training the discriminator model for both the unsupervised GAN task and the supervised classification task simultaneously. As such, the discriminator is trained in two modes: a supervised and unsupervised mode.

- Unsupervised Training: In the unsupervised mode, the discriminator is trained in the same way as the traditional GAN, to predict whether the example is either real or fake.
- Supervised Training: In the supervised mode, the discriminator is trained to predict the class label of real examples.

Training in unsupervised mode allows the model to learn useful feature extraction capabilities from a large unlabeled dataset, whereas training in supervised mode allows the model to use the extracted features and apply class labels.

7.4 Architecture

The generator in a SGAN is same as that of a standard GAN. Fig. 7.3 shows the architecture of generator used in the SGAN. The discriminator is what is

different because the discriminator both outputs the number of classes as well as whether the image is real or fake. There are several approaches to designing the discriminator. However, for this work we designed a single discriminator that gives multiple outputs. This is a single model with one output layer for the unsupervised task and one output layer for the supervised task. Since **classes = 23** in the Hyper Kvasir dataset, the discriminator model predicts **23 + 1 = 24 classes**. Furthermore, since the work aims to do selective semi-supervised learning thus the aim is to train the model such that it predicts the class of images that do not belong to the 23 classes into the 24^{th} class which is essentially a fake class. Besides this the model also predicts fake or real like a standard GAN. Fig. 7.4 shows the model architecture for the discriminator model of a SGAN.

7.5 Training

Once the generator and the discriminator model for SGAN were designed, we trained our model on the same training data as the above experiments using the same seed value. Random noise was provided as the input to the generator. Fig. 7.1 shows a sample input. The test set was hence also the same. The train and test data were such that samples from each class were present. We also used the unlabeled data but we used 50k out of the 99k images.
Fig. 7.2 shows the generator's initial output when the model is not trained. Eventually the Generator and the Discriminator were trained using an adversarial technique and the unlabeled dataset is classified into the 24 classes using the discriminator classifier. For this experiment there was no way of ensuring that the unlabeled samples were classified correctly or not. Furthemore to know if the images classified into class 24 are correct would require manual annotations of the unlabeled data from medical experts.

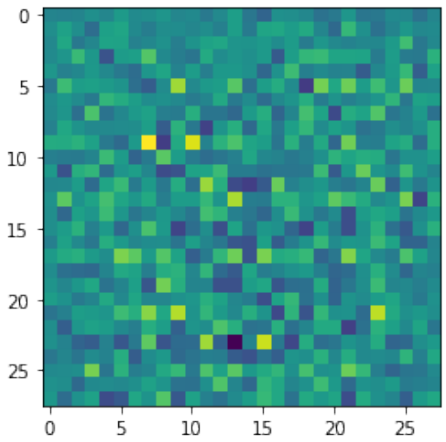

Figure 7.1: Random Noise Input to the Generator

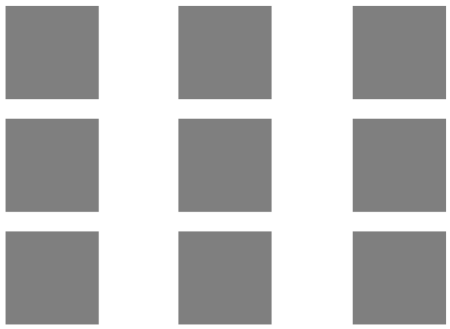

Figure 7.2: Untrained Generator Output

7.6 Results

Once the SGAN was trained we observed some of the images that the GAN had generated while it was learning. Fig. 7.5 shows some of the images that the model had generated during training. Once the model was trained both in supervised and unsupervised mode we tested the SGAN classifier on the test set and Table 7.1 shows the results achieved. As can be seen, the results are still not good enough to be used for medical purposes and therefore further work needs to be done to improve it and make it usable. However, this shows us that there is potential in the unlabeled data being selectively used by a GAN to train a classifier.

Metric	SGAN
Precision	0.587
Recall	0.587
F1	0.587
MCC	0.621

Table 7.1: Results using SGAN Classifier on Test Data

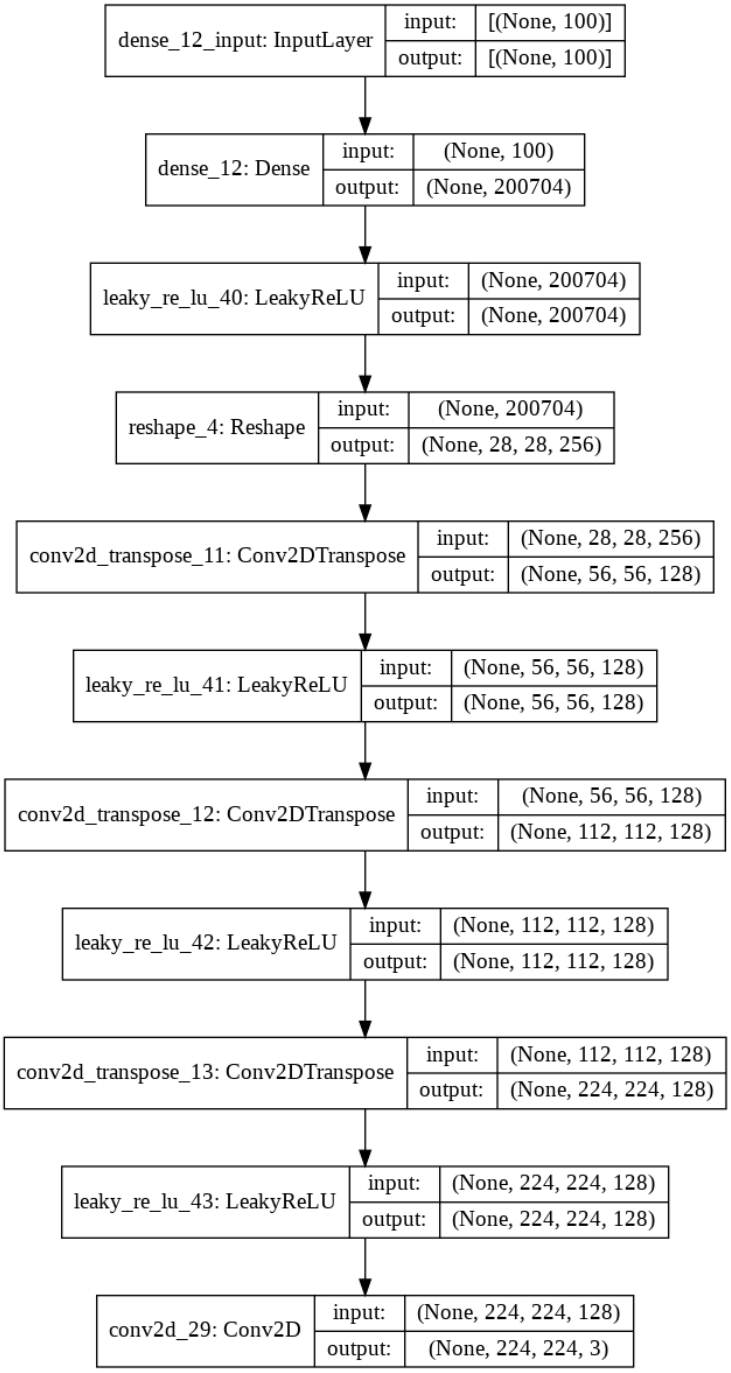

Figure 7.3: Semi-Supervised GAN Generator Model

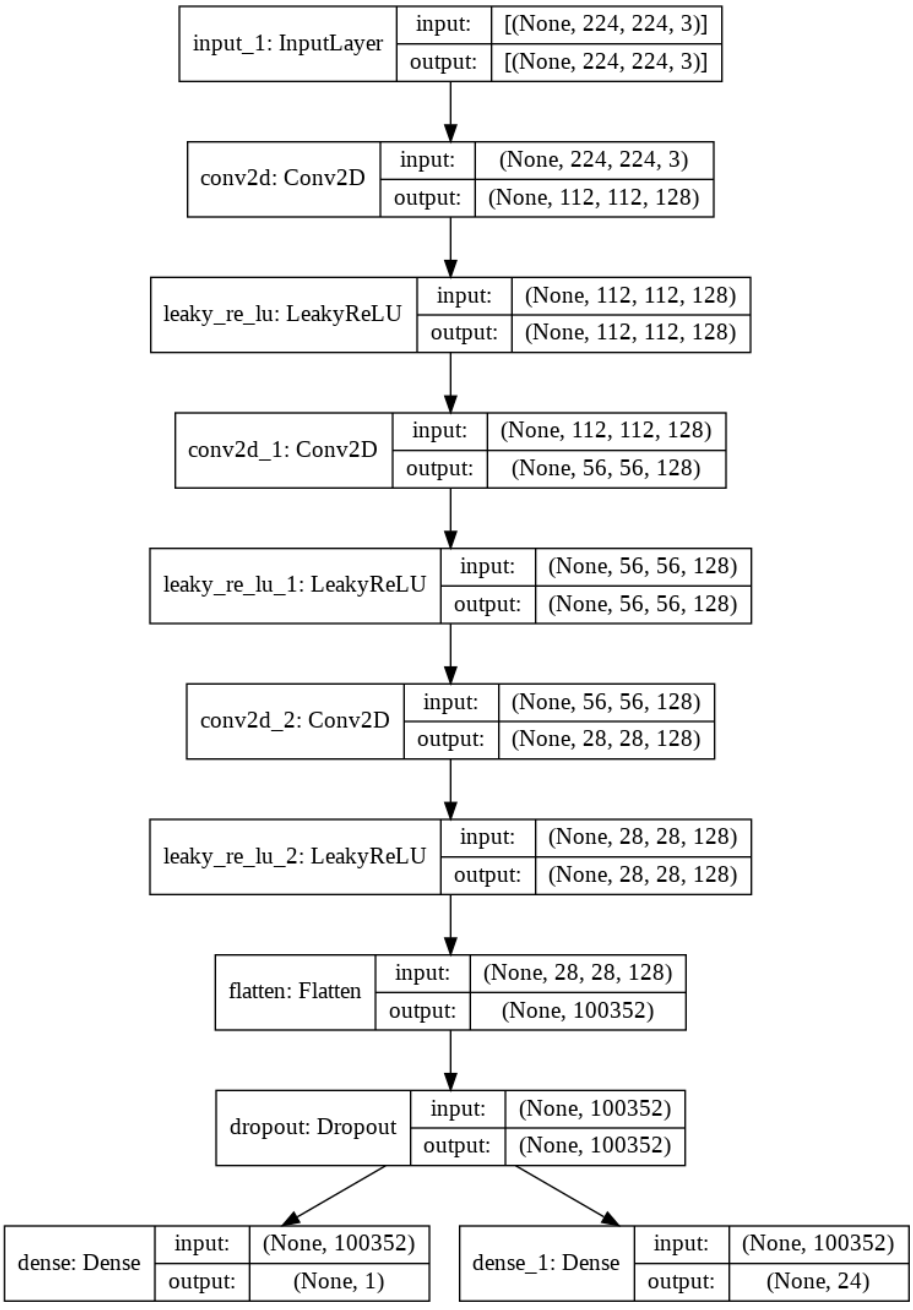

Figure 7.4: Semi-Supervised GAN Discriminator Model With Unsupervised and Supervised Output Layers

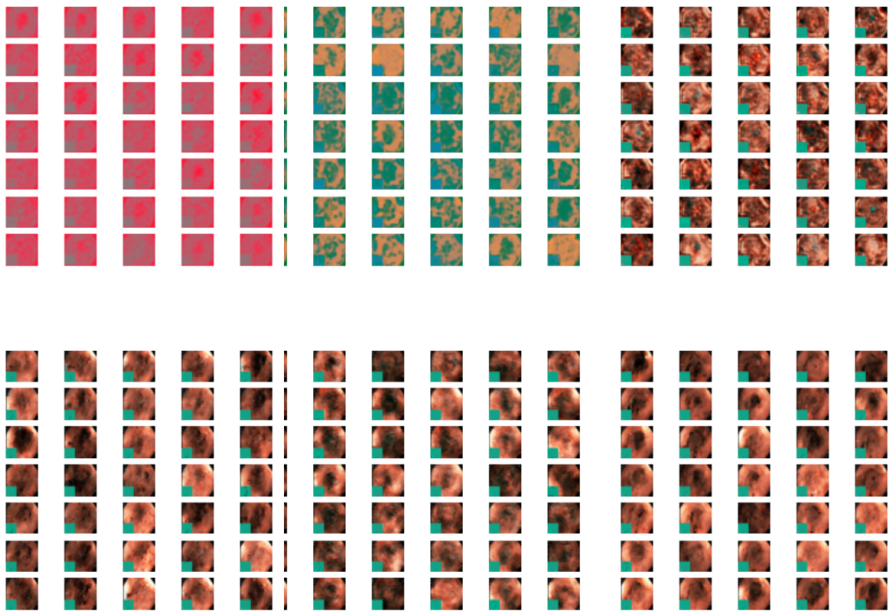

Figure 7.5: Images generated by the Generator with increasing epochs during training

Chapter 8

Segmentation of the Polyps

8.1 UNet

The work done also predicts the segmentation masks of images of class *polyps*, which is an important class of medical findings in the human GI system. Ronneberger et al.[15] proposed the *UNet model* for the purpose of semantic segmentation. Since the current work aims to do the same, it has been done using the *UNet model*. The number of images and their masks in the dataset were about 1,000. Hence, each image along with their masks was augmented using Augmentor[3] and thus the dataset was expanded to about 5k. A Test data was also created using some of the original images. Finally, the remaining data was split into train and validation sets in a *8:2 ratio* and then run for about *2k epochs*. Fig. 8.1 shows the original image with its augmentations while Fig. 8.2 shows the augmentations of the masks of the above images.

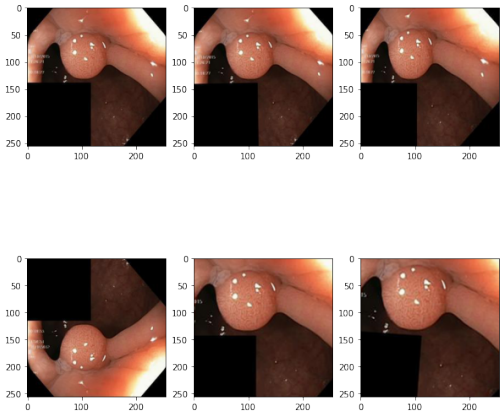

Figure 8.1: Original Image with Augmentations

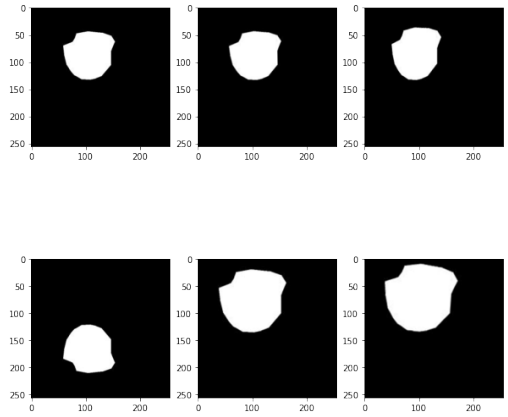

Figure 8.2: Mask of the Original Image and its Augmentations

8.2 Results

Once the UNet model was trained, the model was used for evaluation. The average evaluation metrics for UNet generated on the Test Table is as shown in Table 8.1. The UNet model achieved a decent score on the Segmentation task in terms of the Jaccard distance. One way results could be further improved is by training it for a sufficient amount of time. This could also be done using further post-processing. This could be done by filling up some of the holes that are present in the mask images generated during prediction. Improvements in this could perhaps give better results. Fig. 8.3 shows the ground truth masks of the original images while Fig. 8.4 shows the predicted masks of the images. As can be seen, most of the predicted images have some holes in them and therefore require some form of post-processing.

Metric	Metric Value
Precision	0.821
Recall	0.832
F1	0.802
Jaccard Similarity	0.853

Table 8.1: Segmentation results on Test Data

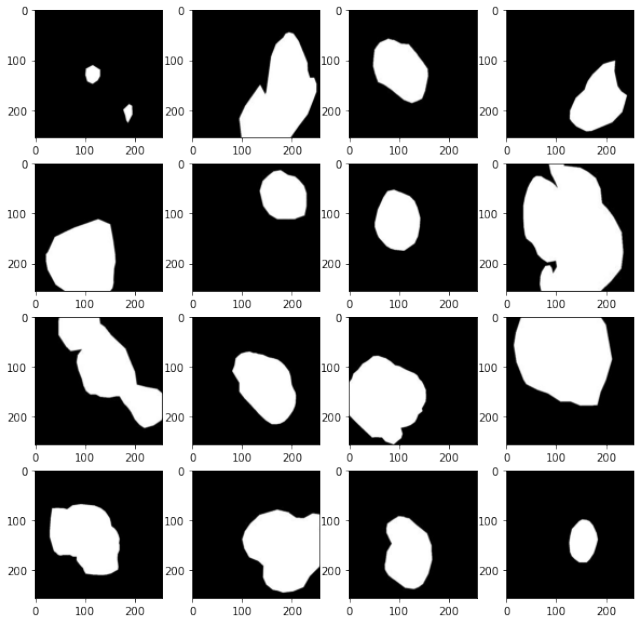

Figure 8.3: Ground Truth Mask Images

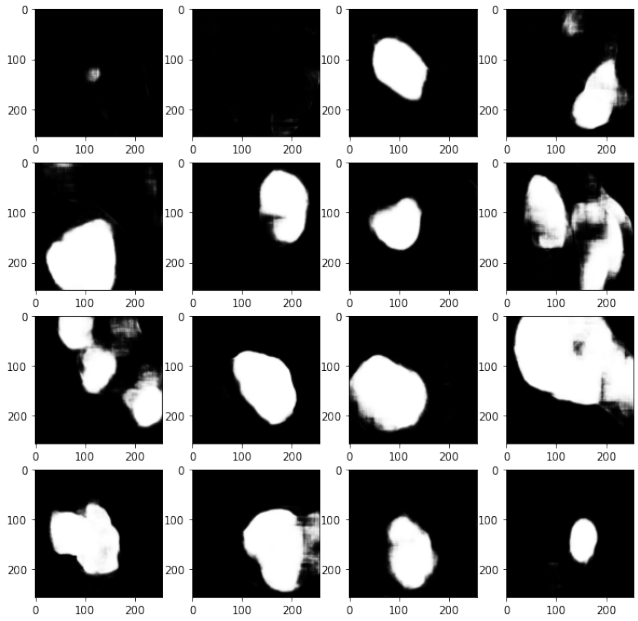

Figure 8.4: Predicted Mask Images

Chapter 9

Conclusion and Future Works

9.1 Conclusion

After performing this work, several observations had been made. Some of the important ones being

- The accuracy achieved by the classification model still needs to be improved upon. However, the *speed* and *size* of the model are worth noticing. It achieved a maximum fps of 60 and the model had a size of only about **3.5 MB**. Although the accuracy of the model needs improvement but this showed potential.

- Effective preprocessing of the images, helped the model in classifying better. Thus proving that narrowing down the focus region of the model helped.

- Semi-supervised learning and then classifying using SGAN proved that the unlabeled data was usable to improve future results.

9.2 Future works

Even though quite an additional amount of work needs to be done, some of the results obtained show good potential for successful future implementation. The motivation to use the labeled data in the beginning was to check how well the model performed using the limited amount of labeled data. Now that we have shown that unlabeled data can be used, therefore attempts to further improve the model shall be made. Furthemore, we would also try to get some images belonging to the *fake* class annotated by medical experts. This would ensure that the model is indeed capable of performing selective semi-supervised

learning. Finally reminding ourselves that all this work has been done with the sole aim of helping doctors, we will continue to improve our work so as to make endoscopy an automated and reliable process.

Bibliography

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. *Wasserstein GAN*. 2017. arXiv: 1701.07875 [stat.ML].

[2] Andrea Asperti and Claudio Mastronardo. "The effectiveness of data augmentation for detection of gastrointestinal diseases from endoscopical images". In: *arXiv preprint arXiv:1712.03689* (2017).

[3] Marcus D Bloice, Peter M Roth, and Andreas Holzinger. "Biomedical image augmentation using Augmentor". In: *Bioinformatics* 35.21 (2019), pp. 4522–4524.

[4] Hanna Borgli et al. "HyperKvasir, a comprehensive multi-class image and video dataset for gastrointestinal endoscopy". In: *Scientific Data* 7.1 (2020), p. 283. ISSN: 2052-4463. DOI: 10.1038/s41597-020-00622-y. URL: https://doi.org/10.1038/s41597-020-00622-y.

[5] Alessandro Bria, Claudio Marrocco, and Francesco Tortorella. "Addressing class imbalance in deep learning for small lesion detection on medical images". In: *Computers in Biology and Medicine* (2020), p. 103735.

[6] Spiros V Georgakopoulos et al. "Weakly-supervised convolutional learning for detection of inflammatory gastrointestinal lesions". In: *2016 IEEE international conference on imaging systems and techniques (IST)*. IEEE. 2016, pp. 510–514.

[7] Ian J Goodfellow et al. "Generative adversarial networks". In: *arXiv preprint arXiv:1406.2661* (2014).

[8] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[9] J Kang and R Doraiswami. "Real-time image processing system for endoscopic applications". In: *CCECE 2003-Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology (Cat. No. 03CH37436)*. Vol. 3. IEEE. 2003, pp. 1469–1472.

[10] Diederik P Kingma et al. "Semi-supervised learning with deep generative models". In: *arXiv preprint arXiv:1406.5298* (2014).

[11] Augustus Odena. "Semi-supervised learning with generative adversarial networks". In: *arXiv preprint arXiv:1606.01583* (2016).

[12] Konstantin Pogorelov et al. “Kvasir: A multi-class image dataset for computer aided gastrointestinal disease detection”. In: *Proceedings of the 8th ACM on Multimedia Systems Conference*. 2017, pp. 164–169.

[13] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434* (2015).

[14] Joseph Redmon and Ali Farhadi. “YOLOv3: An Incremental Improvement”. In: *arXiv* (2018).

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[16] Tim Salimans et al. “Improved techniques for training gans”. In: *arXiv preprint arXiv:1606.03498* (2016).

[17] Leslie N Smith. “Cyclical learning rates for training neural networks”. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2017, pp. 464–472.

[18] Li Tong, Hang Wu, and May D Wang. “CAESNet: Convolutional AutoEncoder based Semi-supervised Network for improving multiclass classification of endomicroscopic images”. In: *Journal of the American Medical Informatics Association* 26.11 (2019), pp. 1286–1296.

[19] Xiaojin Zhu and Andrew B Goldberg. “Introduction to semi-supervised learning”. In: *Synthesis lectures on artificial intelligence and machine learning* 3.1 (2009), pp. 1–130.