university of groningen

faculty of science and engineering

Exploring the latent space of the StyleGAN model for the controlled generation of synthetic colonoscopy images

Arseniy Nikonov

university of groningen

faculty of science and engineering

University of Groningen

Exploring latent space of StyleGAN model for controlled generation of synthetic colonoscopy image

Master's Thesis

To fulfill the requirements for the degree of
Master of Science in Artificial Intelligence
at University of Groningen under the supervision of
dr. F. Cnossen (Director of Education Ba/Ma Artificial Intelligence/Human-Machine Communication — Associate Professor Cognitive Engineering, University of Groningen)
and
dr. ir. P.M.A. van Ooijen (Scientific Researcher / Associate Professor, University of Groningen & University Medical Center Groningen)

Arseniy Nikonov (s2977419)

October 3, 2022

Contents

Acknowledgments

The following paper was done in collaboration with the University Medical Centre of Groningen. I would like to thank the members of both institutions for their guidance, wisdom, and medical insight along the way.

Deepest gratitude to Jouke Van van der Laan for his medical insight and enormous work in labeling the images.

Many thanks to Ioannis Giotis from ZiuZ for their insight into the field of endoscopic imaging.

Abstract

Machine learning models have successfully solved many tasks in the past years, but they usually require a massive amount of data to succeed. Medical dataset typically doesn't have enough data to achieve the best performance of machine learning models. The colonoscopy domain is one of the fields that have only limited publicly accessible data, and not all of that data can be used due to the difference in the equipment. At the same time, computer-assisted diagnosis proved helpful in increasing the detection rate during colonoscopy. To solve the data scarcity, generative adversarial networks(GAN) can be used to synthesize fake images. StyleGAN2-Ada model produced realistic images that can be used as an addition to the original training dataset. Even though the images produced by StyleGAN are quite realistic there is a lack of control over the generated image. Several methods of exploring the latent space of StyleGAN were explored to achieve better control over the generated image. Those methods proved that latent exploration is possible as a concept but only achieved such changes as the vertical/horizontal location of a polyp or change in its size. The performance of a detector with the addition of synthesized images achieved a mean average precision score 83.4% compared to the 82.7% obtained by the detector trained on the original dataset.

1 Introduction

Colorectal Cancer(CRC) is one of the most common types of cancer to be diagnosed worldwide[1]. It is also the second deadliest type of cancer[2]. However, since the 1980s, the amount of death caused by CRC has been slowly decreasing. This trend is mainly attributed to the progress in early detection and treatment. Early detection is important in bringing down the mortality rate and reducing the treatment cost.

Colonoscopy is the recommended CRC visual examination screening method. The colonoscopy procedure has several advantages, such as high sensitivity and the ability to remove precancerous and small cancerous lesions at the time of detection[3].

Based on the shape, polyps can be separated into three categories: sessile polyps, pedunculated polyps, and flat polyps.

Most of CRC develops through the adenoma-carcinoma sequence[4]. Normally this takes a period of 10-15 years[5] which allows for an intervention at the earlier stages of its development. Early detection is critical for preventing the development and spread of cancer. The polyp detection success rate relies on the person performing a colonoscopy[6]. It was shown that the polyp detection rate goes down as the day progresses[7]. It is more difficult to detect flat or sessile polyps[8].

Due to those reasons, a computer-aided diagnosis(CAD) framework would be helpful for physicians. Such a system would extract features from the colonoscopy image/video frame and output location and label predicted by the system. Successful implementation of the system may minimize variations in the adenoma detection rate between different endoscopists[9] or between procedures performed at different times of the day[10]. As an additional benefit, CAD can also be used for training endoscopists. The emergence of deep learning led to the development of new, faster, and more accurate methods than traditional artificial intelligence technologies.

One of the limitations of the deep learning approach is the need for a massive amount of training data to obtain excellent performance. Labeling must be carried out by medical practitioners, which leads to a shortage of labeled images. To create a dataset of such size, a collaboration between multiple medical centers is required. Possible problems for the collaboration include data privacy laws, different endoscopy equipment, and competing interests.

A possible solution to the size of the dataset is data augmentation. Data augmentation is a data science technique to increase the size of the dataset by adding modified versions of the existing images or newly synthesized images. Data augmentation techniques can be split into several categories: basic augmentation techniques, deformable augmentation techniques, and deep learning augmentation techniques[11]. Affine geometric transformations, cropping, noise injection, intensity changes, and combinations of those are part of basic augmentation techniques. Deformable techniques can be used to provide additional variability. The deformable augmentation techniques include randomized displacement field, spline interpolation, deformable image registration, and statistical shape models. At last, deep learning augmentation techniques are used to learn the representation of images and overall distribution automatically. Using those, it is possible to generate synthetic images to add to a dataset. Generative adversarial networks(GAN) are the most common deep learning method used for image synthesis. Less common deep learning augmentation techniques include variational auto-encoders, different adversarial methods, variational dropout, and many others.

This project focuses mainly on generative adversarial techniques to generate synthetic images. GAN-based techniques were used in multiple studies to generate realistic images in the medical domain
[12][13][14][15]. Realistic colonoscopy images were generated during the preceding master thesis by Mihai Popescu[16].

Latent space exploration is one of the extensions of the generative adversarial network field of research. It was shown that by alternating latent space vectors it is possible to perform semantic editing of images in a person's face domain [17][18][19]. If a similar procedure can be performed for the medical domain that would enrich medical databases. Another advantage of synthetic images is the fact that it doesn't fall under the restrictions of privacy laws[20]. The addition of synthetic data to training the dataset might also improve the performance of classification/detection models. This project mainly focused on manipulating synthetic images using the latent space of GAN and improving the performance of a detector using synthetic images.

1.1 Research Questions

To summarize, this thesis focuses on the following problems:

Q1. Can input latent space vector be manipulated in a way to generate images with certain properties?

Q2. Which of the latent space manipulation methods is more suitable for the colonoscopy domain?

Q3. Can the performance of the detection system be improved by using generated images?

1.2 Thesis Outline

The structure of the thesis is as follows: Chapter 2 provides the theoretical background relevant to the research, including neural networks, deep learning, generative adversarial networks, detection models as well as state of the art for latent space exploration. Chapter 3 is dedicated to the methodology of the research performed for this paper, including dataset description and preparation, network evaluation, and latent space exploration methods. In Chapter 4 experimental setup and configurations of various pipelines are presented. Chapter 5 displays the results of the experiments. Chapter 6 focuses on discussion of the results. In Chapter 7 interpretation of the results related to the research questions are given and potential improvements and future research is discussed.

2 Background Literature

This section provides the theoretical background for this study. In subsection 2.1 brief introduction is given concerning the general concepts of Neural Networks(NN) and Deep Learning(DL). Subsection 2.2 is dedicated to detection models. Subsection 2.3 provides the theoretical background of generative adversarial networks(GAN), and subsection 2.4 describes the current state-of-the-art methods.

2.1 Neural networks

2.1.1 Artificial neural network

Artificial neural networks(ANN) are a type of machine learning method. ANNs consist of layers of neurons containing input, hidden, and output layers. Each neuron connected to another had a corresponding weight and threshold. Weights of the connections are typically learned during training, which can be split into alternating forward and backward passes. During forward pass, sets of input are fed into the network resulting in an output. During backward passes, the error between the output of the network and the desired output is calculated. Finally, the network weights are updated based on the error in a process called backpropagation[21].

2.1.2 Types of learning

There are three main types of learning: supervised, unsupervised, and reinforcement learning. Supervised learning requires having labeled data that is used during the training to 'supervise' the training. Unsupervised learning algorithms discover patterns of data without the use of human supervision. Different parts of this project use different learning paradigms. Supervised learning is the most fitting for the task of detecting a polyp. Generative methods are usually represented as an unsupervised learning problem. GANs, in particular, fall under the subcategory of unsupervised learning: self-supervised learning. Self-supervised learning problems are part of unsupervised learning that is framed as a supervised problem in order to apply supervised learning algorithms. Different methods of latent space exploration fall under both categories of supervised and unsupervised problems.

2.2 Detection models

Object detection is a common computer vision task used to detect instances of objects in digital mediums such as images or videos. The ultimate goal of object detection can be described with a question: "Where is what?". The ideal system would be able to correctly tell us which objects are present at which location in the image. Using that description, the object detection task can be split into two sub-tasks: object localization and object classification. Given those two tasks, the object detection pipeline can usually be divided into the following stages: informative region selection, feature extraction, and classification[22]. The way object detection is performed can be split into two branches: traditional image processing techniques or deep learning-related methods.

A classical approach to object detection would involve scanning the whole image with multiple sliding windows, extracting visual features, and trying to classify a target object. PASCAL VOC competition was used to evaluate the new state-of-the-art methods between 2005 and 2012. In the last years of competition, however, only small gains were acquired on the object detection task, [23] mainly because the sliding window approach is redundant and classification is limited with hand-crafted features. In the last several years, object detection performance was also significantly

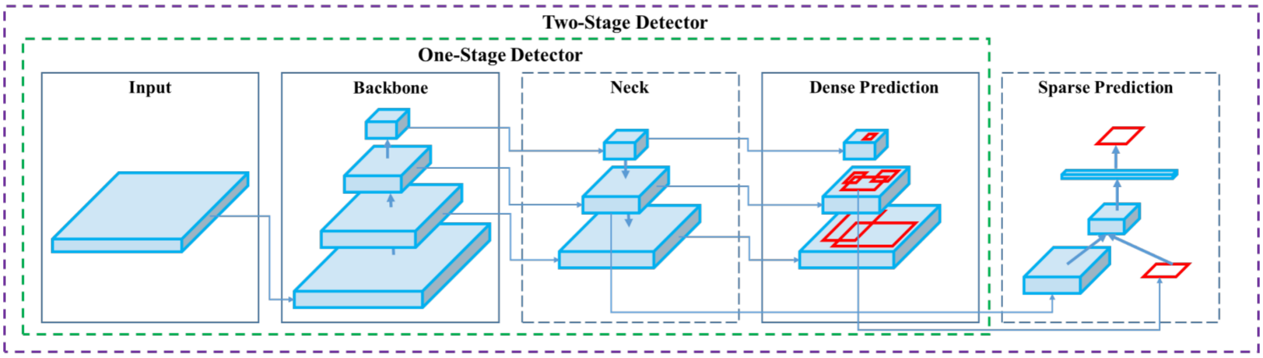


Figure 1: Object detector. Backbone is usually pre-trained on ImageNet. The Head comprises dense/dense and spark modules and is used to predict classes and bounding boxes of an object. Taken from [32].

improved due to advances in the deep learning field. This project used only deep learning-related methods, and the following paragraphs will focus on those.

Significant gains were obtained with the emergence of deep learning and the creation of the region convolutional neural network(RCNN) network[24]. Modifications to RCNN were made in Fast RCNN[25], Faster RCNN[26], Mask R-CNN[27], and the latest evolution granulated RCNN(G-RCNN)[28]. One of the disadvantages of the RCNN family of networks is the fact that they belong to a subclass of two-stage detectors. Two-stage detectors typically search the region of interest and classify that region in two different steps. As a result, those systems are slower but have higher classification accuracy. Another downside of the two-stage detectors is the inability to train in an end-to-end fashion since the step between the region of interest search and the classification is not differentiable.

As an alternative, one-stage detector systems treat object detection as a simple regression problem by taking an input image and learning the class probabilities and bounding box coordinates. The most popular one-stage object detectors are ”You only look once”(YOLO)[29], RetinaNet[30], and Single Shot MultiBox Detector(SSD)[31]. One-stage detectors’ main advantage is their speed, allowing them to use them in real-time applications. As the main detector system, an architecture based on YOLOv4[32] was used for most tasks.

A general object detector architecture can be seen in Figure1.

2.2.1 YOLO architecture

YOLO architecture reframed object detection as a single regression problem. Doing so allowed end-to-end training since there is no need for a complex pipeline. Moreover, the YOLO network is extremely fast, allowing real-life detection, using the whole image, and learning generalizable representations of objects[29]. For the colonoscopy procedure, a detector capable of live detection is needed, which leads to a choice of YOLO based model as a detector system for this project. In particular, the YOLOv5 detector[33] was used mainly for technical reasons.

The original YOLO model architecture was inspired by the GoogLeNet model for image classification[34]. The architecture of the original YOLO system is presented in Figure 2. YOLO system divides the image into an S x S grid. A grid cell is responsible for detecting an object if the center of the object falls into that grid cell. Each grid cell predicts bounding boxes which consist of 5 predictions: x,y,w,h, and confidence, where x and y are the centers of the bounding box while w and h are the width and

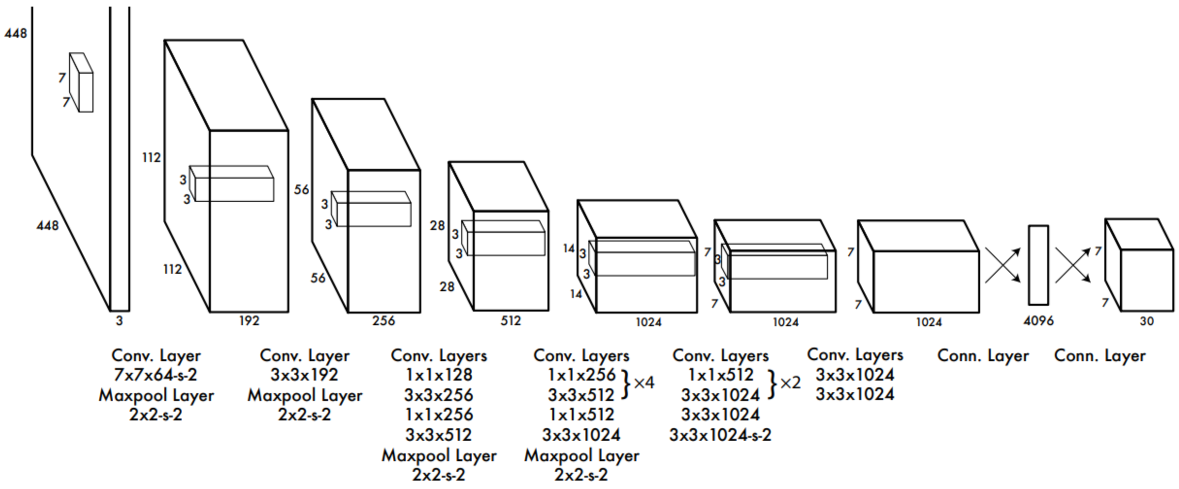


Figure 2: The architecture of the original YOLO detection system. Taken from [29].

height of the bounding box. Apart from the bounding boxes, each grid cell also predicts conditional class probabilities. The second iteration of YOLO architecture was YOLO9000[35]. Several improvements were made for YOLO9000, with the most notable being a change in the classification model from GoogLeNet to Darknet-19. The next iteration of YOLO - YOLOv3[36] introduced a number of small improvements as well as another chance to the classification model from Darknet-19 to Darknet-53. Darknet-53 is a middle ground between Darknet-19 and ResNet[37] models, achieving better performance than Darknet-19 but faster than ResNet models.
The latest iterations of YOLO architecture are YOLOv4 and YOLOv5. YOLOv5 was published shortly after YOLOv4[32], and a big part of the changes from YOLOv3 to YOLOv5 and YOLOv3 to YOLOv4 are similar. One of the main differences is that YOLOv5 utilizes the PyTorch framework instead of the Darknet that was used for YOLOv1-4. Nepal and Eslamiat showed that YOLOv5 had higher accuracy than YOLOv3 and YOLOv4 models[38]. YOLOv5 is an extremely fast detector, and on top of that, it can be used on conventional GPUs such as NVIDIA GTX 2070.

2.3 Generative methods

Generative models are created for the purpose of generating new data instances by means of capturing the distribution of the data itself. Generative adversarial networks are a subset of generative methods. Other generative methods include variations of Boltzmann machines, directed generative networks, autoencoders, and generative stochastic networks[39].

2.3.1 Generative Adversarial Networks

Generative adversarial networks were introduced in 2014 by Goodfellow et al. [40]. The GAN model comprises two submodels: generator(G) and discriminator(D). The GAN process can be described as a two-player zero-sum game where the generator tries to capture the distribution of real-life data and generates synthetic samples while the discriminator tries to identify whether the given image is real. In practice, cycles of training for discriminator and generator alternate. In addition, some research was done on simultaneous updates for both generator and discriminator[41].

The GAN objectives are captured in the value function V(G,D):

$$\min_G \max_D V(G,D) = E_{x \sim p_{\text{data}}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{1}$$

where x is a data point from the dataset p_{data} is the distribution of the dataset, and z is a noise vector in a latent space p_z.

The Figure 3 shows the standard GAN architecture. The original stopping criterion for training was 50% accuracy of the discriminator, indicating that the discriminator can no longer distinguish between real and generated data.

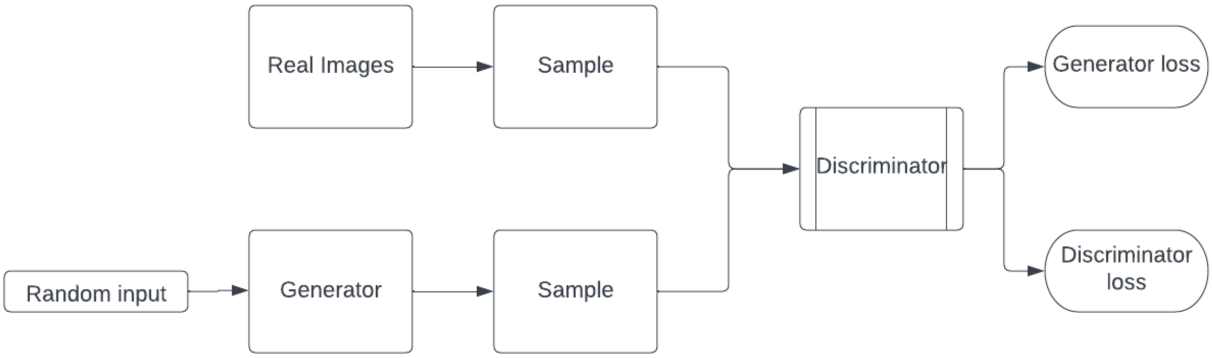


Figure 3: GAN architecture.

Conditional GAN An idea of a conditional generative adversarial network was proposed shortly after the original GAN paper by Mirza and Osindero[42]. The main idea is to condition both the generator and discriminator on some additional information **y**. The simplest example of such **y** would be a class label. **y** is added to both generator and discriminator as an additional label. The Figure4. Shows a simple conditional adversarial network. Multiple papers were published that improved the original cGAN in one of the following ways: change in the architecture of the generator, change in the architecture of both generator and discriminator, or adding extra regularization and changes in the loss function[43][44][45].

Common Problems There are a number of common problems that might be encountered during GAN training. There are cases when the discriminator is too good, which leads to generator training failing due to vanishing gradients[46]. Another problem concerns the nature of a generator. The generator network tries to generate the most plausible image, which can lead to a situation where the generator produces an especially plausible output and, as a result, starts producing only that output. This form of failure is called mode collapse. Convergence of GAN is another problem. The initial goal of training is to reach 50% accuracy for the discriminator, which would indicate that the generator succeeds perfectly. If the training is continued after that point, that might lead to completely random feedback from the discriminator and a subsequent quality collapse of the generator.

Performance measure The performance of GAN can not be reliably estimated by looking at generator and discriminator losses. To evaluate the performance of GAN, the researchers proposed a

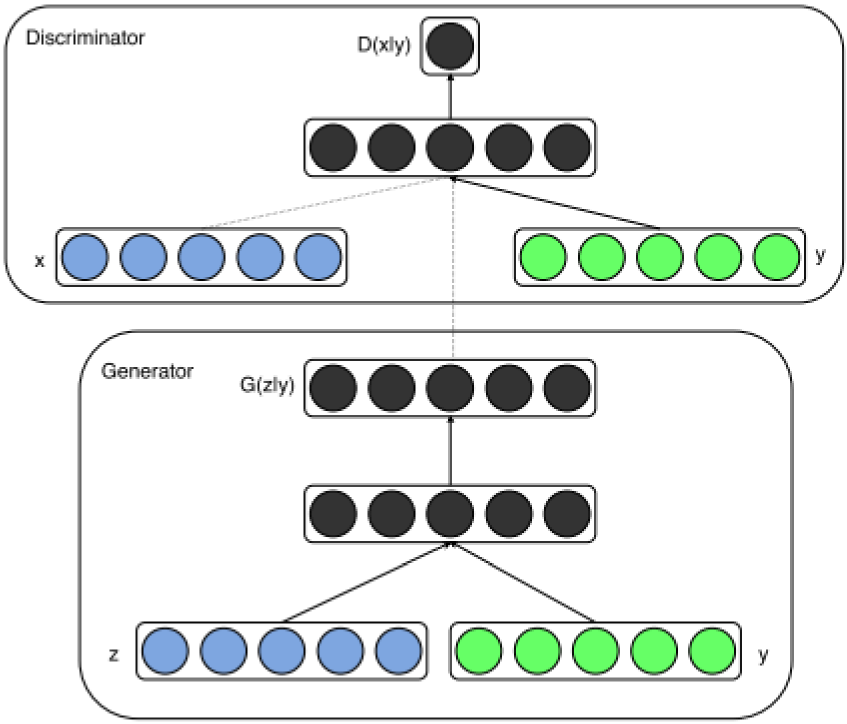


Figure 4: Conditional adversarial network. Taken from [42].

number of scores. The most commonly used performance measures are Inception Score(IS)[47] and Fréchet Inception Distance(FID).

Inception score uses two criteria: the generated images' quality and diversity. To calculate IS pretrained Inception v3[48] model is used. Generated images act as an input to the network which classifies those images. If the image is classified strongly as one class over other classes, it is considered to have high quality. The probability distribution of all generated images is estimated to estimate the diversity, and a more diverse network is expected to have a probability distribution split over a large number of images. Two of that criteria are combined to calculate IS.
Just as IS, the FID score also uses the Inception network, but unlike IS, it extracts features from the intermediate layer. Using those features, data distribution is modeled using multivariate Gaussian distribution.

$$FID(x,g) = ||\mu_x - \mu_g||_2^2 + Tr(\sum_x + \sum_g - 2(\sum_x \sum_g)^{1/2}) \tag{2}$$

where x is the real images and g is the generated images, Tr - sums up all the diagonal elements.

The lower the FID score is, the better the model.

2.3.2 StyleGAN

Generating high-resolution images is a difficult task due to the discriminator improving at a faster rate than the generator. That happens because more information is present in the image compared to information extracted from the features. Another problem is the smaller size of the mini-batches used during training due to memory constraints of the GPU. That leads to an even bigger instability during the learning.

Karas et al. modified the generator architecture in the following way: the generator always starts with constant input and modifies the "style" of the image at each convolutional layer[49]. That leads to a separation of high and low-level features. A detailed overview of the architecture of the network can be inspected in Figure5.

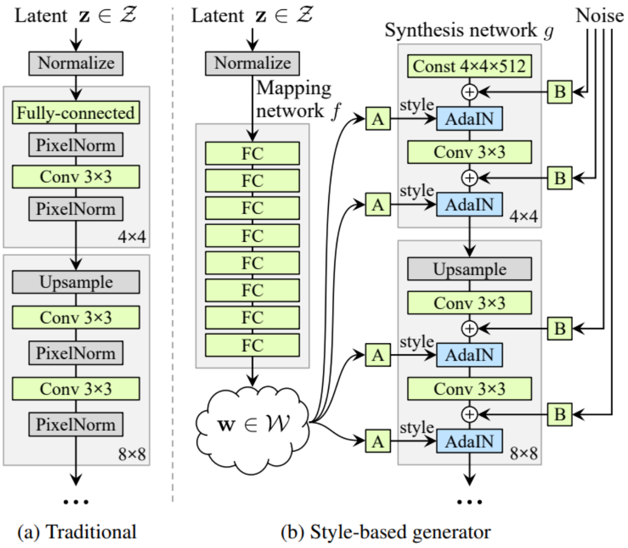


Figure 5: Traditional generator (a) and StyleGan generator (b). Taken from [49].

Latent space Traditionally latent code vector serves as an input layer of the generator. In StyleGAN network input layer is represented by a constant value. A latent vector, instead of serving as an input to the generator, is used as an input to an auxiliary mapping network f: $Z \longrightarrow W$. Resulting latent vector **w** serves as a base for a style transfer. Style is fed into the generator network at different layers, with the initial layer's style being responsible for higher-level features while later layers are responsible for smaller details. According to the authors of the StyleGAN paper, latent space W is less entangled than the original latent space Z which leads to a potentially easier modification based on the latent state vectors. Given the sufficiently disentangled latent space, it should be possible to find directions in the latent space that would correspond to meaningful and controlled variations.

StyleGAN2 StyleGAN2[50] is a follow-up paper on StyleGAN, which introduced various improvements to different parts of the architecture. Changes between architectures can be seen in Figure6. Changes from StyleGAN to StyleGAN2 architecture can be inspected in Figure6.

Two important modifications from StyleGAN to StyleGAN2 include moving bias and noise vector outside of the style module and changing instance normalization operations(AdaIN in Figure 6) to demodulation technique. The former leads to a more predictable result, while the latter removes some of the artifacts while retaining full controllability.

StyleGAN2-Ada StyleGAN2-Ada version was developed mainly to improve training with limited data[51]. However, trying to train GAN with a limited amount of data often leads to discriminator overfitting. Furthermore, the usual approach of dataset augmentation that is used as a standard solution for training with small datasets is not applicable for GAN training since it causes the generator

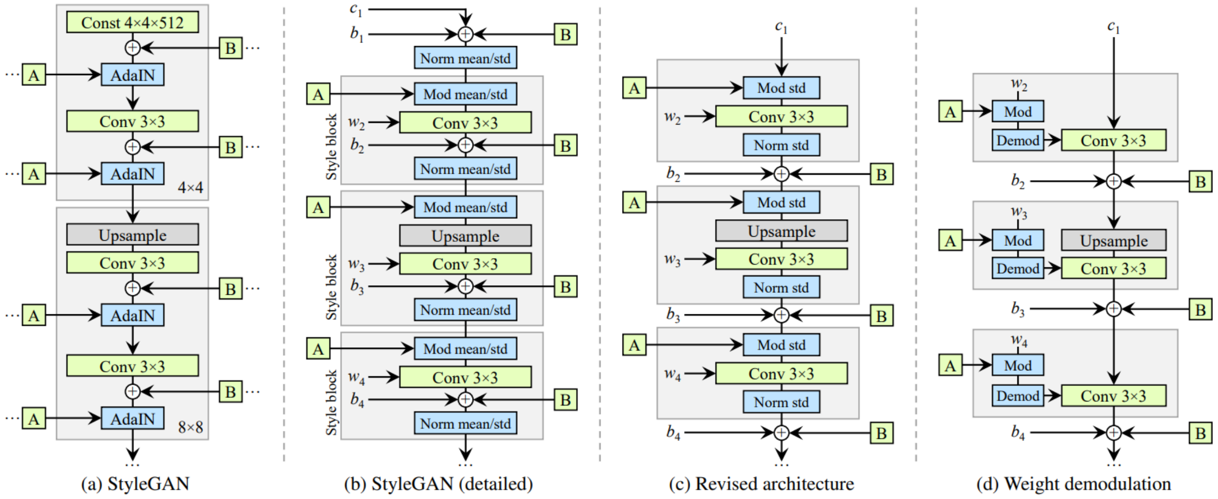


Figure 6: Changes from StyleGAN to StyleGAN2. Grey blocks are the ”style” block of the network. The main improvement between architectures came from moving bias(b) and noise(B)) outside of the style block. Taken from [50].

to learn augmented distribution. In their paper, Karras et al. demonstrated how to use augmentations while preventing augmentation leaks to generated images.

The main idea is to train discriminators only on augmented images. This approach was named stochastic discriminator augmentation. It was shown that while training under these conditions, the training implicitly undoes the corruptions and learns the correct distribution[52]. Such augmentations are called non-leaking. Karras et al. applied a maximally diverse set of augmentations and fine-tuned each augmentation strength using adaptive control for their augmentation pipeline.

2.4 State of the art

A few years after the introduction of GANs by Ian Goodfellow, powerful versions of GAN capable of generating high-resolution and high-quality images were created. An example of those networks would be StyleGAN[49][51] and BigGAN[53]. However, models themselves do not provide many control options over the image content. Several methods to add control were developed that could be split into the following categories: supervised learning of latent directions[54][19][55][56] or training with labeled images[57][58][42]. There are some papers that do not fit into either two of those categories as well. This project will mainly use the following methods: one is based on ”Interpreting the Latent Space of GANs for Semantic Face Editing”[19], second is based on ”GANSpace: Discovering Interpretable GAN Controls”[18].

2.4.1 InterFaceGAN

Shen et al.[19] proposed a method to study semantics encoded in the latent space of the generative models. The work was focused on interpreting representations learned during the training of PGGAN[59] or StyleGAN[49] trained on CelebA-HQ dataset[59]. The study aimed at manipulating the semantic attributes of a given image.

According to Shen et al.[19], other studies aiming at a similar goal typically included such additional steps as designing new loss functions, adding extra labels, and training new models. Their paper focused on manipulating attributes using a fixed GAN model. An example of manipulations achieved in their work can be seen in Figure 7.

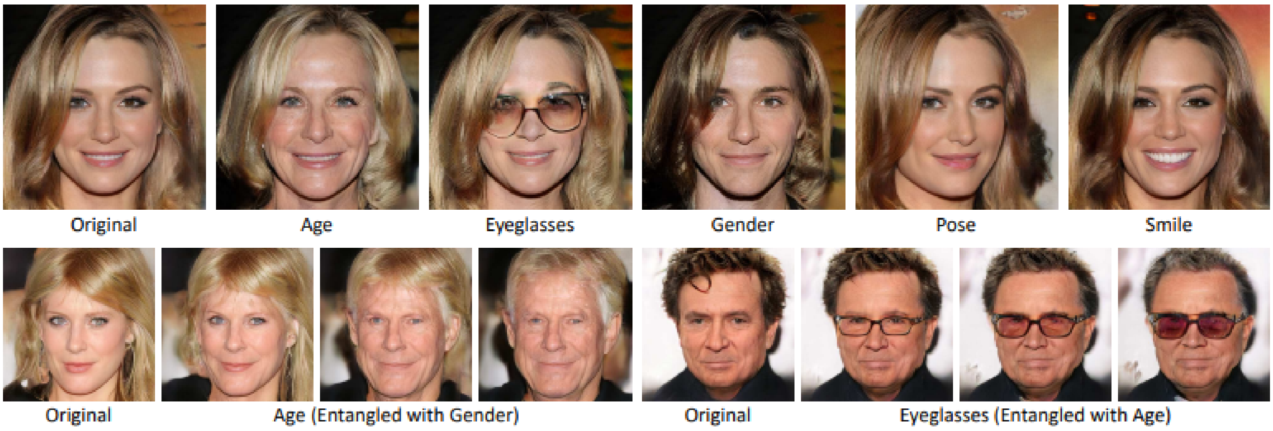


Figure 7: Example of manipulation of various facial attributes. Taken from [19].

This method is based on the assumption that for any binary semantic, there exist a hyperplane in the latent space that acts as a separation boundary. When the latent code lies on one side of the separation boundary, semantics should stay unchanged, but it should switch on the other side of the separation boundary. Distance **d** from a sample to a hyperplane was defined as:

$$d(n,z) = n^T z. \tag{3}$$

When the distance switches its sign, it indicates that the latent vector lies on the other side of a separation boundary, and the semantics should also switch. Based on that, we can define a linear relation between distance and semantic as:

$$f(g(z)) = (n,z) \tag{4}$$

where $\lambda > 0$ and indicates how fast the semantics change with the change of distance.

Shen et al. selected five facial binary attributes: pose, smile (expression), age, gender, and eyeglasses. The following procedure was performed to find the separation boundary for each attribute. First, 500K synthetic images were generated using random latent vector codes. Then, each of those images was evaluated using pretrained classifiers and assigned attribute scores for each of the semantics. Separately, for each of the attributes, 10K images with the highest score and 10k images with the lowest scores were chosen. Using those images, linear SVM was trained from which a decision boundary can be calculated. To modify the image in the direction of one of the semantics, latent codes should be changed in the direction perpendicular to the separation boundary that was calculated previously. As can be seen in Figure 7 and 8 the methods were successful for single attribute manipulation using latent space of the GAN.

The main downside of the following method is the need for a separate classifier for each of the binary attributes, which would usually entail the need for a labeled dataset to train those classifiers. Apart from that, generating 500k synthetic images and evaluating them using those classifiers can also be computationally expensive.

2.4.2 GANSpace

One of the downsides of the previous method is the need for a way to label newly generated images. Shen et al.[19] generated 500K synthesized images that were later classified for each semantic

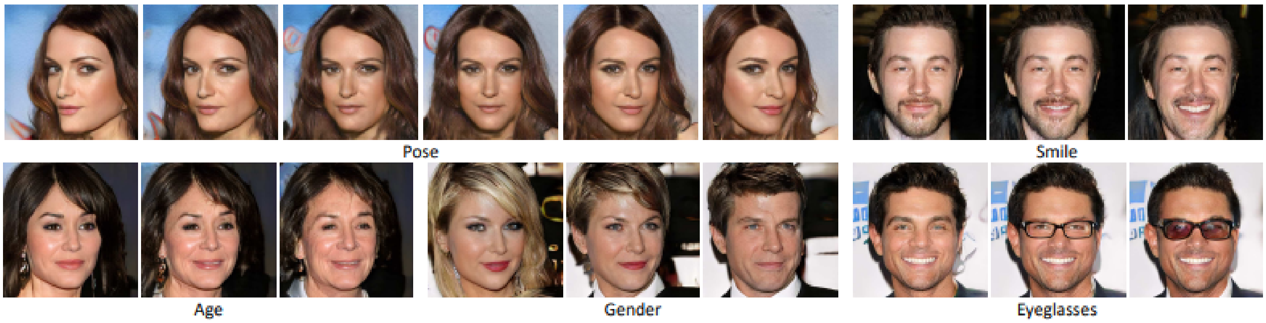


Figure 8: Another example of single attribute manipulation. Taken from [19].

boundary. That method is computationally expensive and requires separate networks to be trained to classify each semantic boundary. Training such a network would usually require the original dataset to be labeled for each condition, often requiring expert knowledge.

The approach by Härkönen et al.[18] requires no post hoc supervision or expensive optimization. One of the major discoveries of the paper is the ability to find latent space directions by using Principal Component Analysis in the latent space of StyleGAN. An example of modifications that were performed using the previously mentioned method can be observed in Figure9, One of the advantages of this method is its algorithmic simplicity. StyleGAN network takes latent vector **z** and label **c**(for conditional version of the network) as its input. Based on the **z** and **c** using supplementary network latent vector **w** is computed. The procedure to identify meaningful directions requires to sample N random vectors **z** and **c**, compute corresponding **w** vectors and apply PCA of these **w** values. As a result of PCA calculations, basis **V** is acquired for Ω latent space. To modify a new image that is defined by a latent vector **w** PCA coordinates can be adjusted based on the following formula:

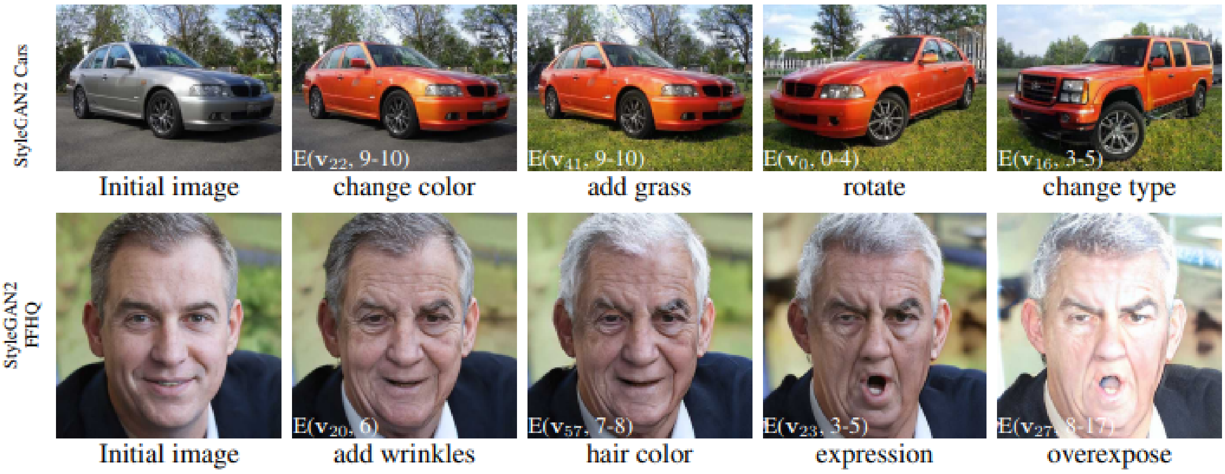


Figure 9: Example of image edits performed by Härkönen et al. Taken from [18].

$$w' = w + Vx \tag{5}$$

where **x** is a vector of the dimensions of PCA, and each entry corresponds to modifying the Principal Component of the index of said component, **w** is the original latent vector, and **V** is the PCA basis that was computed earlier.

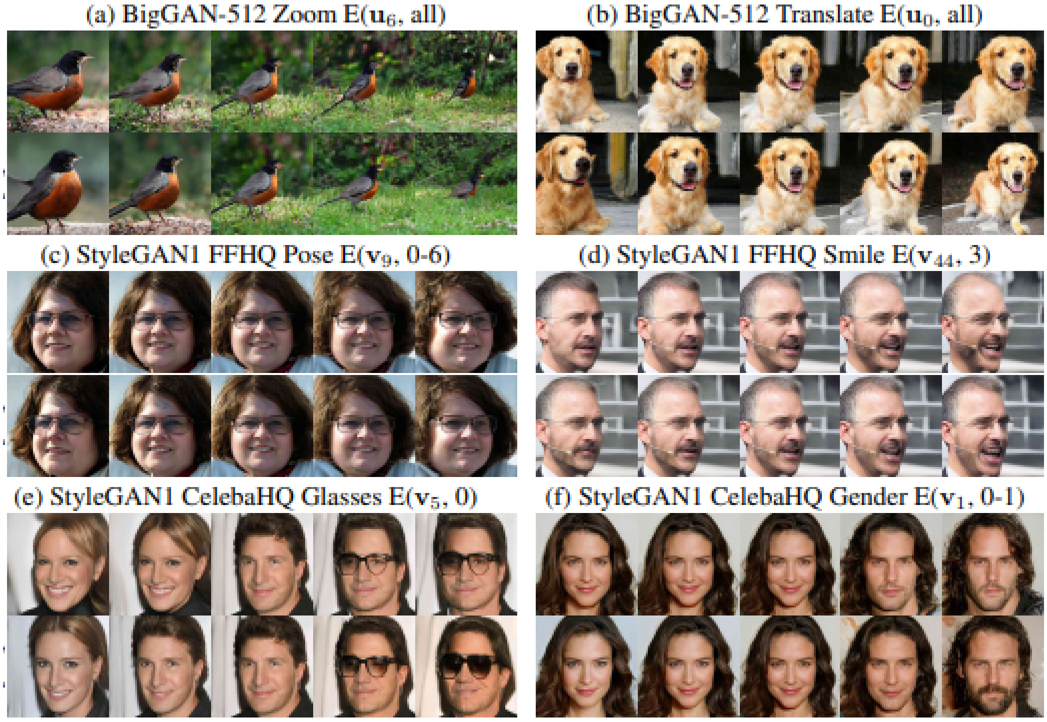

(a) BigGAN-512 Zoom E($\mathbf{u}_6$, all)
(b) BigGAN-512 Translate E($\mathbf{u}_0$, all)
(c) StyleGAN1 FFHQ Pose E($\mathbf{v}_9$, 0-6)
(d) StyleGAN1 FFHQ Smile E($\mathbf{v}_{44}$, 3)
(e) StyleGAN1 CelebaHQ Glasses E($\mathbf{v}_5$, 0)
(f) StyleGAN1 CelebaHQ Gender E($\mathbf{v}_1$, 0-1)

Figure 10: Comparison of edit directions using PCA to those achieved in other works such as[19][54]. The top row of each sub-figure shows modifications achieved using PCA, while the bottom part shows modifications from other works. Taken from [18].

With the StyleGAN style generator, it is possible to apply modifications of **w** only to some of the generator layers. According to the authors changing **w** vector for only some of the layers often leads to a more defined modification.

Härkönen et al. achieved performance comparable to other methods, such as [19][54].

The method's main advantages are its computational simplicity and no need for a labeled dataset. Of course, these methods still require some human work to identify which directions are responsible for which modifications, but overall, it demands significantly fewer human and computational resources.

3 Methods

Two main methods for latent space exploration were used for alteration of a synthesized colonoscopy image: hyperplane boundary separation method[19] and finding directions using PCA on latent vectors[18]. Results were mainly evaluated visually with additional evaluations for each method. Those evaluations are outlined in the relevant sections. Due to time and computational constraints, other methods of latent space exploration were vetoed during respective earlier stages of testing. However, suggestions of those methods and other possible methods are given in the discussion section.

The detector's performance was measured using recall, precision, F1, and MaP50 scores.

3.1 Dataset

This section describes the dataset used for this project. An anonymized dataset of N images was provided by UMCG. The dataset consists of images of varying sizes taken during a colonoscopy procedure.

3.1.1 Labeling

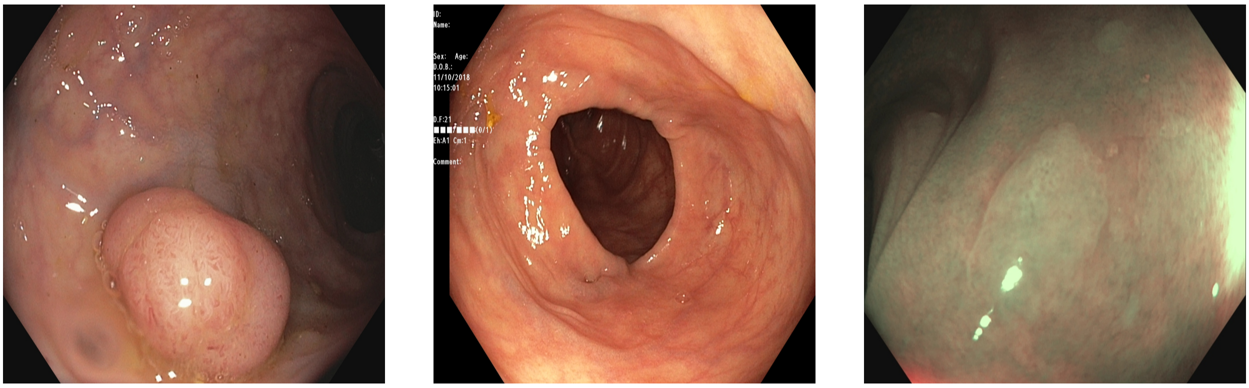
With the help of a medical practitioner, all of the original images were annotated. As a result of this annotating process from the original 10300 colonoscopy images, 2976 images were annotated as having a polyp. Images with polyp were split into several groups: singular polyp, multiple polyps, doubt polyp, and NBI polyp. Examples of polyps from some of the groups can be observed in Figure 11. The total amount of images in each group can be seen in Table1.

Group	Number of images
Singular polyp	1934
Multiple polyps	240
Doubt polyp	357
NBI polyp	445
No polyp	7324

Table 1: Number of images in each group of images.

Out of those groups, images from the singular polyp group and doubt polyp group were used for the experiments. Multiple polyp images were not used because some experimental setups, mainly training conditional versions of GAN, required one polyp in the image. Doubt polyps consisted of images that could not be confidently classified as polyps purely from images. That group often potentially included difficult to spot polyps such as flat polyps. It would be beneficial to have such images in a dataset to provide the model with a large range of images. Images from NBI polyp groups were taken with the use of a narrow-band imaging technique which involves using special filters that change the coloring of an image. Different coloring of those images could negatively affect some models, so they were excluded from the dataset.

As a result of the labeling procedure, 2291 images were selected as a primary dataset for all the experiments. For each image, a bounding box was defined by a pair of two coordinates of the top left corner and bottom right corner.



(a) Singular polyp (b) Doubt polyp (c) NBI polyp

Figure 11: Examples of images from different groups.

3.1.2 Preprocessing

Each image that was selected was resized to a resolution of 512x512. Normalization was applied to all resized images to change the range of pixel intensity value to [-1 1]. No other image augmentation/transformation was applied to the dataset since StyleGAN2-Ada has a built-in augmentation pipeline that is designed to not 'leak' into the generator. [51]

3.2 GAN training

Based on the previous work by Mihai Popescu,[16] StyleGAN2-Ada was selected as a primary generator architecture. Two different configurations were trained using that architecture: conditional generator and unconditional generator. Both conditional and unconditional networks were used for several reasons: some of the methods used later were designed only for the unconditional version of the network. Furthermore, a comparison of latent space exploration can yield potentially valuable results for methods that can be used for both conditional and unconditional networks. Some of the methods are designed only for the conditional network.

3.2.1 Unconditional generator

A set of images specified in the previous section was used as an input for the unconditional generator. For the generator and discriminator, a set of default parameters described in[51] was used. The network was trained in total for 16000 kimg where 1 kimg represents 1000 images being shown to a generator. Based on the original paper 25000 kimg was selected as an original goal of training, but due to computational complexity and lack of improvement, training was stopped earlier.

3.2.2 Conditional generator

For conditional generator as an additional input, a label in the form of a bounding box was given to a network at each step. Apart from that, the additional configuration of the network is the same as in the previous case. Finally, the network was trained for 3000 kimg to match the training of the previous work [16] and be able to potentially make a valid comparison.

The default values for hyperparameters presented in the original paper[51] were used for both conditional and unconditional generator training.

Labeling function To generate images using a conditional network apart from the random latent code **z**, a label **c** of the same dimensions as the one used during the training should be provided. In this particular example label is 4 coordinates, 2 of which are x,y coordinates of the top left corner and two are x,y coordinates of the bottom right corner. Those 4 coordinates define the bounding box of the location where the network would try to generate a polyp. For each image generated by a conditional one, a random label was created using the following rules: coordinates of the top left corner(x_1 and y_1) were chosen randomly from the range [0 400], random width and height were generated from the range [100 256], coordinates of the bottom right corner were calculated by adding width and height to the x_1 and y_1 respectively if those coordinates were large than 512 they were reduced to 512. Those parameters were chosen after some experimenting to cover the largest possible range of polyps while retaining the high performance of the generative network.

3.2.3 Network evaluation at different steps of training

Evaluation of GAN performance is an open research question with no definite answer. Moreover, since GAN performance is hard to define, it also creates a problem at the training stage, mainly: ”When should the training of the network stop?”.

Several problems, such as mode collapse, non-convergence, and instability, are often encountered during GAN training. Those problems can lead to increased difficulty or even the impossibility of latent space exploration. Therefore, latent space exploration heavily depends on having a high quality generator. Therefore, several evaluation methods were investigated to choose a network at the most suitable state to solve this problem.

In total 4 metrics were investigated for model selection.

Inception Score The Inception v3 network is a classifier model trained on ImageNet. It provides a vector of probabilities with a dimension (1000,1) as its output. Each row in the output represents the probability of the image belonging to one of the 1000 classes of the ImageNet dataset. The inception score is calculated using the Inception v3 network.

$$IS(G) = exp(E_{x \sim p_g} D_{KL}(p(y||x)||p(y))), \tag{6}$$

where x $\sim$ p_g indicates that x is an image sampled from the distribution learned from a generator, $D_{KL}(p||q)$ is the KL-divergence between the distributions p and q, p(y——x) is a conditional distribution and p(y) is marginal distribution[60]. KL-divergence is a measure of how one probability distribution differs from another and can be described as an information gain achieved if p would be used instead of q.

Inception score to codify two qualities: Inception score should be highly confident that there is a single object in an image, and the generative algorithm should output high diversity of images. In addition, it was shown that the Inception score correlates with human judgment estimations[47].

Frechet Inception Distance Second, ’Frechet Inception Distance(FID)[61]. FID is an improvement on the Inception score that sues statistics of real world samples. Authors claim that FID is more consistent than Inception Score. Unlike the inception score, the FID compares the distribution of generated images with the distribution of real images. The procedure is quite similar to calculating the Inception score. Inception v3 network trained on ImageNet is used to calculate FID. However, to calculate the FID score, both real images and synthetic images are fed into the Inception network and compare output from one of the later layers of the network between real and synthesized images.

For both IS and FID scores, 10000 synthetic images were generated and compared to ~2000 real images.

Structural Similarity Index Measure As a third metric structural similarity index measure(SSIM)[62] was used. SSIM is used to measure the similarity between two images. SSIM takes into account the luminance, contrast, and structure of the object.

$$SSIM(x,y) = l(x,y)^{\alpha} * c(x,y)^{\beta} * s(x,y)^{\gamma} \tag{7}$$

Where α, β, γ are the weight of the each component, they are equal to 1 in this case.

For each state of the network 250 images were generated and SSIM was calculated for each possible pair of images. An average SSIM was used as a metric, with a lower number indicating better diversity of the generated images.

YOLOv5 evaluation Finally, as the last method, the YOLOv5 detector system was used to detect a polyp on each of the generated images. As a result of such evaluation, a bounding box from the detector and confidence score was acquired. For images with a confidence score larger than 0.22 location of the polyp was calculated using bounding boxes. In total, for each state of the network 1500 images were generated by the StyleGAN2 network and evaluated by the YOLOv5 network. As an intermediate metric, each polyp was assigned to either top or bottom half of the image based on the location of its center. The same was done for the left and right half of the image. After all the manipulation for each state of the network, the following values were calculated: mean and standard deviation of size of the polyp, x and y location of the center, proportion of images in left and right half, the proportion of images in top and bottom. The network that produced images with the higher confidence score and equal left/right and top/down split was considered a better network.

$$\text{YOLOv5 score} = (s1 + s2 + s3)/3 \tag{8}$$

where

$$s1 = 1/(\frac{max(l,r)}{min(l,r)} + \frac{max(t,d)}{min(t,d)} - 1) \tag{9}$$

where **l** - the number of polyps detected by YOLOv5 that have their center x coordinate smaller than 255 and confidence higher than 0.2.
where **r** - the number of polyps detected by YOLOv5 that have their center x coordinate larger than 255 and confidence higher than 0.2.
where **t** - the number of polyps detected by YOLOv5 that have their center y coordinate smaller than 255 and confidence higher than 0.2.
where **d** - the number of polyps detected by YOLOv5 that have their center y coordinate larger than 255 and confidence higher than 0.2.
s1 is 1 if both l = r and t = d, and goes towards to 0 if the ratio of polyps increases.

$$s2 = \frac{\text{images with confidence} > 0.2}{\text{total number of evaluated images}} \tag{10}$$

s2 can take values between 0 and 1 depending on the number of images that had confidence higher than a threshold.

$$s3 = \frac{1}{\ln(256 + |x\_centre - 256|)/256 + \ln(256 + |x\_centre - 256|)/256 + 1} \tag{11}$$

where x_centre is the average coordinate of the polyps detected by YOLOv5 that had confidence higher than 0.2
where y_centre is the average coordinate of the polyps detected by YOLOv5 that had confidence higher than 0.2
If the center of the polyps is near the 256 mark, then ln of that will be close to 0, resulting in **s3** being close to 1.

3.3 Latent space exploration

3.3.1 Manipulating the latent space using a support vector machine

Latent Space Separation For this experiment, 4 binary attributes were chosen: horizontal location of the polyp, the vertical location of the polyp, size of the polyp, and confidence score from YOLOv5 evaluation. A total of 100000 images were generated using the StyleGAN2-Ada generator. Each image generated by the StyleGAN network was evaluated by the YOLOv5 detector network, which assigned to each image a bounding box of the polyp and the confidence of the network that the polyp is present in that bounding box. Only images with a confidence score higher than 0.3 were used for SVM training, while the rest of the images were discarded. From a total 100000 for each binary attribute, 20000 images were selected that can be split into two groups. Images belonging to one of the groups lie on one side of the hyperplane and have as high an attribute score as possible. In contrast, the other group includes images with attribute scores being as low as possible.

In other words, to calculate hyperplane boundary, two groups of images were used that have as high of a difference in their average attribute score as possible. For the calculation of the hyperplane, latent vector **z** or ω were used as samples, and previously calculated binary attribute scores were used as labels.

To confirm the initial hypothesis of such a hyperplane existing in the latent space, each of the trained SVM was tested on a held-out validation dataset. Held-out validation dataset consisted of 10000 images that were evaluated by YOLOv5 and had a confidence score higher than 0.3. The same procedure was performed for each version of the network that was used in the experiment.

Disentanglement analysis Disentanglement analysis was performed using hyperplane boundaries to assess the correlation between the different semantics. As a measure of disentanglement angle between normal vectors to the hyperplane boundaries was calculated.

$$Angle = \arccos(n_1 n_2^T), \text{ where } n_1 \text{ and } n_2 \text{ are normal vectors to their respective boundaries}$$

An angle close to 90°means that the two attributes are almost independent of each other. The more angle deviates from 90°in either direction, the more correlated attributes are. For example, the angle being between [0°,90°) implies positive correlation while the angle between (90°,180°] implies negative correlation.

Latent Space Manipulation Manipulation of single binary attributes was done by modifying the initial latent vector(z or ω depending on the experiment) in the direction perpendicular to the separation boundary found for that attribute. Both z and ω latent vectors were modified for unconditional networks. For the conditional network, only ω latent vector was modified since for the conditional network on top of the latent vector z. The bounding box was also given as an input introducing an additional parameter that didn't fit into hyperplane calculation. On the other hand, the ω latent vector is calculated using the original latent vector z and the bounding box of the label; hence an additional parameter of the location of the polyp is not present at that stage of the algorithm. A vector perpendicular to a hyperplane can be pointed in opposite directions. Changing an attribute along with one of those vectors positively affects the binary attribute, while changing the attribute alongside the opposite vector leads to negative changes in the attribute.

Evaluation of Latent Space Manipulation Evaluation of how successful manipulation is done in two ways: visual inspection of the modification of random images or evaluation based on the YOLOv5 detector. For visual inspection, an image is generated from a random latent vector, and that vector is modified based on the decision boundary. Since it is not clear how strongly would the original vector be modified, a number of distances were tried for each modification. After modifying an image in both directions, including intermediate steps, by visually inspecting the results, a conclusion is made if the modification of the attribute is successful or not. A total number of M images is inspected, and the percentage of the images with visible modifications is used as a metric.

One of the problems of the previous approach is the fact that it is highly subjective. As a result, a more objective method is tried to evaluate latent space. This approach is based on the YOLOv5 detector system. To evaluate the modifications, the following set of steps were performed for each network and boundary:

1. Generate an image from a random latent vector for unconditional network or random latent vector and bounding box of the preferred location of polyp for conditional network

2. Evaluate that image using the YOLOv5 detector system

3. Keep the image if the confidence score of YOLOv5 evaluation is larger than 0.3; otherwise, go back to step 1

4. Modify original latent vector in both directions by the chosen distance **D**

5. Generate intermediate images between both modified images and original images. In total 9 images are generated, which go from -D to D modifications of the latent vector with the same distance step between each two image

6. Calculate the difference between the chosen binary attribute between each generated image and the original image

7. Repeat steps 1-6 until a 500 images are modified and evaluated and compute an average change in the chosen attribute

During the exploration stage, a sweep across different values of D was done to find a range of values in-between which the manipulation does not degrade the image. Sweep was done using the following set of values for D: 1,2,4,8,12,16,24,32. As a final value for calculations, distance D = 20 was chosen.

As an outcome of the described algorithm, an average change for each distance change in each binary attribute is acquired.

Experiments were performed with a conditional and unconditional version of the network.

3.3.2 PCA

As the first step of the following methods, principal components are computed. To do so, random 100000 vectors $z_{1:N}$ and 100000 random labels $c_{1:N}$ where c is generated using the labeling function 3.2.2. Next, using z and c, corresponding ω latent vectors $w_{1:N}$ are computed. Finally, using ω latent vectors, PCA is computed, which gives a basis **V** for ω latent space.
Modifications of the image were performed using the equation 5.

PCA variation explained As a first step number of dimensions of the latent space that are relevant for image synthesis are calculated. Explained variance is part of the PCA calculations and can be represented as a function of the ratio of related eigenvalues and the sum of all eigenvalues.

Investigating effects of different principal components The influence of modifying certain principal components is investigated at first. 4 different steps were performed in the first experiments:

- Fix first 8 PCA coordinates, randomize remaining
- Randomize the first 8 PCA coordinates, fix the remaining 504
- Fix 8 random PCA coordinates, randomize the others
- Randomize 8 random PCA coordinates, fix the remaining 504
- possibly something else

Meaningful modifications using PCA method 11 images were chosen as candidate images for modifications. To each of those images procedure described below was applied:

- 20 directions based on the first 20 PCA components were calculated
- Modifications in those 20 direction with values from the list:[-2 σ, - σ, σ, 2σ] were computed, where σ is a variance of that particular principal component across the whole generated dataset
- Those modifications were applied to each image in 5 different ways: to all layers of w vector, to 0-2 layers, to 0-3 layers, to 3-6 layers, to 6-16 layers.
- All modified images were saved and inspected to attempt to identify each of the modifications semantically.

3.4 Detector training with an augmented dataset

The YOLOv5 model was used as a detector for this part of the project. The model was trained 5 times with a real dataset and 5 times with an augmented dataset, with each run training for a 100 epochs. Multiple runs were done to diminish potential stochastic differences during the training.

The model's performance was evaluated by using average precision(mAP), accuracy, recall, and F1 scores. Default hyperparameters and YOLOv5s checkpoint were used, more details can be found in [33]. A separate dataset was used for testing the final version of the network. Real and/or fake images were split into training/validation sets in a proportion of 80%/20%.

Human evaluation was done following this procedure: one medical practitioner labeled synthetic images by drawing a bounding box around the polyp(if presented) and assigning an image to one of the following groups: high/low confidence polypoid, high/low confidence flat polyp, image with artifacts, no polyp image.

To evaluate the effect of adding synthetic images to a dataset mAP of the YOLOv5 detector trained on the original dataset was compared to the mAP of the detector using an augmented dataset that consisted of original images + synthetic images belonging to one of the following groups: high/low confidence polypoid, high/low confidence flat polyp.

4 Experimental Setup

The task of manipulating the latent space of the generative network was carried out in a structured pipeline involving various GAN models, different methods of latent space exploration, and different evaluation techniques. Different sections of a pipeline are presented in the following sections. The task of latent space manipulation requires a generative model to be trained, and depending on the method of latent space manipulation, a form of a detector/classification network is also needed. The generative model used to generate synthetic images was StyleGAN2-Ada[51]. It was picked due to the previous results[16]. Two versions of the network were trained for the purpose of this project: conditional and unconditional generators. The choice of training, both conditional and unconditional versions, is motivated by the potential changes in latent space structure. As a detection model, YOLOv5[33] architecture was selected. The choice was made due to the fast inference speed of the detector and the existence of pretrained weights, which would reduce the time to train the model.

4.1 Tools and Technologies

The models were developed using Pytorch or Tensorflow libraries. In particular, StyleGAN2-Ada models used Pytorch, while the YOLOv5 detector used Tensorflow. Standard data science and machine learning libraries such as CV2, scikit-learn, Pandas, and NumPy were used for preprocessing and data manipulation. Experiments were performed either on one NVIDIA V100 GPU that belongs to the Peregrine cluster of the University of Groningen, NVIDIA GTX 3080 on a home desktop or using Google collaborator, which provides either NVIDIA K40 or P100. A different instance of the environment was used due to varying difficulty of the setup of the different experiments where Peregrine cluster was used for the longest experiments that would require a running time of several days while home desktop and google collaboratory was used for shorter experiments or exploratory experiments. All of the instances were single GPU instances, limiting the exploration that could have been done for the study.

4.2 Experimental Configurations and Performance Criteria

4.2.1 StyleGAN training

Two versions of the StyleGAN2-Ada network were trained for the project: conditional and unconditional. For the conditional version of StyleGAN as an additional input, a location of the polyp in the real image is provided to the network. It is done to enable the possibility of specifying the bounding box of a polyp as an input to a generator so the generated image would have a polyp in that bounding box.

Unconditional network Evaluating the performance of the generative network is an open question in the research community. Due to that, it was unclear at which point the network should stop training. The network was trained for 16000 kimg(where kimg is defined as 1000 images from a dataset shown to the generator), and for each 1000 kimg, a checkpoint was saved, which resulted in 16 checkpoints. Multiple checkpoints were tested due to the fact that after visual inspection of the said checkpoint, it seemed that at some points network preferred to generate only images of a certain kind which would reduce the potential of latent space exploration methods. An assumption was made that network that generates less diverse images would have less defined latent space. As a performance measure for this part of the project, a combination of FID score, IS score, SSIM, results of YOLOv5 detection, and visual inspection was used. First, all of the checkpoints that lost a pairwise comparison to other checkpoints were removed from the set of candidate models. Then, using the remaining networks, some images were generated using each network which then was visually inspected to reduce the number of candidate models to 3, which were used for the latent space exploration part of the project. A pipeline for unconditional network training/evaluation can be observed in Figure 12.

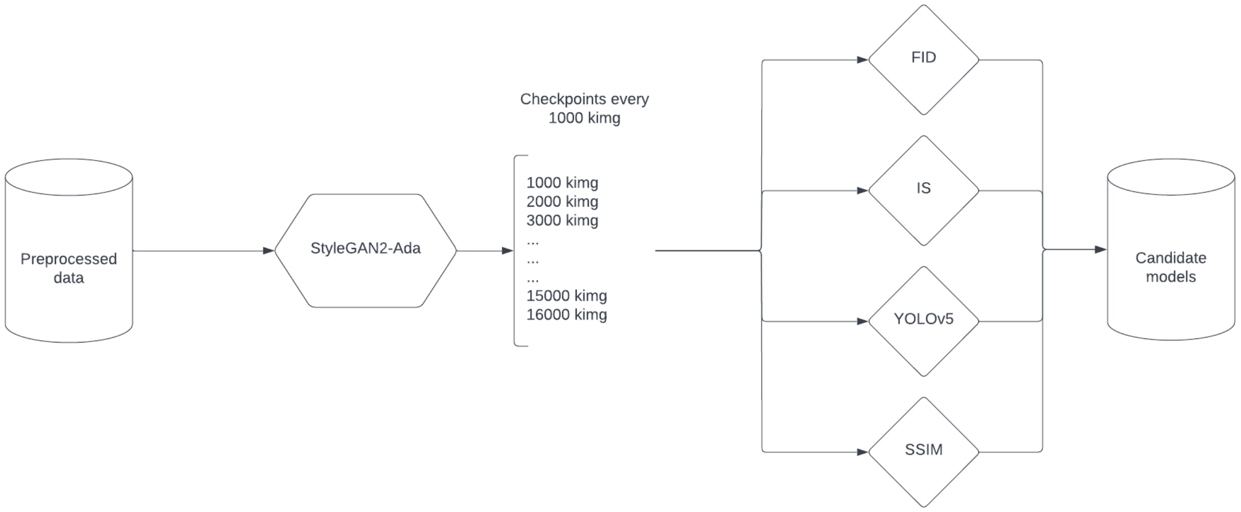


Figure 12: Experimental setup for training and selecting unconditional StyleGAN architecture. The model was trained for 16000 kimg and was evaluated using 4 different methods defined in 3.2.1.

Conditional network Since the diversity of generated images is less of a problem for conditional networks due to the fact that we specify a location in which the polyp is to be generated by giving a bounding box to a network, only the FID score was used as a selection mechanism. The training was stopped as soon as there was no improvement in the FID score for 400 kimg shown to the generator.

4.2.2 Latent space exploration using a support vector machine

Evaluation of the latent space exploration using a support vector machine can be split into several steps: finding the separation boundary, evaluating the separation boundary itself, and evaluating the modified images.

To calculate the separation boundary, a following set of operations is performed: a number of images are generated using the StyleGAN generator model, those images are fed into a pre-trained YOLOv5 detector as an output bounding box of the polyp, and the confidence of the network that polyp is present in that bounding box is acquired, using coordinates of the bounding box, and confidence binary attributes are assigned to each image(attributes being horizontal, vertical locations, size, and quality) while images with low confidence for the first 3 attributes are filtered out. Then, using those attributes as labels and depending on the experiment, either **z** or ω latent codes as a list of features, a separate SVM is trained for each of the binary attributes. As a result of SVM training, a separation boundary is acquired for each of the attributes. The schema of the process can be seen in Figure 13.

Each trained SVM is evaluated on a held-out dataset to confirm the performance of a separation boundary on previously unseen images.

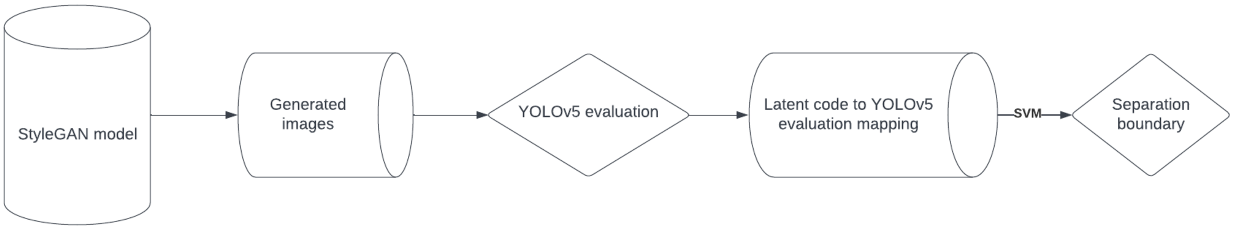


Figure 13: Schema of the pipeline for separation boundary calculation. Both conditional and unconditional versions of StyleGAN can be used as StyleGAN models, but only calculations using ω latent space are possible for the conditional version.

For the evaluation step, a number of images are generated and then modified. Original and modified images are evaluated using the YOLOv5 detector, and the average change in the location of the center of the polyp is calculated. Schema of the process can be observed in Figure 14.

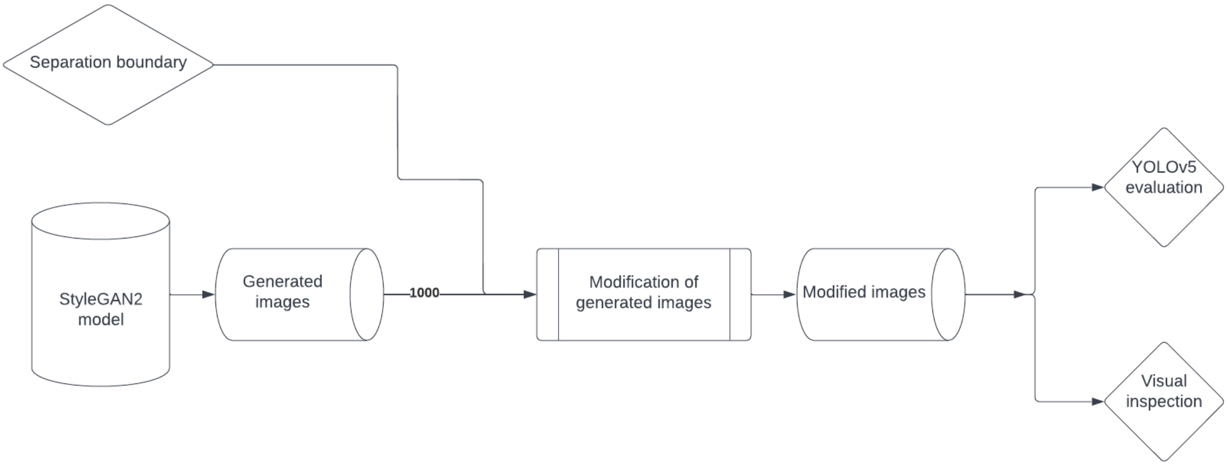


Figure 14: Schema of SVM evaluation pipeline.

4.2.3 PCA based method

This method can be split into two parts: calculating PCA and modifying images using principal components. For the first part, the following procedure is performed: N random z latent vectors and labels are generated, mapping network of StyleGAN is used to calculate ω latent vectors. Then, using ω latent vectors, PCA is computed. After PCA is computer basis **V** is acquired and can be used to transform latent vectors into principal components coordinates. Finally, latent vector directions corresponding to individual principal components are computed in ω latent space to modify the images. By changing the image in one of those directions, images can be altered. The schema of the pipeline is depicted in Figure 15.

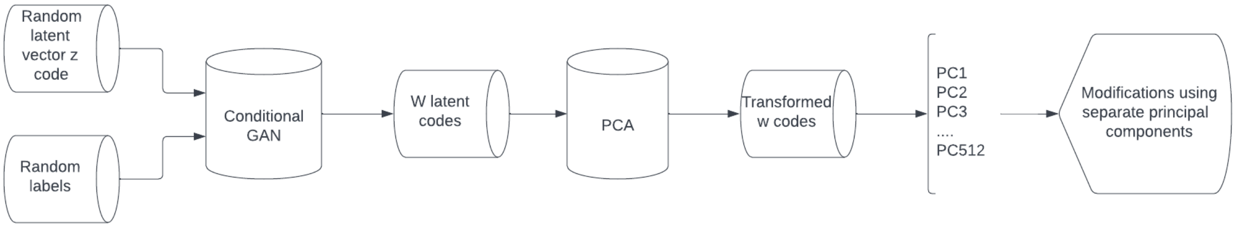


Figure 15: Schema for the pipeline for PCA-based method.

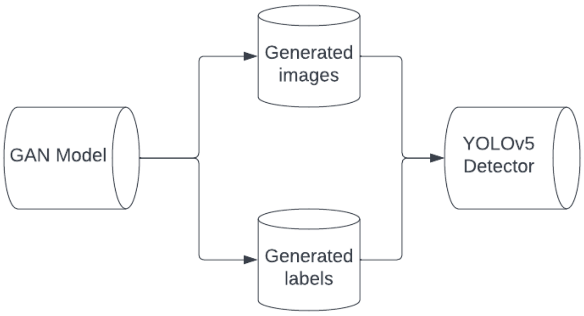


Figure 16: Schema for the detector training.

4.2.4 Detector training

The procedure for detector training is relatively straightforwards. To train the detector with synthesized images, said images are generated using the StyleGAN network. Those images are labeled either with the help of a medical practitioner or by using the labels given to the network if a conditional StyleGAN network is used. Several modes of training were tested during the project: training only with real data, training with a combination of real data and synthesized data, and training only with synthesized data.

5 Results

5.1 Network evaluation

The following section showcases the results of a model selection process.

5.1.1 Unconditional GAN

The network was evaluated using 4 different scoring techniques that were described in the methods section. Results of the evaluation can be observed in Table 2.

kimg	IS	FID	SSIM	YOLOv5 evaluation
3000	1.0011	17.5	0.733	0.432
4000	1.0008	19.7	**0.719**	0.368
5000	1.0009	16.15	0.748	0.380
6000	1.0011	16.63	0.774	0.462
7000	1.0010	18.87	0.802	0.445
8000	1.0011	19.62	0.788	0.368
9000	1.0010	18.93	0.786	0.384
10000	1.0011	18.77	0.793	0.419
11000	1.0013	17.05	0.805	0.476
12000	1.0009	18.19	0.772	0.430
13000	1.0016	17.87	0.751	0.459
14000	1.0014	17.60	0.778	0.477
15000	1.0011	**15.92**	0.794	**0.520**
16000	1.0011	17.13	0.795	0.473

Table 2: Results of evaluating network at different training steps.**kimg**: number of thousands of images. **IS**: Inception Score.**FID**: Frechet Inception Distance. **SSIM**: Structural Similarity Index Measure. **YOLOv5 evaluation**: a score based on YOLOv5 evaluation calculated as described in Equation 8.

5.2 Latent space exploration

5.2.1 Hyperplane separation method

As the first step of the hyperplane separation method and evaluation of the separability assumption. Table 3 shows the results for ω latent space for the conditional version of the network while Table 4.

Similar calculations were also performed for the boundaries computed using the latent space of the unconditional StyleGAN network. Table 4 summarizes the results.

Disentanglement analysis From the computed boundaries disentanglement analysis was performed. It was done by finding the angle between two perpendiculars to a boundary. Table 5 reports correlation metrics between boundaries calculated using the conditional network.

Attribute	Accuracy full dataset	Accuracy test dataset	Accuracy of 20% most removed
Horizontal location	87%	87%	96%
Vertical location	80%	79%	94%
Size	76%	76%	84%
Quality	75%	75%	83%

Table 3: Classification accuracy on separation boundaries in ω latent space for the conditional network.

Attribute	Accuracy full dataset	Accuracy test dataset	Accuracy of 20% most removed
Horizontal location	65%	66%	92%
Vertical location	73%	75%	87%
Size	66%	68%	82%
Quality	71%	71%	86%

Table 4: Classification accuracy on separation boundaries in ω latent space for the unconditional network.

Semantic manipulation Using boundaries computed for each attribute each of the boundaries was evaluated using the procedure defined in 3.3.1. Results are summarized in Table 6. Examples of some of the manipulations can be seen in Figures 17, 18 and 19.

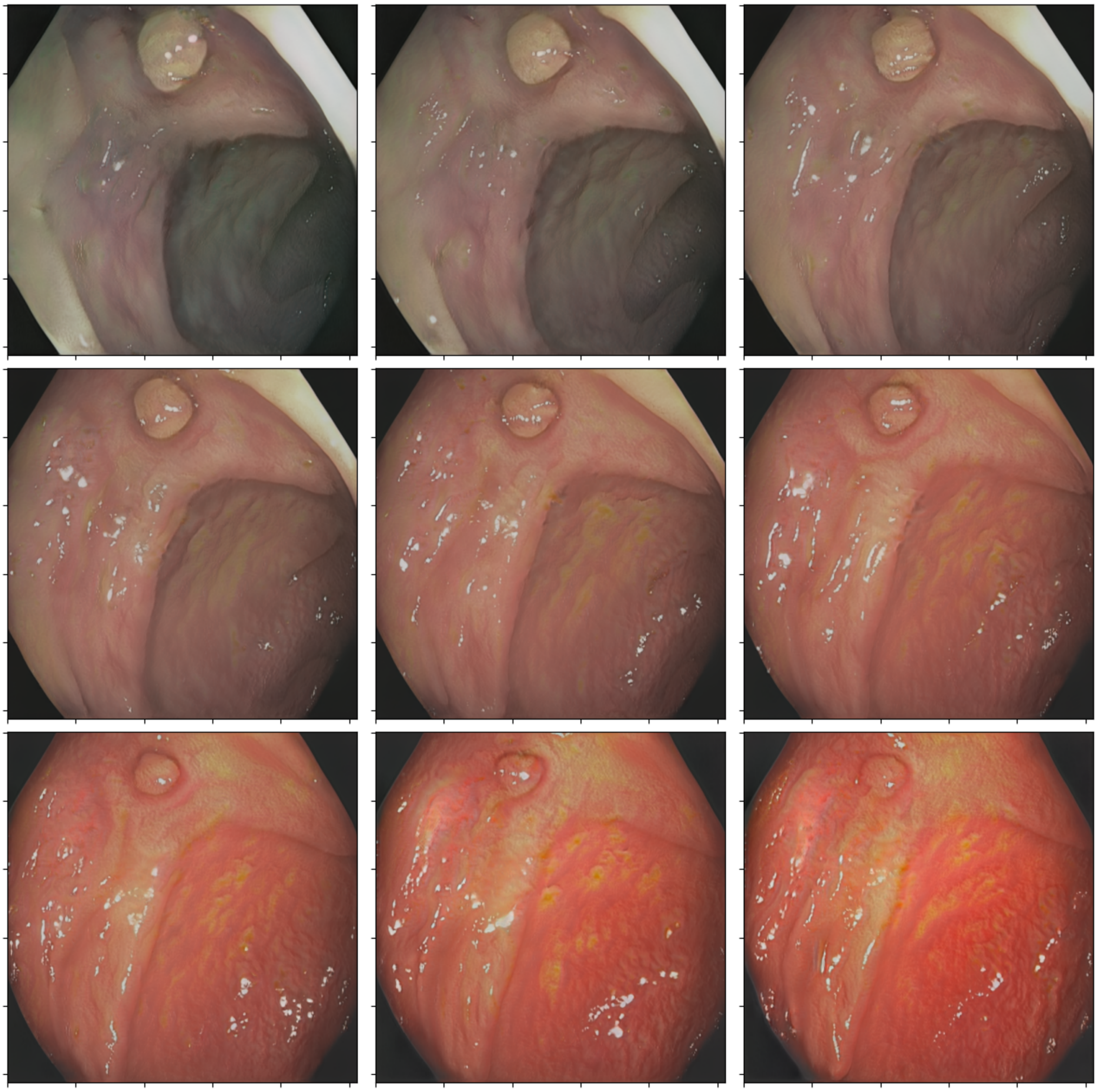


Figure 17: An example of quality boundary change using conditional network. Middle image is the original one. Going left represent the change in one direction while going right in the other.

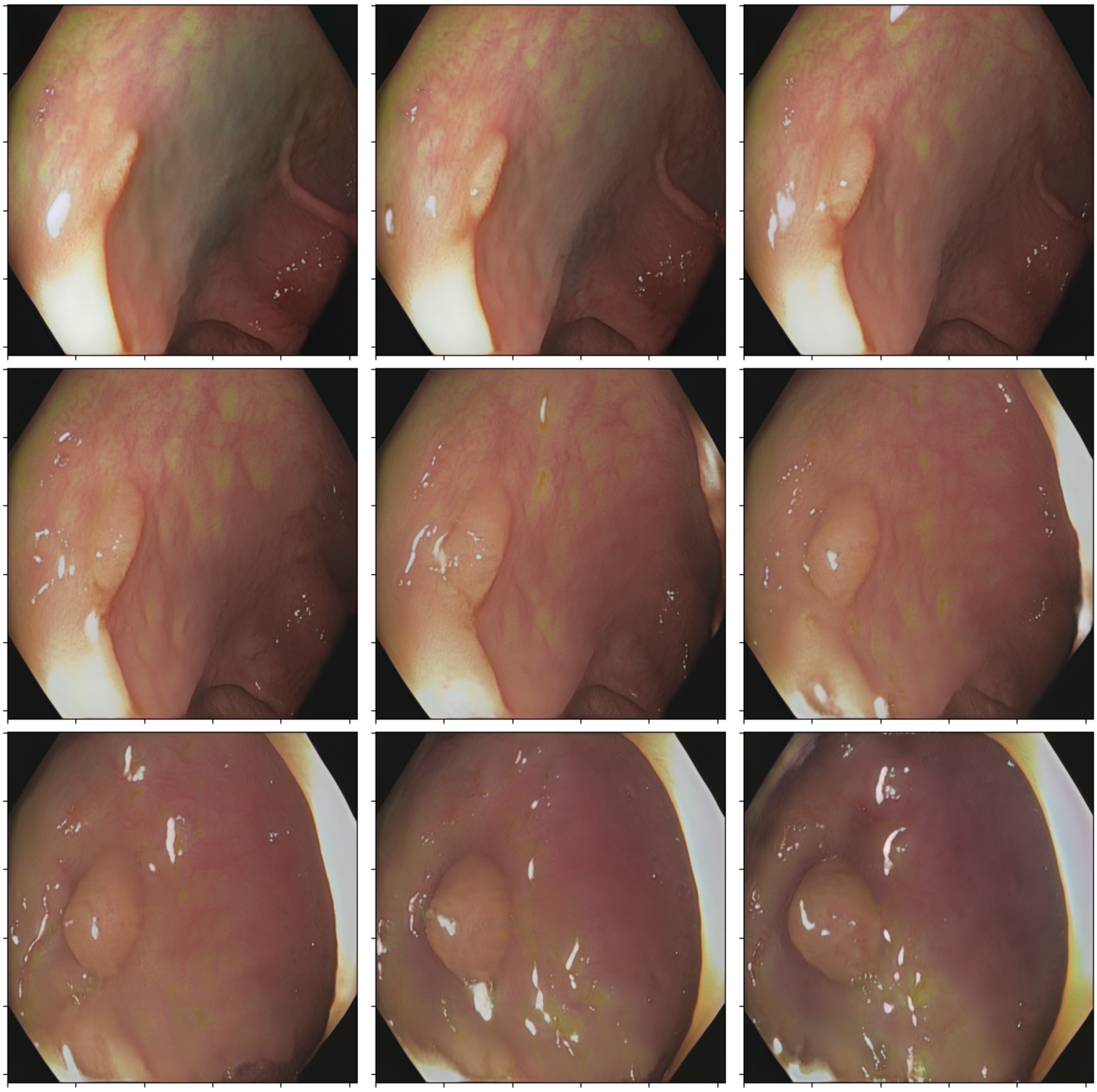


Figure 18: An example of size boundary change using unconditional network. Middle image is the original one. Going left represent the change in one direction while going right in the other.

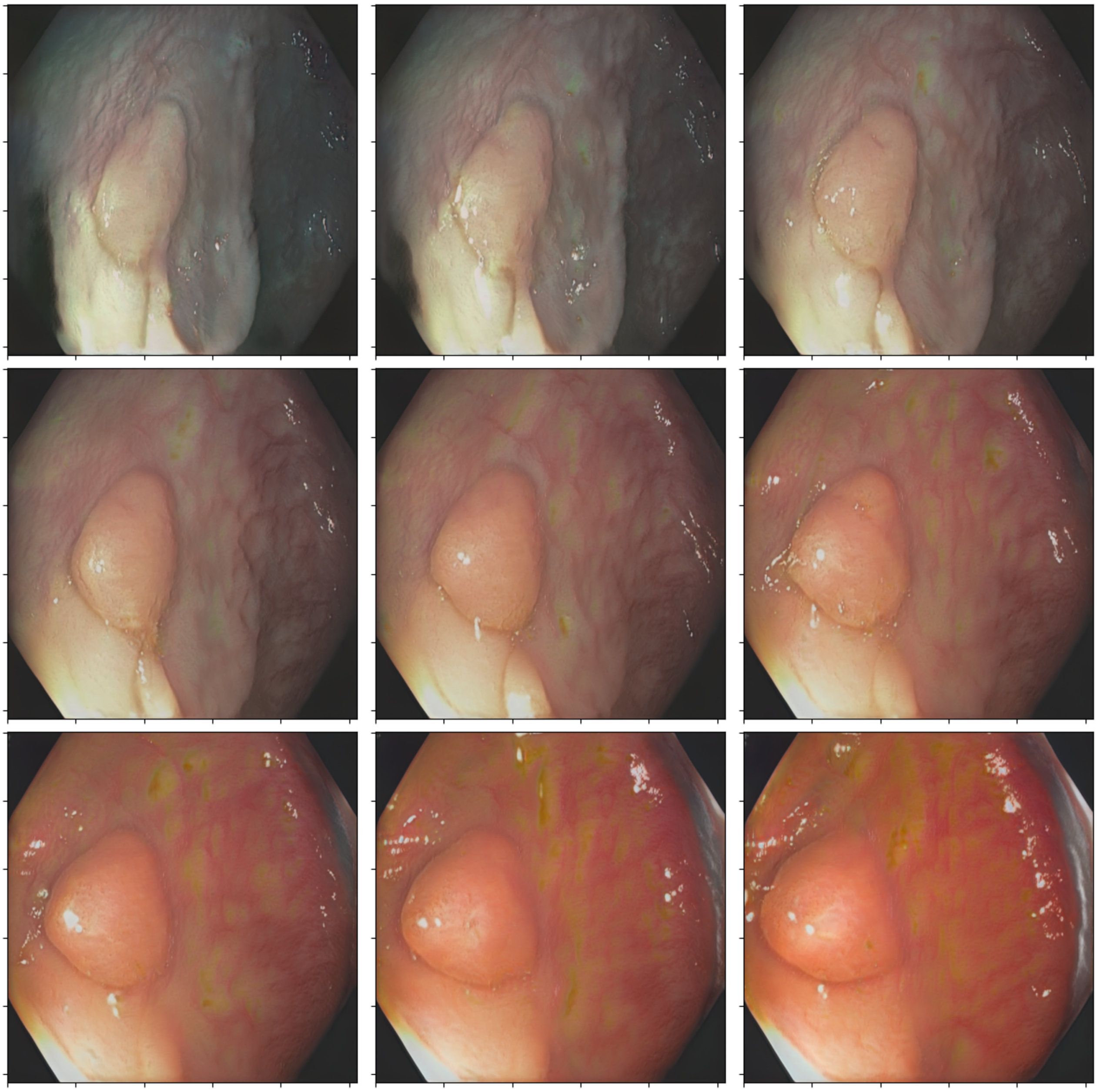


Figure 19: An example of horizontal boundary change using conditional network. Middle image is the original one. Going left represent the change in one direction while going right in the other.

	Horizontal position	Vertical position	Size	Quality
Horizontal position	0°	83.2°	91.3°	87.7°
Vertical position		0°	79.8	86.62°
Size			0°	97.2°
Quality				0°

Table 5: The angle between different decision boundaries is calculated with the conditional network. 90% angle implies that vector are independent of each other.

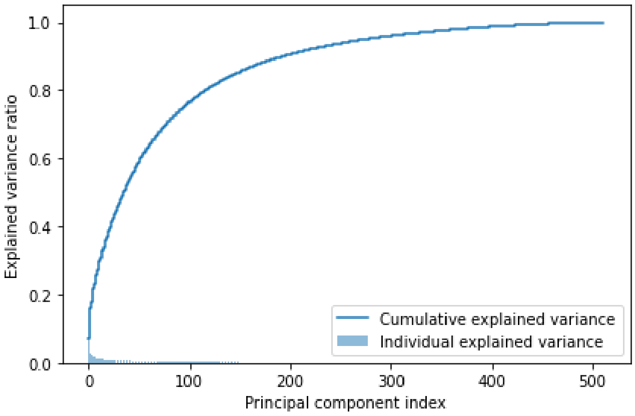


Figure 20: Variance captured by each dimension of the PCA and cumulative variance.

5.2.2 PCA

Amount of component needed/Explained Variance First, the variance explained by each principal component is estimated. The figure 20 shows the variance captured by each dimension as well as the cumulative variance captured by the first N dimensions. The first 100 components explain 76% of the variance, the first 200 - 90.5%, and the first 400 - 98.7%. Visual examples of the representation of a different number of principal components can be observed in Figure 21.

Investigating effect of different principal components on the image and comparison of changing PCA coordinates compared to basic ones The following modifications were made to investigate the effect of different principal components as well as compare them to changes in basic components.

- **Randomize the first 8 PCA, keep the rest**
- **Keep the first 8 PCA, random the rest.**
- **Randomize 8 basic components while keeping the rest.**
- **Keep random 8 basic components, randomize the rest.**

Visual examples of the modifications can be observed in Figure 22.

Investigating PCA components directions The first twenty components were investigated. Each of the components was applied to several ranges of layers to investigate if applying more targeted modification only to certain layers achieves better results. Most of the meaningful modifications found are summarized in a table 7.

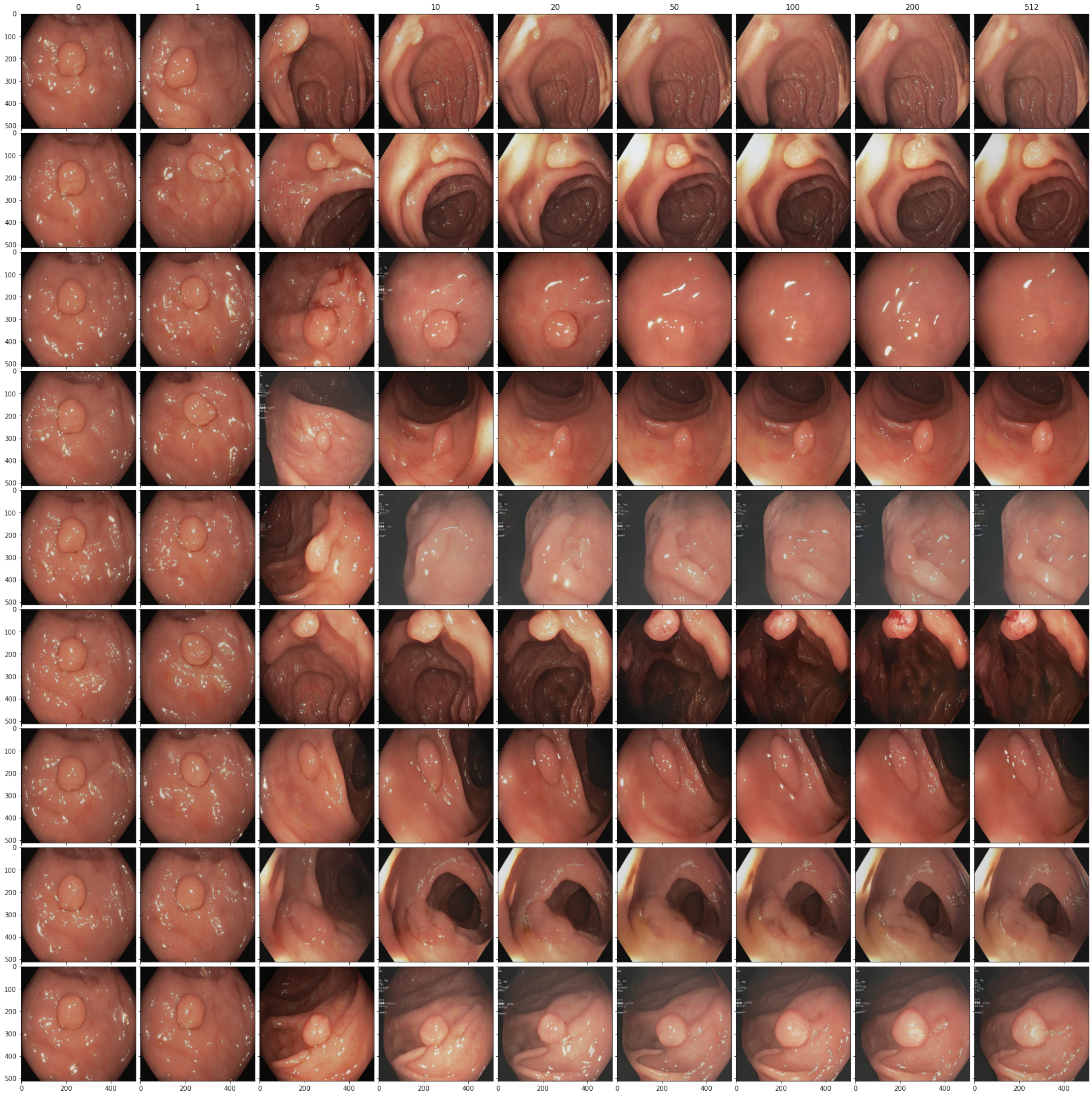
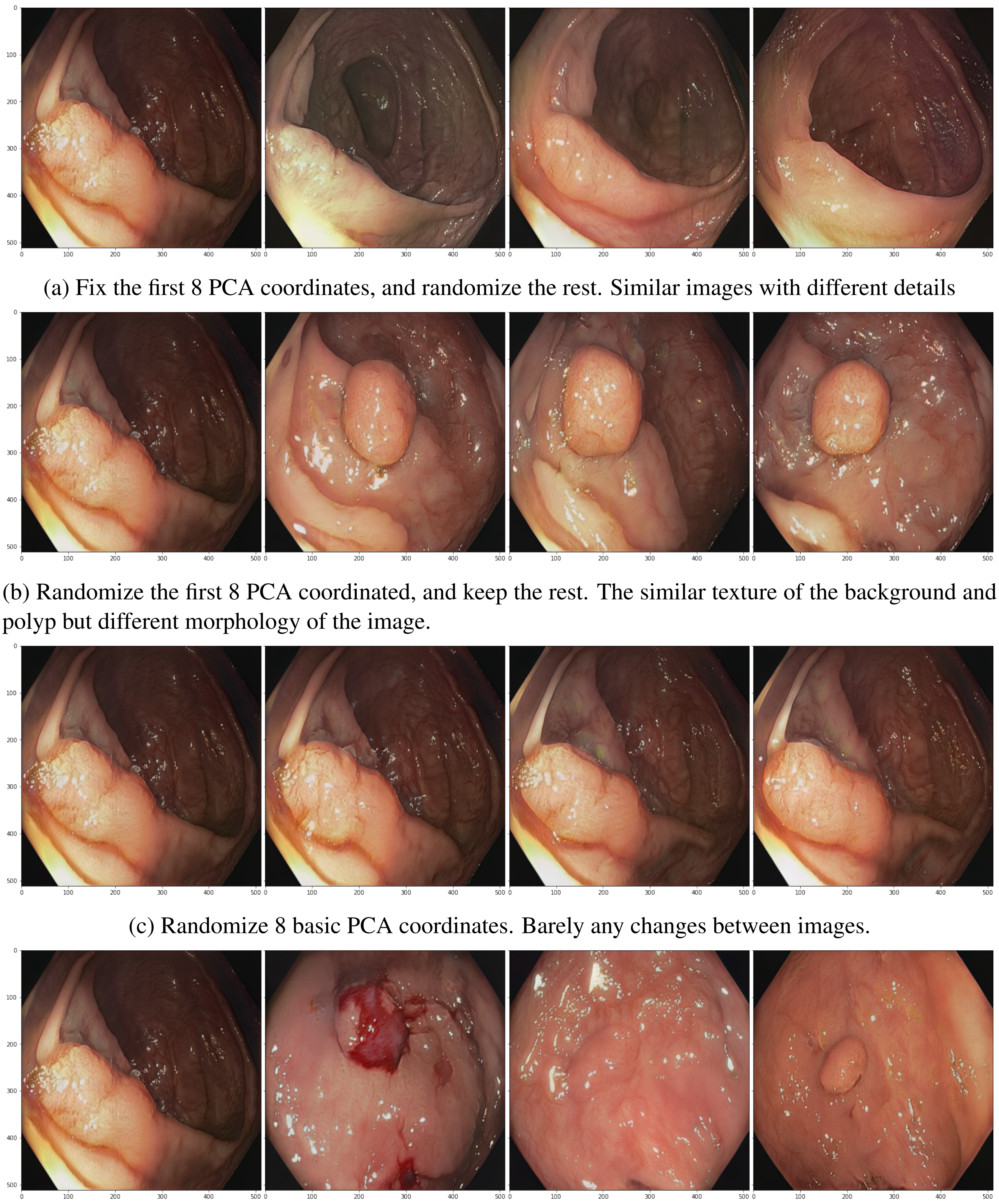


Figure 21: Randomly sampled images, projected onto reduced numbers of PCA dimensions: 0, 1, 5, 10, 20,50,100, 512 (full dimensional).



(a) Fix the first 8 PCA coordinates, and randomize the rest. Similar images with different details

(b) Randomize the first 8 PCA coordinated, and keep the rest. The similar texture of the background and polyp but different morphology of the image.

(c) Randomize 8 basic PCA coordinates. Barely any changes between images.

(d) Keep random 8 basic coordinates, and randomize the rest. Completely random images.

Figure 22: Illustration of the importance of principal components compared to random basic components and illustration of the effect of larger principal components compared to the smaller ones. Subplots (a) and (b) with PCA coordinates change show changes in texture and morphology based on either 8 first kept components or 8 first randomized ones. In comparison changing random 8 basic coordinates didn't change anything while keeping 8 random basic coordinates while changing the rest leads to random images(c,d)

Distance change	C-Hor	UNC-Hor	C-Ver	UNC-Ver	C-Size	UNC-Size	C-Qual	UNC-Qual
-16	-24	-138	-40	-105	-1050	-1500	-0.2	-0.26
-14	-20	-134	-34	-98	-2800	-1300	-0.17	-0.25
-12	-17	-138	-27	-91	-3200	-1200	-0.16	-0.25
-10	-15	-135	-20	-82	-2400	-1000	-0.14	-0.23
-8	-13	-128	-17	-72	-1600	-556	-0.12	-0.22
-6	-10	-116	-11	-60	-1400	-250	-0.08	-0.22
-4	-6	-98	-6	-38	-1000	-500	-0.05	-0.22
-2	-3.5	-56	-5	-22	-700	-100	-0.04	-0.14
0	0	0	0	0	0	0	0	0
2	4	24	0	16	1200	700	0.05	0.18
4	5	55	2	38	2100	600	0.10	0.23
6	8.5	82	2	60	3100	1300	0.17	0.51
8	12	102	1	76	4200	2300	0.24	0.55
10	14	120	0	82	4800	3650	0.30	0.56
12	10	133	-2	95	5200	4300	0.35	0.55
14	10	140	-4	95	6000	6300	0.39	0.55
16	7.76	144	-7	83	7300	9300	0.42	0.54

Table 6: The summary of the evaluation of the boundaries. Prefix C- indicates conditional version of the network, prefix UNC- indicates unconditional version. Hor represents horizontal boundary, Ver - vertical boundary, Size- size boundary, Qual - quality boundary. Horizontal and vertical boundaries evaluated as the average change in pixels of the center of polyp after the modification. Size os evaluated as a change in width*height. Quality is evaluated as an average change in the confidence of a YOLOv5 network.

5.3 Detector training

The mean average precision of the model trained on the original data and the mean average precision of the model trained on the combination of real and synthetic data is reported in Fig 23. After 100 epochs mAP for the original dataset reached 0.827 while for the augmented dataset it reached 0.834.

An example of the detector outcome can be observed in Fig 25 while Fig 24 show original labels annotated by a medical practitioner.

Layers affected	PCA component	-Modification	+Modification	Extra notion	Ref
0-1	2	Abridgment in vertical direction	Elongating in vertical directions		26
0-2	0	Polyp to the right	Polyp to the left		27
0-2	1	Polyp down	Polyp up		28
0-2	2	Abridgment in vertical direction	Elongating in vertical directions		29
0-3	0	Polyp to the right	Polyp to the left		30
0-16	0	Polyp to the right	Polyp to the left	Similar to 0-2,0-3 layers but goes outside of the well-laid territory and produces garbage images during larger modifications	31
0-16	1	polyp down	polyp up	images degrade with large modification	32
0-16	5	’Wall’ view	’Tunnel’ view		33
0-16	6	Smaller polyp	Larger polyp		34
0-16	7	Smaller polyp	Larger polyp		35
3-6	0	Polyp right	Polyp left	More morphology change compared to layers 0-2	36
3-6	1	Polyp down	Polyp up	Similarly more morphology change	37
3-6	3	Smaller polyp	Larger polyp		38
3-6	5	’Wall’ view	’Tunnel’ view	similar to 0,16 but less pronounced	39

Table 7: Meaningful directions were found during the exploration of PCA directions in the latent space.

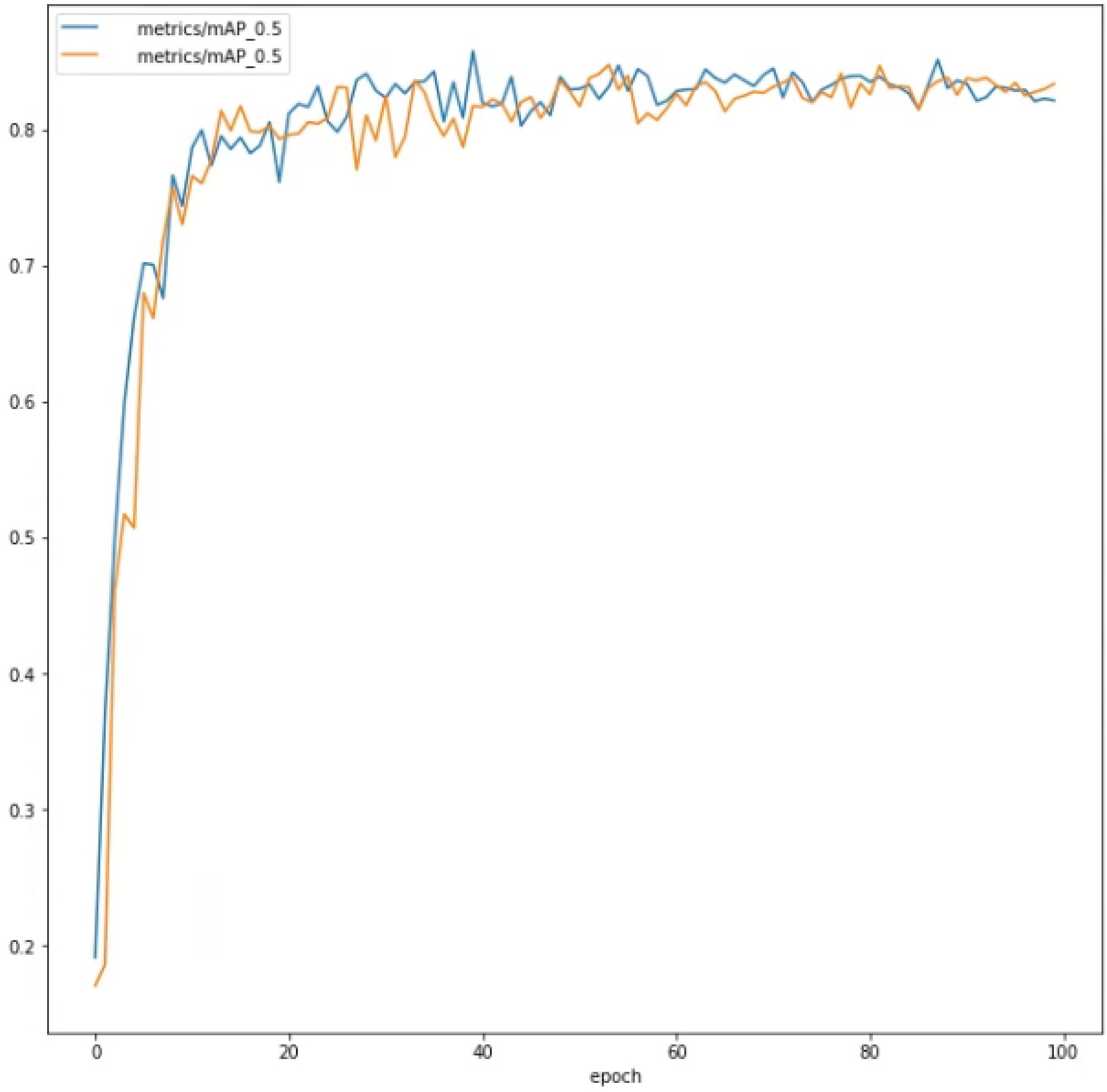


Figure 23: YOLOv5 mAP comparison between no augmentation added to the training set and 408 augmented images selected and annotated by a medical practitioner added to a dataset.

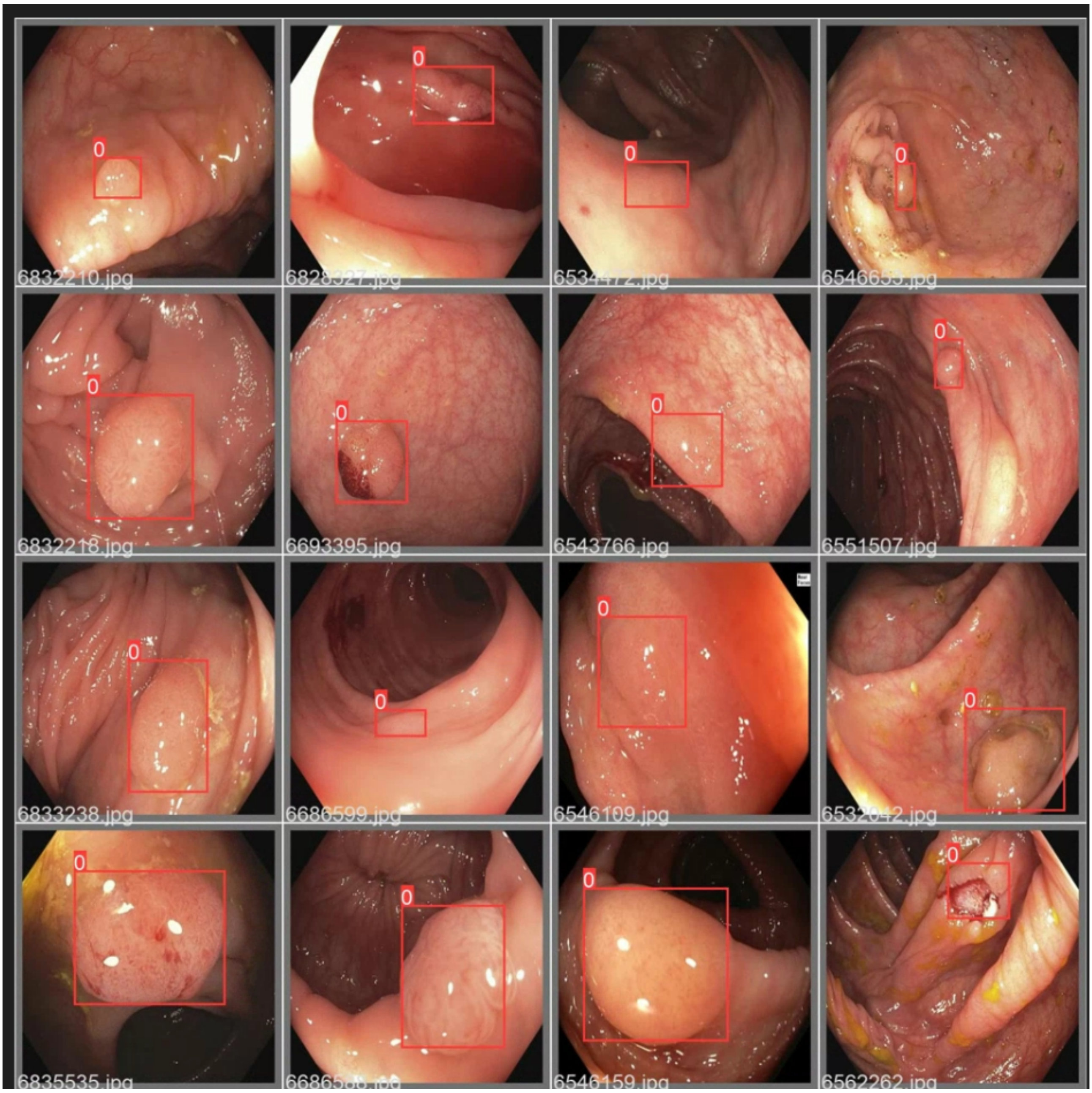


Figure 24: Example of images and bounding boxes used for YOLOv5 training.

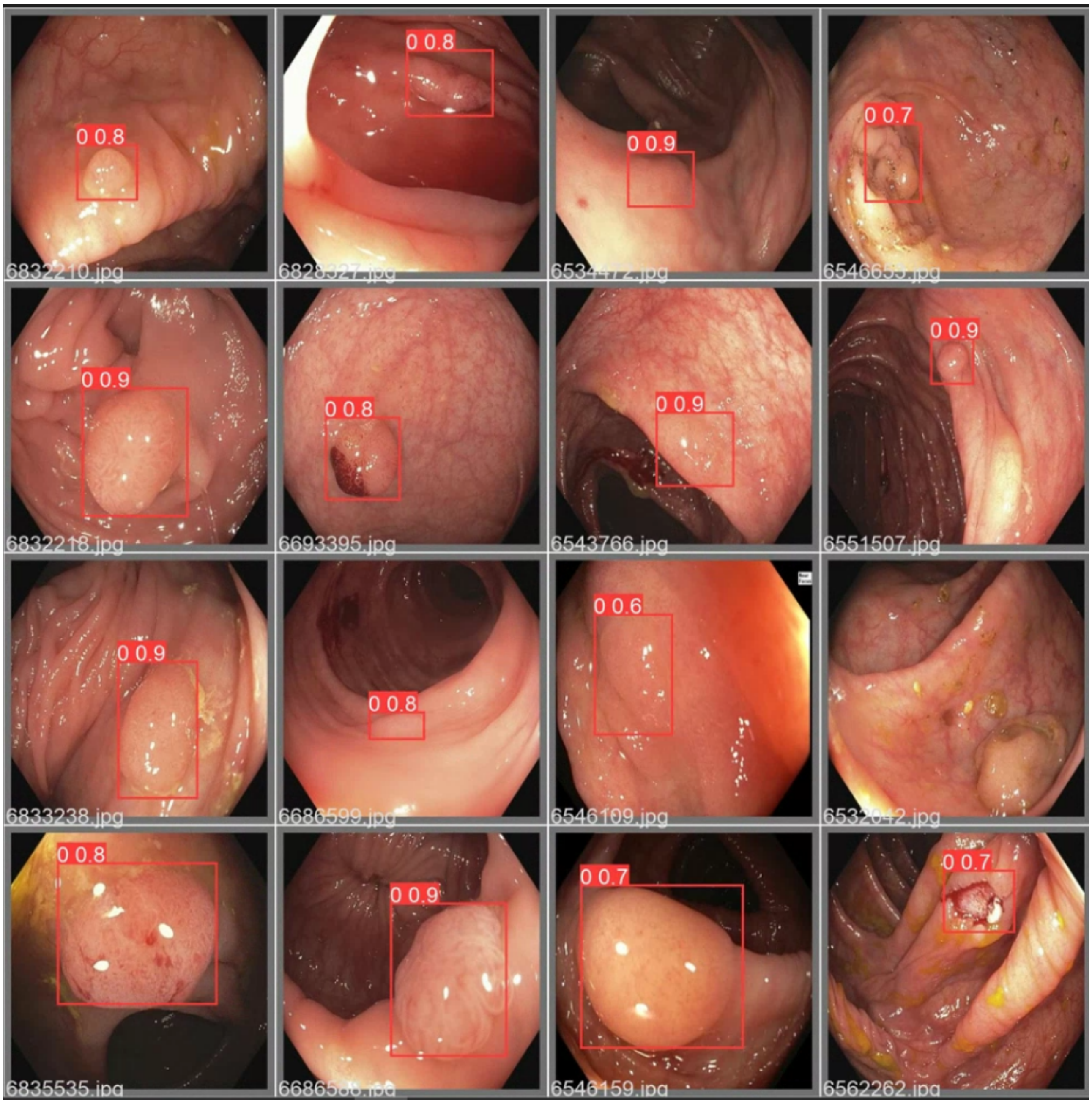


Figure 25: Example of YOLOv5 detector results for comparison with the original labels.

6 Discussion

6.1 Network evaluation

As can be seen from the Table 6 some of the metrics are not suitable for the task of evaluating a generative adversarial network that was trained on a colonoscopy dataset. Inception score is especially ineffective out of 4 chosen metrics.

The inception score is more or less the same for every evaluated step which gives no useful information for the selection of the network. Arguably, the main problem with the Inception Score, in this case, is the fact that the Inception network pre-trained on ImageNet is used for its calculations. ImageNet pretraining is probably not very viable in this case since colonoscopy images are quite different from any of the ImageNet classes and regardless of the difference in the images get classified as the same class from ImageNet.

FID score is another metric that doesn't seem very representative. During the early stages of training(first 2000 kimg) FID went down with the increase in training but from around 3000kimg it mostly oscillated around the value of 17-18 with occasional dips and rises. Based on the FID network at 15000 kimg can be chosen as the best network but it might be just an accidental improvement. Just like the Inception Score, the FID score uses the Inception v3 model pretrained on ImageNet. FID is computed by using the last pooling layer. Activations of that layer are used to compare real images to generated images but since the model was trained on ImageNet it might not extract some of the features that would be relevant for colonoscopy images.

Judging by the SSIM score network checkpoint at the early stages of training(3000 and 4000 kimg) are better than other networks. The question remains how effective that measurement is. Due to the nature of colonoscopy images background of most of the images would be quite similar to each other leading to a higher SSIM score. A lower score during earlier stages of training may indicate lower quality images that are more diverse but not very usable for the next tasks.

At last, YOLOv5 evaluation encompasses YOLOv5 confidence, split between polyps located in the right/left and top/down halves of the image and the average location of the center of polyps. Based on that score network at 15000kimg performed better than the others.

Based on those metrics network 4000 and network 15000 were evaluated higher than others and network 15000 was chosen as a final network for the experiment due to much higher confidence of YOLOv5 evaluation(47% average confidence for 15000 against 19.5% for 4000).

6.2 Latent space manipulation

6.2.1 Hyperplane separation method

All linear boundaries for conditional network achieve at least 75% accuracy while the accuracy of the most removed samples is at a minimum of 83%. Considering the fact that a lot of images with polyps lying close to the center of the image can easily fall into both groups, which is confirmed by the increase in accuracy for the most removed samples, this experiment confirmed the assumption that binary attributes can be separated using separation plane in the latent space of StyleGAN.

Accuracy for the separation boundary of the unconditional network is lower compared to the conditional. It is especially noticeable when tested on the full or test dataset. The biggest difference is between accuracies for the horizontal location that reaches 21-22% for the accuracy of full and test datasets. That difference can be attributed to two potential reasons. First, there is a possibility that the latent space of a conditional network is ”laid out” better due to the need of fitting the labels given to the generator. Since over the long period generated labels would cover the potential locations of the polyps evenly the latent space would better represent different properties. However unconditional generator has no input about the location of the polyp and there is a possibility that a certain type of image would fool the discriminator more consistently leading to a worse diversity and representation of certain types of images. To confirm that the assumption location of the polyp was evaluated for all of the generated images and for the conditional generator the proportion of the image lying in the left half of the image compared to the right half was 2:1. At the same time, the vertical location was close to 1:1 which might have lead to better accuracy for vertical location separation boundary which was just 5-7% lower than a conditional vertical location boundary. The second reason might be the fact that the conditional network generated better images in terms of the confidence of the YOLO evaluator. The number of images with confidence higher than 0.3 for the conditional network over 100k images was close to 70% while the average confidence of the unconditional network was 40%. That might have led to a worse boundary learned since all of the images with the confidence of less than 30% were filtered out before computing the separation boundaries. The second reason is probably less likely since even with more images filtered there were several thousand images in each category to compute the boundary.

Disentanglement analysis Most of the attributes are barely correlated with each other with size and vertical position boundaries being the most correlated to each other with size and vertical position boundaries being the most correlated. That correlation implies that when we would modify one of the attributes using latent space it is likely that at the same time there would be some change in the other attribute. In this example when we modify the size of the polyp its vertical position might also change. On top of that, even a 90° angle between boundaries does not guarantee that only one binary attribute would change with each of the modifications since that would only be true if our decision boundary would perfectly represent each attribute in the latent space which is not the case.

Semantic manipulation Based on those results it seems that manipulation of a latent space of the unconditional network was more effective than manipulation of a conditional network. All 4 modifications tested using the latent space of an unconditional generator lead to meaningful changes. On the other hand modifications of the horizontal and vertical location of the polyp were less successful for the latent space of a conditional GAN. The most unsuccessful modification was the modification of a positive vertical location. Size and quality modifications worked reasonably well and were somewhat related to each other. It seems that larger and more pronounced polyps lead to an increase in the confidence of the YOLOv5 detector and a decrease in size(or even removal of the polyp) leads to a decrease in the confidence of the detector.

6.2.2 PCA

Variation explained To some extent, most images are very close to the original even with 20 components used out of 512. It seems that the remaining components mainly add more details, and change the texture of the background or the polyp but doesn't affect the general morphology of an image compared to 20 components.

Investigating effect of different principal components on the image and comparison of changing PCA coordinates compared to basic ones

- **Randomize the first 8 PCA, keep the rest** It seems that keeping the first 8 principal components while randomizing the rest mainly changes the texture type of the image while preserving the overall morphology and location of the polyp. Polyp size seems to be slightly changing.

- **Keep the first 8 PCA, random the rest.** Keeping the first 8 components while randomizing the rest seems to preserve the texture the most while changing everything else about the image: such as the morphology of the image, location of the polyp(not fully clear)
- **Randomize 8 basic components while keeping the rest.** More or less no changes and those small changes might be caused by the noise of the StyleGAN architecture.
- **Keep random 8 basic components, randomize the rest.** More or less completely different images.

Investigating PCA components directions As can be observed from the Table 7 we've been able to find meaningful directions for changing the location and size of the polyp which leads to the same functionality as the previous method. On top of that, a direction to change the morphology of the image in terms of background view was also found. Changing the 5th component either for all layers or for layers 3-6 leads to a change of background view from a full 'wall' view to a 'tunnel' view. It is possible that some of the modifications were missed during the inspection due to the lack of medical knowledge. Most likely more modifications can be found with an inspection of more components and layers combination and the involvement of a medical practitioner.

7 Conclusion

This chapter will discuss various aspects of the thesis in an attempt to answer the proposed research questions as well as potential improvements and future works for each topic.

7.1 Research Question 1: Estimating the possibility of the latent space manipulation

Various methods of latent space exploration were tested during this project, with two of those being investigated more thoroughly while others were dropped during the exploration stage. Both of the two main methods were proven to work, at least for simple geometric manipulation, such as changing the polyp's location or the polyp's size. In addition, the PCA-based method showed an ability to change the image morphology using some of the tested modifications. However, it is possible that many targeted modifications weren't discovered using the PCA method due to the lack of medical knowledge and time for the investigation.

Latent space manipulation using SVM to find separation boundaries also produced positive results. However, only geometric manipulation of the location of the polyp was achieved using this method.

7.2 Research Question 2: Investigation of various methods of latent space manipulation

Two main latent space manipulation methods were investigated: the SVM-based method for finding separation boundary for a binary attribute in the latent space and the PCA-based method for finding directions for manipulation in the latent space. Both methods have some advantages and disadvantages, and it is hard to tell that one method is strictly better than another.

The SVM-based method has the following disadvantages. First, it works only for binary attributes, which limits its effectiveness for such manipulations as changing polyp types. Second, it requires a classifier for each attribute, which is often not possible, especially in a situation where there is already a lack of data in a field. Third, it is quite computationally expensive since, ideally, a lot of images need to be generated, each of those images needs to be assigned a binary attribute, and then SVM needs to be trained.

On the other hand, if all of the prerequisites are available, such as a classifier for a binary attribute, researchers are interested in a binary attribute, and a powerful computational cluster is available. Furthermore, the method is quite simple to implement and can provide acceptable results.

On the other hand, the PCA-based method is far less computationally expensive, does not require external classifiers, and can potentially change more than a binary attribute. It has quite a few advantages over SVM based method, but there are also some downsides. Mainly there is a need for human evaluation of each of the PCA directions found. Especially since it was found that manipulating only some layers of the ω latent space provides more targeted results compared to modifying all of the layers. Evaluation of those directions will require a lot of menial work and, in a domain such as the medical domain, preferable involvement of a medical practitioner. Overall, the PCA-based method might have more potential due to the ability to discover target modifications that are almost impossible to find using the SVM-based method.

7.3 Research Question 3: Detector performance with the augmented dataset

The third objective of the study was to investigate the effect of synthetic images on the performance of automated polyp detection systems.

Adding synthetic images to the real data during the training showed close to no effect compared to just training on the original data. Even though no improvement in performance was observed, there also was no decrease in performance. It might indicate that synthesized images achieved similar quality compared to the real images. If synthesized images were of a lower quality, it would negatively affect the performance of the detection system. Another reason for the lack of improvement might be the type of images generated by the network. Since the generator would try to represent the real life distribution of data, a big part of the generated images would represent relatively easy to spot polyp. Even though those images would be of high quality, they wouldn't improve a detector's performance since it would easily identify this type of polyps in the first place.

One of the limitations of the current structure is the need for human involvement in the inspection of generated images. Since generative network generates images of different quality, there is a need for human inspection of those images. There are at least two potential solutions to that problem: improving the quality of generated images or having some automated way of quality control for generated images that would filter low-quality images by itself.

One of the possible additions for improving the detector is an improvement in latent space exploration methods that unfortunately weren't achieved during this project. In the ideal world, using latent space manipulation, we would like to have the ability to generate a certain type of polyps, preferably a type of polyps that are rare, hard to spot, and underrepresented in real life data. If the generation of such polyps had been possible, it would likely improve the performance of a detection system.

7.4 Future research

There are various paths for future research based on the outcomes of this project.

- This project focused on exploring latent space in already trained networks. It is possible that modifying the training sequence of StyleGAN can result in a 'better' latent space which might lead to more meaningful findings.

- Investigating different methods of image generation such as diffusion models which can lead to a better quality of image generation [63].

- Finding other methods of latent space exploration that may yield better results

- Conditional GAN can be an effective way of controlling images generated by a network, by conditional GAN requires more labeled data for training.

7.5 Conclusion

This project focused on two main topics: manipulating latent space and improving detector performance using augmented images. Different techniques for latent space manipulation were investigated, and those methods achieved some success. Mainly geometric manipulations of the polyp's location or size are possible using both SVM-based and PCA-based methods. Using PCA based method, some modifications were found that also affect the morphology of the image itself. Unfortunately, we couldn't achieve the change in the type of polyps which was one of the things desired at the beginning of the project. In the end, we can achieve simple manipulation of a polyp location but unfortunately, not much on top of that.

Detector performance neither increased nor decreased after using the augmented dataset. Unfortunately, latent space exploration didn't reach the level where it could be used to increase the performance of a detector. By default, for images produced by GAN, the performance of the detector stayed at the same level.

Bibliography

[1] K. Thanikachalam and G. Khan, “Colorectal cancer and nutrition,” *Nutrients*, vol. 11, no. 1, 2019.

[2] F. Bray, J. Ferlay, I. Soerjomataram, R. L. Siegel, L. A. Torre, and A. Jemal, “Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries,” *CA Cancer J Clin*, vol. 68, pp. 394–424, Sept. 2018.

[3] V. Balchen and K. Simon, “Colorectal cancer development and advances in screening,” *Clinical Interventions in Aging*, vol. Volume 11, pp. 967–976, 07 2016.

[4] S. M. e. Silva, V. F. Rosa, A. C. N. d. Santos, R. M. d. Almeida, P. G. d. Oliveira, and J. B. d. Sousa, “Influence of patient age and colorectal polyp size on histopathology findings,” *Arq Bras Cir Dig*, vol. 27, pp. 109–113, Apr. 2014.

[5] E. C. Kim and P. Lance, “Colorectal polyps and their relationship to cancer,” *Gastroenterol Clin North Am*, vol. 26, pp. 1–17, Mar. 1997.

[6] F. Stracci, M. Zorzi, and G. Grazzini, “Colorectal cancer screening: tests, strategies, and perspectives,” *Front Public Health*, vol. 2, p. 210, Oct. 2014.

[7] M. Y. Chan, H. Cohen, and B. M. R. Spiegel, “Fewer polyps detected by colonoscopy as the day progresses at a veteran’s administration teaching hospital,” *Clin Gastroenterol Hepatol*, vol. 7, pp. 1217–23; quiz 1143, July 2009.

[8] R. M. Soetikno, T. Kaltenbach, R. V. Rouse, W. Park, A. Maheshwari, T. Sato, S. Matsui, and S. Friedland, “Prevalence of Nonpolypoid (Flat and Depressed) Colorectal Neoplasms in Asymptomatic and Symptomatic Adults,” *JAMA*, vol. 299, pp. 1027–1035, 03 2008.

[9] Y. Mori, S.-E. Kudo, T. M. Berzin, M. Misawa, and K. Takeda, “Computer-aided diagnosis for colonoscopy,” *Endoscopy*, vol. 49, pp. 813–819, May 2017.

[10] C. Hassan, M. Spadaccini, A. Iannone, R. Maselli, M. Jovani, V. T. Chandrasekar, G. Antonelli, H. Yu, M. Areia, M. Dinis-Ribeiro, P. Bhandari, P. Sharma, D. K. Rex, T. R¶sch, M. Wallace, and A. Repici, “Performance of artificial intelligence in colonoscopy for adenoma and polyp detection: a systematic review and meta-analysis,” *Gastrointestinal Endoscopy*, vol. 93, no. 1, pp. 77–85.e6, 2021.

[11] P. Chlap, H. Min, N. Vandenberg, J. Dowling, L. Holloway, and A. Haworth, “A review of medical image data augmentation techniques for deep learning applications,” *Journal of Medical Imaging and Radiation Oncology*, vol. 65, no. 5, pp. 545–563, 2021.

[12] J. Islam and Y. Zhang, “GAN-based synthetic brain PET image generation,” *Brain Informatics*, vol. 7, p. 3, Mar. 2020.

[13] M. L. Olender, J. M. de la Torre Hernández, L. S. Athanasiou, F. R. Nezami, and E. R. Edelman, “Artificial intelligence to generate medical images: augmenting the cardiologist’s visual clinical workflow,” *European Heart Journal - Digital Health*, vol. 2, pp. 539–544, 06 2021.

[14] C. Han, H. Hayashi, L. Rundo, R. Araki, W. Shimoda, S. Muramatsu, Y. Furukawa, G. Mauri, and H. Nakayama, "Gan-based synthetic brain mr image generation," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 734–738, 2018.

[15] D. F. Bauer, T. Russ, B. I. Waldkirch, C. Tönnes, W. P. Segars, L. R. Schad, F. G. Zöllner, and A.-K. Golla, "Generation of annotated multimodal ground truth datasets for abdominal medical image registration," *International Journal of Computer Assisted Radiology and Surgery*, vol. 16, pp. 1277–1285, Aug. 2021.

[16] M. Popescu, "Generating synthetic training data using deep generative adversarial networks in medical endoscopy images," Master's thesis, 2020.

[17] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of gans for semantic face editing," *CoRR*, vol. abs/1907.10786, 2019.

[18] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris, "Ganspace: Discovering interpretable GAN controls," *CoRR*, vol. abs/2004.02546, 2020.

[19] Y. Shen, C. Yang, X. Tang, and B. Zhou, "Interfacegan: Interpreting the disentangled face representation learned by gans," *CoRR*, vol. abs/2005.09635, 2020.

[20] N. C. Abay, Y. Zhou, M. Kantarcioglu, B. Thuraisingham, and L. Sweeney, "Privacy preserving synthetic data release using deep learning," in *Machine Learning and Knowledge Discovery in Databases* (M. Berlingerio, F. Bonchi, T. Gärtner, N. Hurley, and G. Ifrim, eds.), (Cham), pp. 510–526, Springer International Publishing, 2019.

[21] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-propagating Errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[22] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[23] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, pp. 98–136, Jan. 2015.

[24] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, 2013.

[25] R. B. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, 2015.

[26] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[27] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017.

[28] A. Pramanik, S. K. Pal, J. Maiti, and P. Mitra, "Granulated rcnn and multi-class deep sort for multi-object detection and tracking," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 1, pp. 171–181, 2022.

[29] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *CoRR*, vol. abs/1506.02640, 2015.

[30] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” *CoRR*, vol. abs/1708.02002, 2017.

[31] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, “SSD: single shot multibox detector,” *CoRR*, vol. abs/1512.02325, 2015.

[32] A. Bochkovskiy, C. Wang, and H. M. Liao, “Yolov4: Optimal speed and accuracy of object detection,” *CoRR*, vol. abs/2004.10934, 2020.

[33] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, TaoXie, J. Fang, imyhxy, K. Michael, Lorna, A. V, D. Montes, J. Nadar, Laughing, tkianai, yxNONG, P. Skalski, Z. Wang, A. Hogan, C. Fati, L. Mammana, AlexWang1900, D. Patel, D. Yiwei, F. You, J. Hajek, L. Diaconu, and M. T. Minh, “ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference,” Feb. 2022.

[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” *CoRR*, vol. abs/1409.4842, 2014.

[35] J. Redmon and A. Farhadi, “YOLO9000: better, faster, stronger,” *CoRR*, vol. abs/1612.08242, 2016.

[36] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *CoRR*, vol. abs/1804.02767, 2018.

[37] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[38] U. Nepal and H. Eslamiat, “Comparing yolov3, yolov4 and yolov5 for autonomous landing spot detection in faulty uavs,” *Sensors*, vol. 22, no. 2, 2022.

[39] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. Book in preparation for MIT Press.

[40] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” 2014.

[41] V. Nagarajan and J. Z. Kolter, “Gradient descent gan optimization is locally stable,” 2017.

[42] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *CoRR*, vol. abs/1411.1784, 2014.

[43] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *CoRR*, vol. abs/1611.07004, 2016.

[44] T. Miyato and M. Koyama, “cgans with projection discriminator,” *CoRR*, vol. abs/1802.05637, 2018.

[45] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training gans,” *CoRR*, vol. abs/1606.03498, 2016.

[46] M. Arjovsky and L. Bottou, “Towards principled methods for training generative adversarial networks,” 2017.

[47] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training gans,” *CoRR*, vol. abs/1606.03498, 2016.

[48] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” *CoRR*, vol. abs/1512.00567, 2015.

[49] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” *CoRR*, vol. abs/1812.04948, 2018.

[50] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of stylegan,” *CoRR*, vol. abs/1912.04958, 2019.

[51] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, “Training generative adversarial networks with limited data,” *CoRR*, vol. abs/2006.06676, 2020.

[52] A. Bora, E. Price, and A. G. Dimakis, “AmbientGAN: Generative models from lossy measurements,” in *International Conference on Learning Representations*, 2018.

[53] A. Brock, J. Donahue, and K. Simonyan, “Large scale GAN training for high fidelity natural image synthesis,” *CoRR*, vol. abs/1809.11096, 2018.

[54] A. Jahanian, L. Chai, and P. Isola, “On the ”steerability” of generative adversarial networks,” *CoRR*, vol. abs/1907.07171, 2019.

[55] C. Yang, Y. Shen, and B. Zhou, “Semantic hierarchy emerges in deep generative representations for scene synthesis,” *CoRR*, vol. abs/1911.09267, 2019.

[56] L. Goetschalckx, A. Andonian, A. Oliva, and P. Isola, “Ganalyze: Toward visual definitions of cognitive image properties,” *arXiv preprint arXiv:1906.10112*, 2019.

[57] L. Tran, X. Yin, and X. Liu, “Disentangled representation learning gan for pose-invariant face recognition,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1283–1292, 2017.

[58] D. Bau, H. Strobelt, W. S. Peebles, J. Wulff, B. Zhou, J. Zhu, and A. Torralba, “Semantic photo manipulation with a generative image prior,” *CoRR*, vol. abs/2005.07727, 2020.

[59] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” *CoRR*, vol. abs/1710.10196, 2017.

[60] S. Barratt and R. Sharma, “A note on the inception score,” 2018.

[61] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter, “Gans trained by a two time-scale update rule converge to a nash equilibrium,” *CoRR*, vol. abs/1706.08500, 2017.

[62] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *Trans. Img. Proc.*, vol. 13, p. 600–612, apr 2004.

[63] P. Dhariwal and A. Nichol, “Diffusion models beat gans on image synthesis,” *CoRR*, vol. abs/2105.05233, 2021.

Appendices

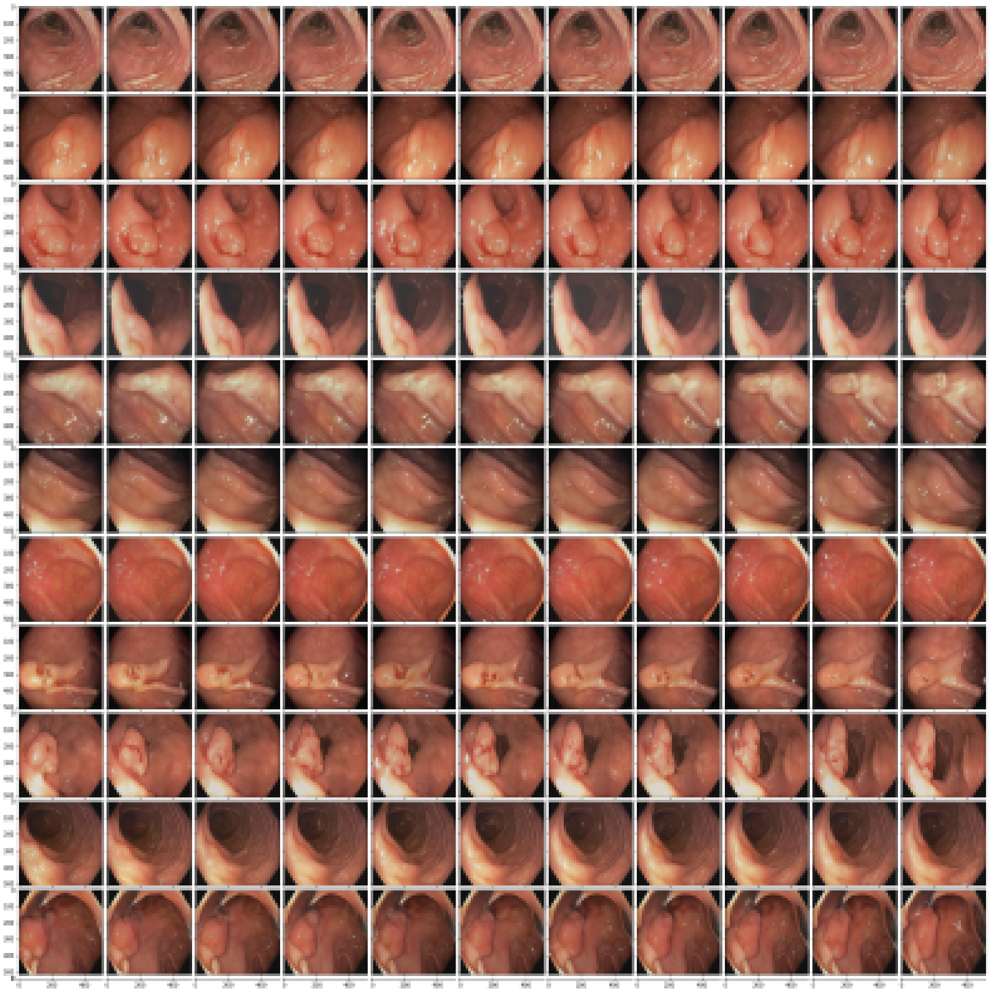


Figure 26: Example of modifications caused by changing principal component 0 in layers 0-1.

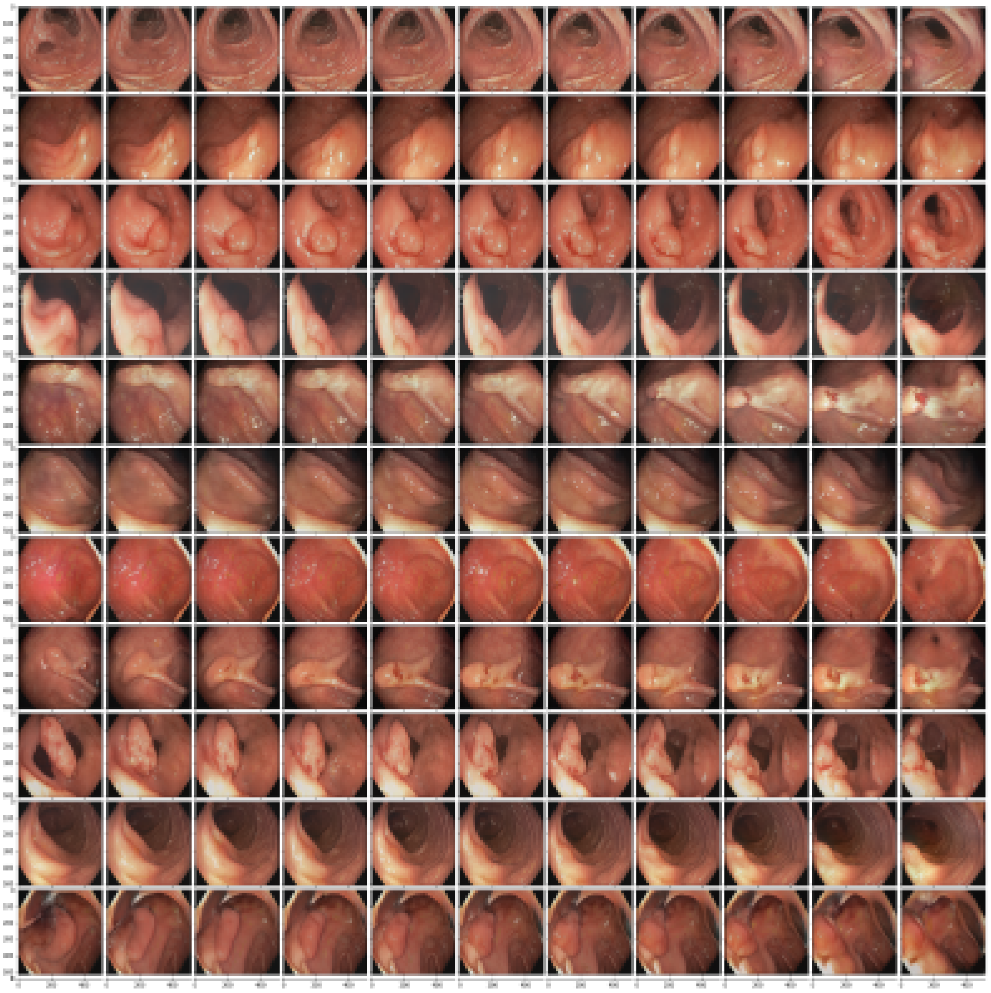


Figure 27: Example of modifications caused by changing principal component 0 in layers 0-2.

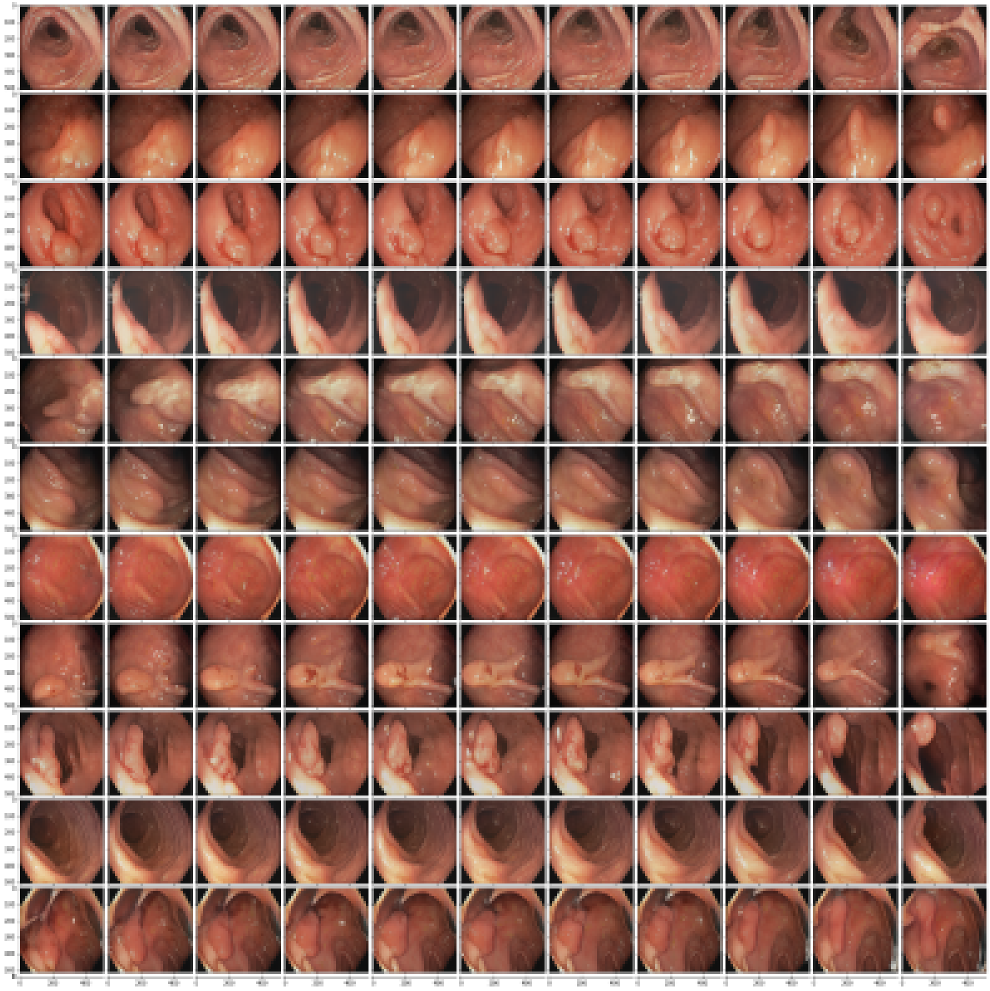


Figure 28: Example of modifications caused by changing principal component 1 in layers 0-2.

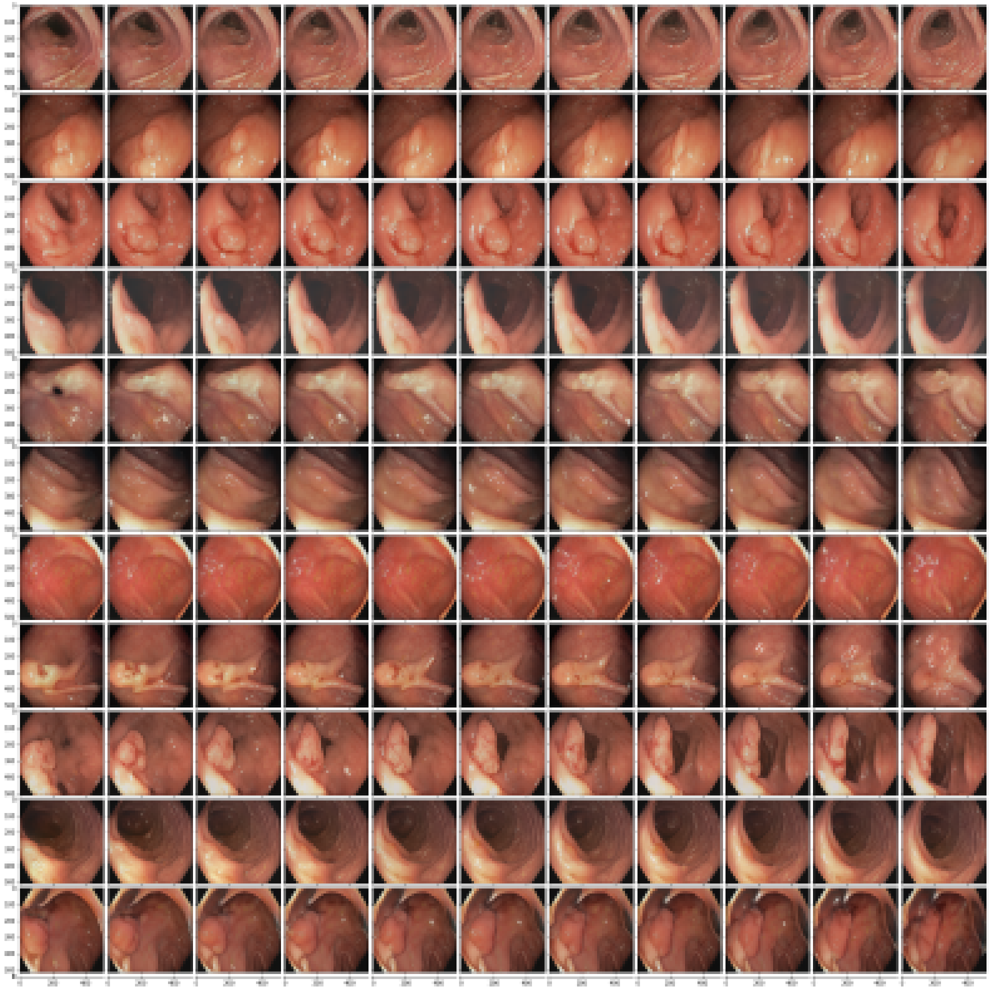


Figure 29: Example of modifications caused by changing principal component 2 in layers 0-2.

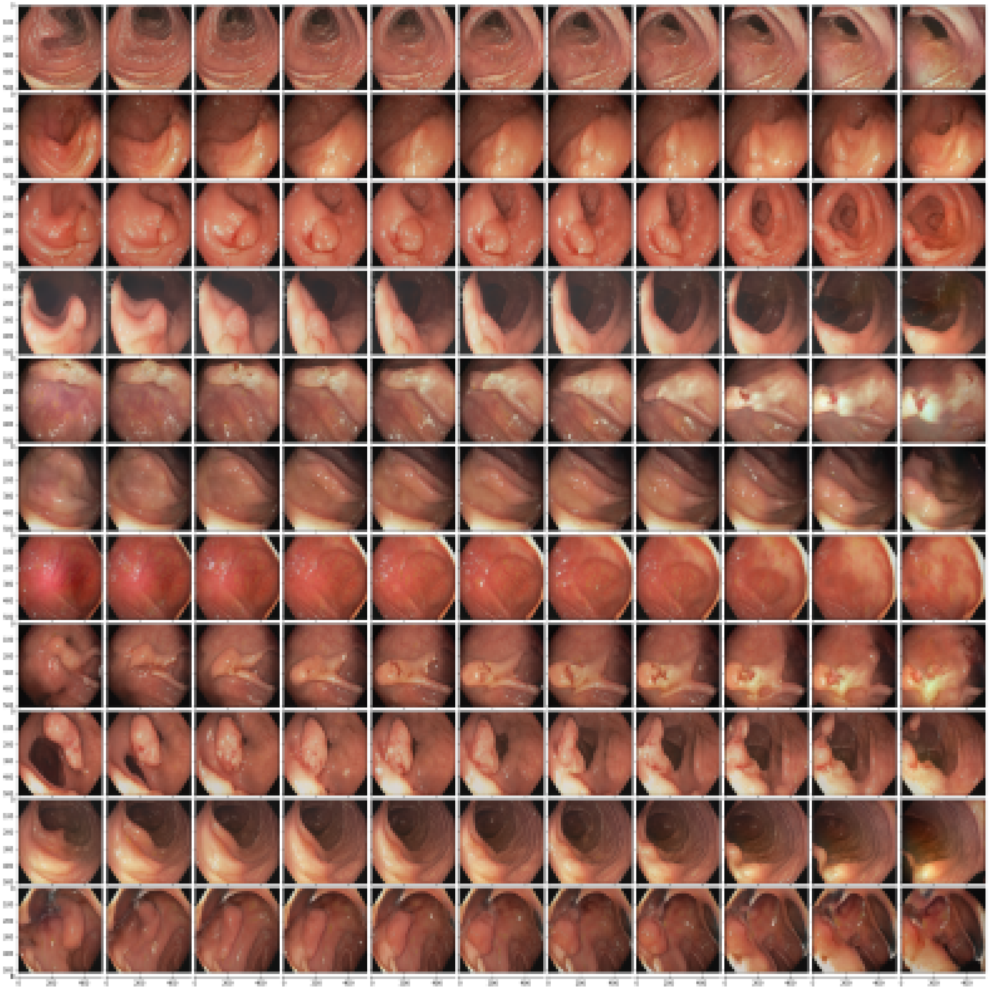


Figure 30: Example of modifications caused by changing principal component 0 in layers 0-3.

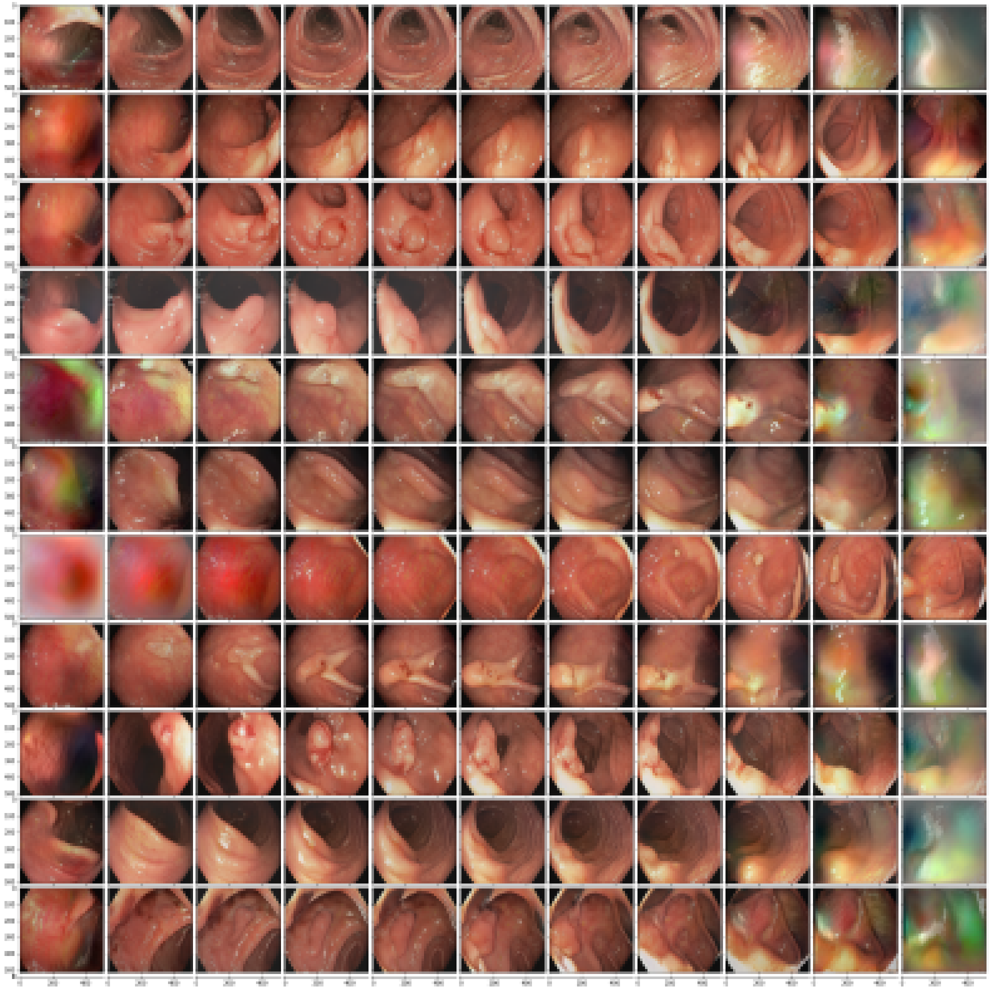


Figure 31: Example of modifications caused by changing principal component 0 in layers 0-16.

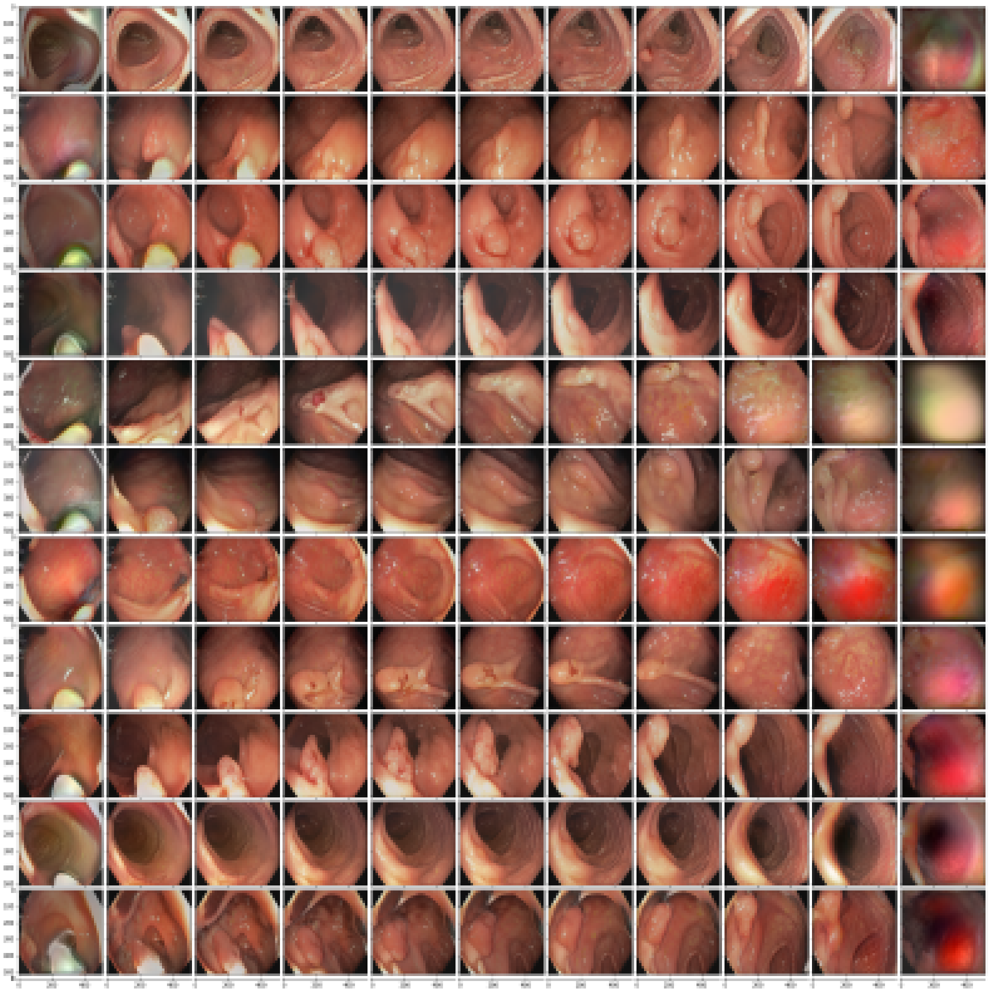


Figure 32: Example of modifications caused by changing principal component 1 in layers 0-16.

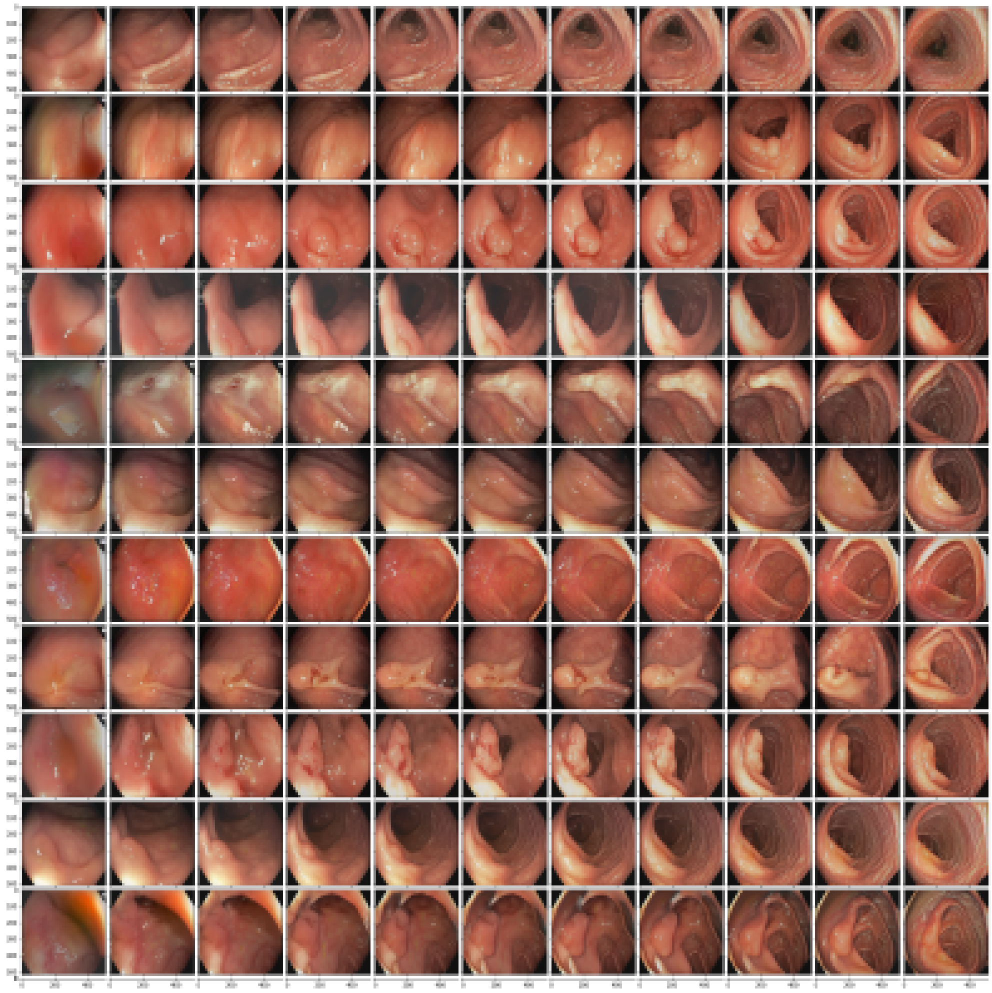


Figure 33: Example of modifications caused by changing principal component 5 in layers 0-16.

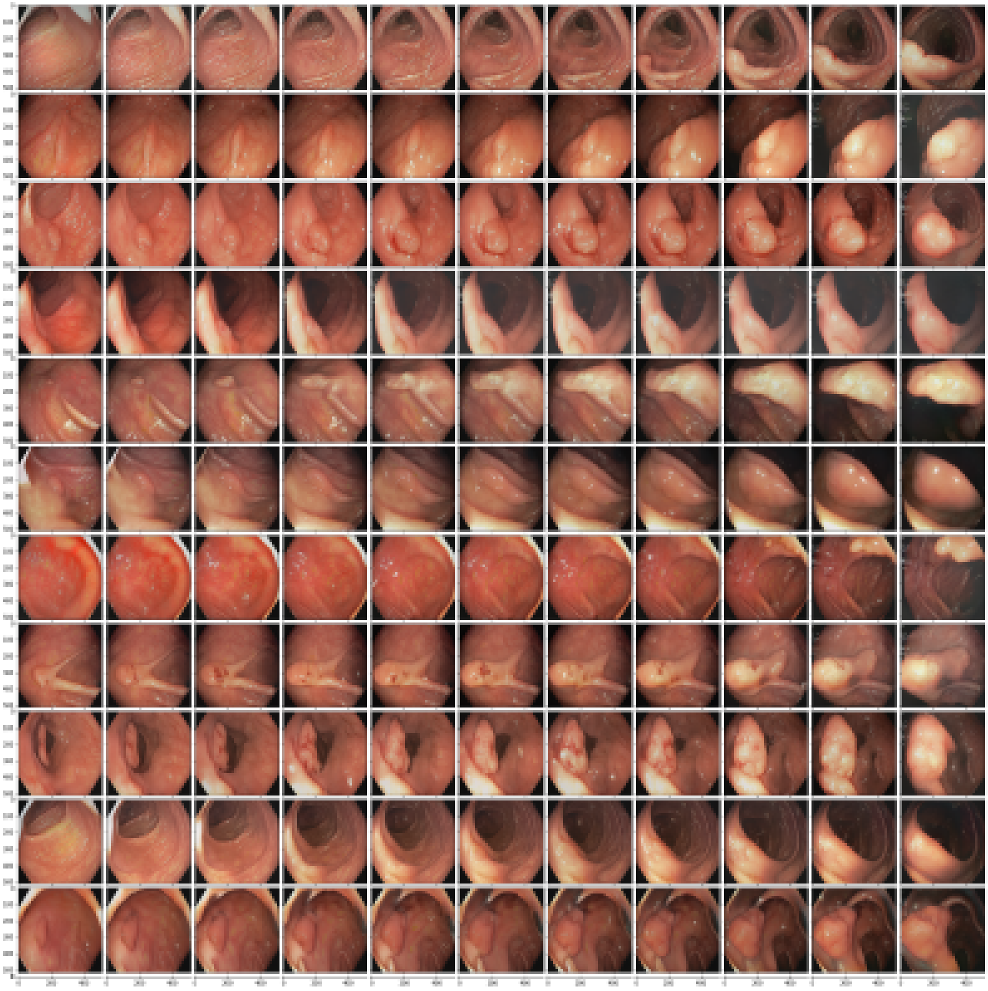


Figure 34: Example of modifications caused by changing principal component 6 in layers 0-16.

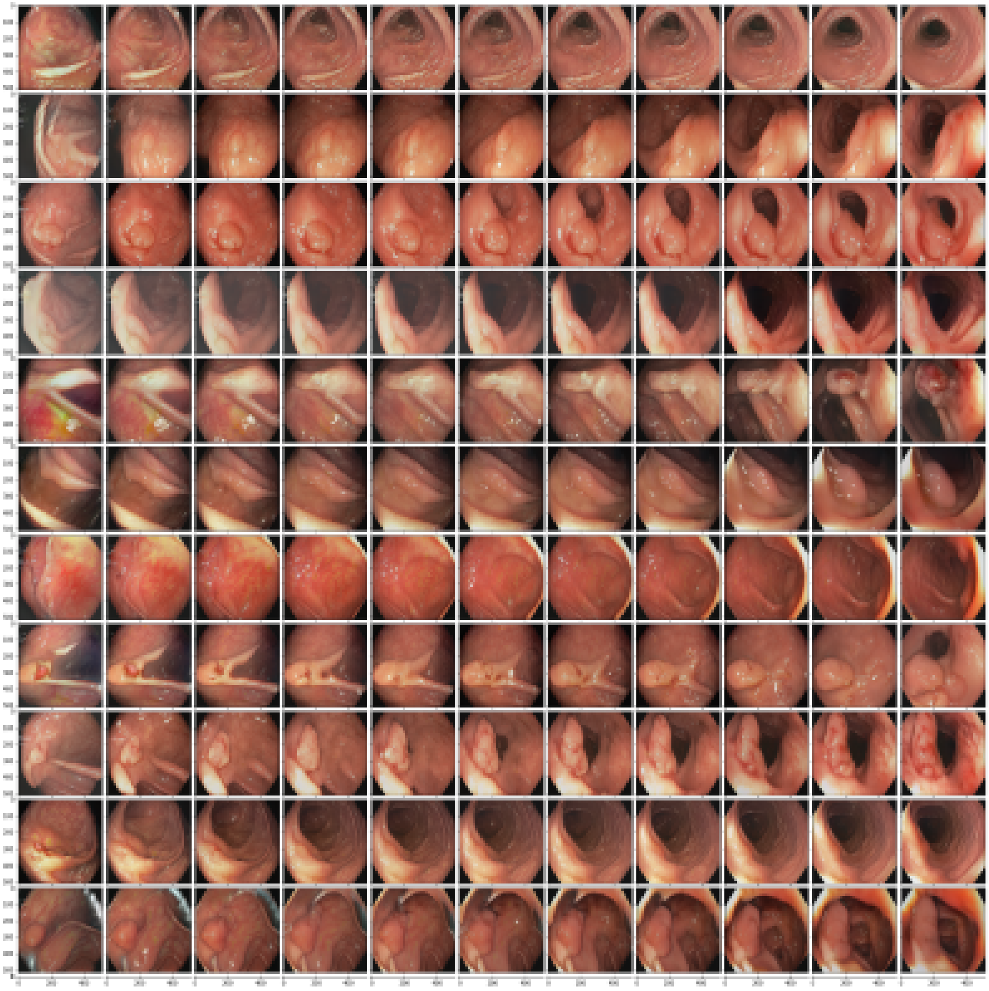


Figure 35: Example of modifications caused by changing principal component 7 in layers 0-16.

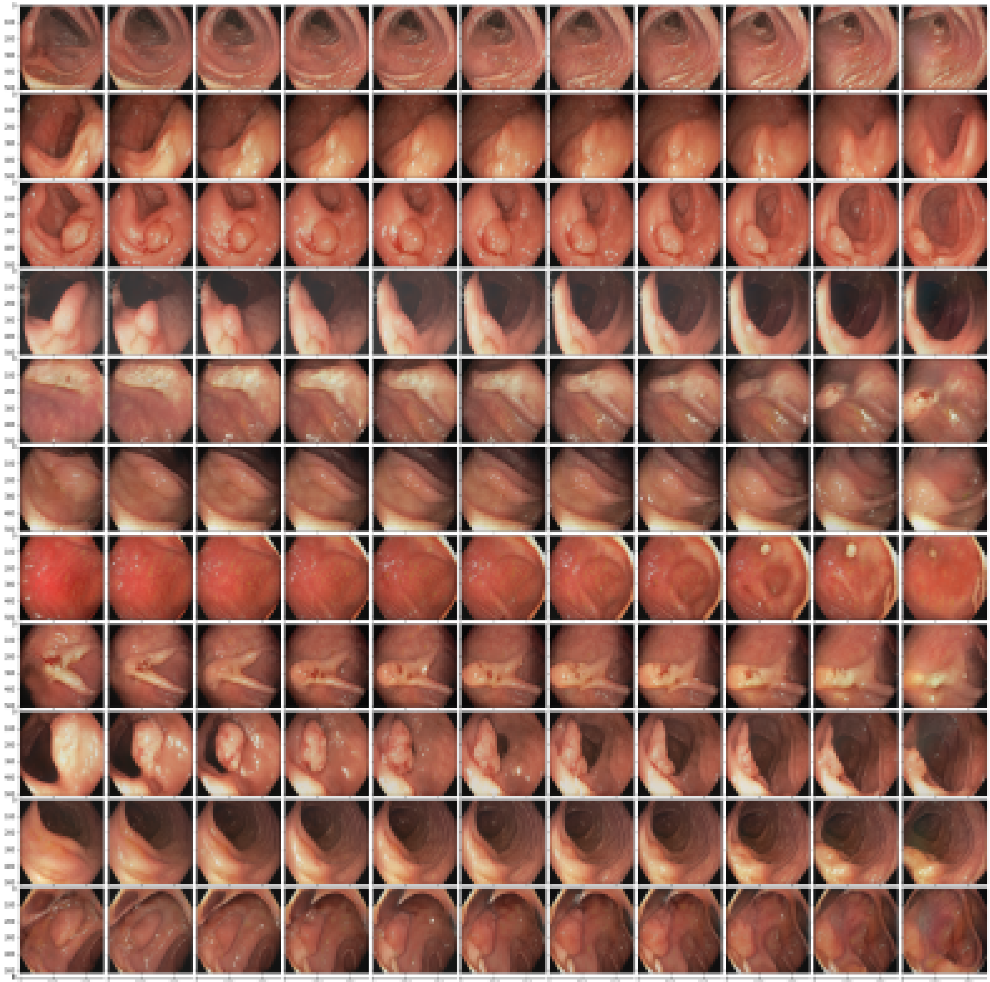


Figure 36: Example of modifications caused by changing principal component 0 in layers 3-6.

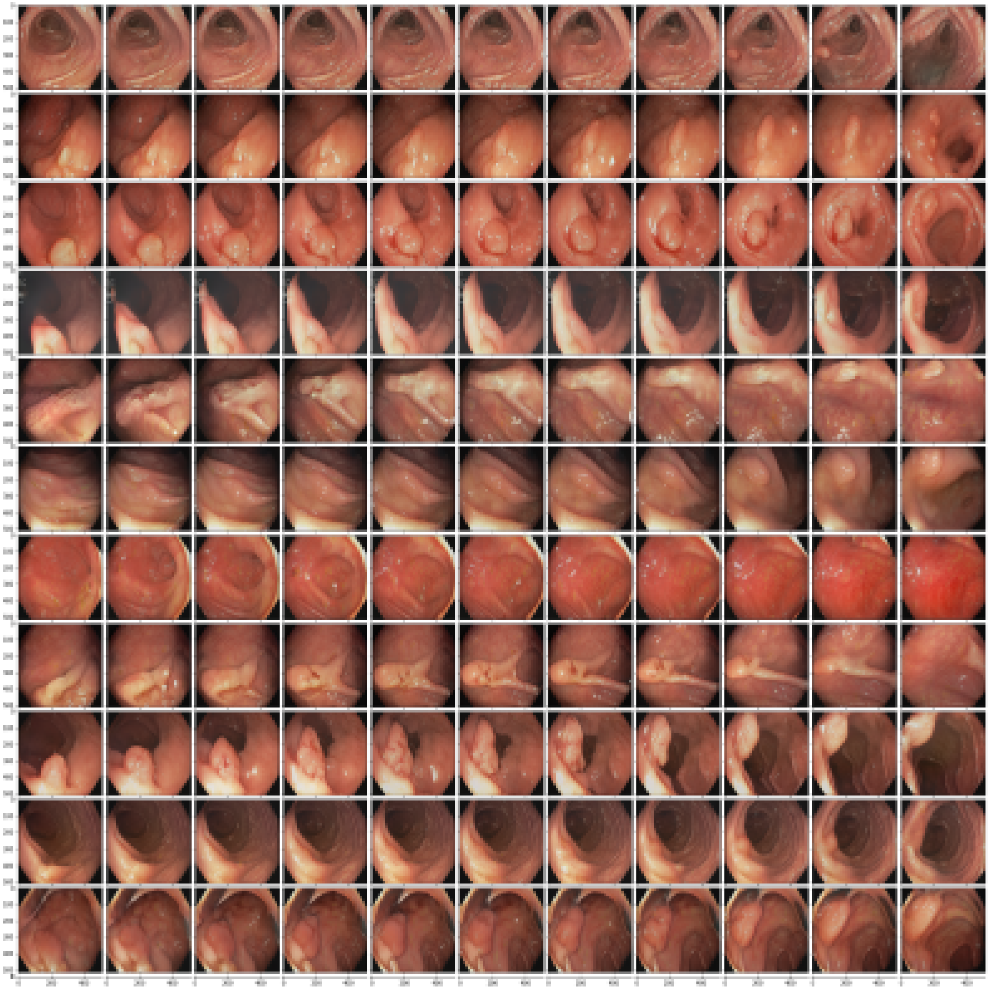


Figure 37: Example of modifications caused by changing principal component 1 in layers 3-6.

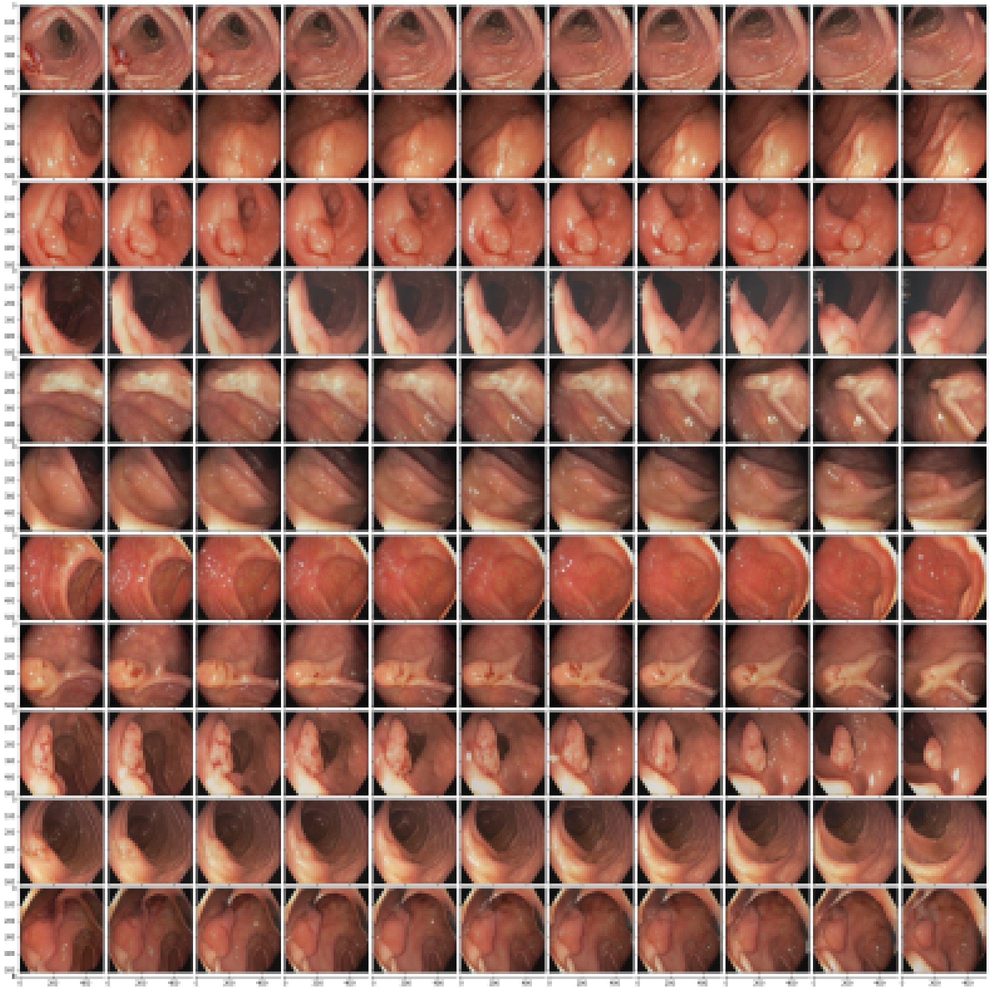


Figure 38: Example of modifications caused by changing principal component 3 in layers 3-6.

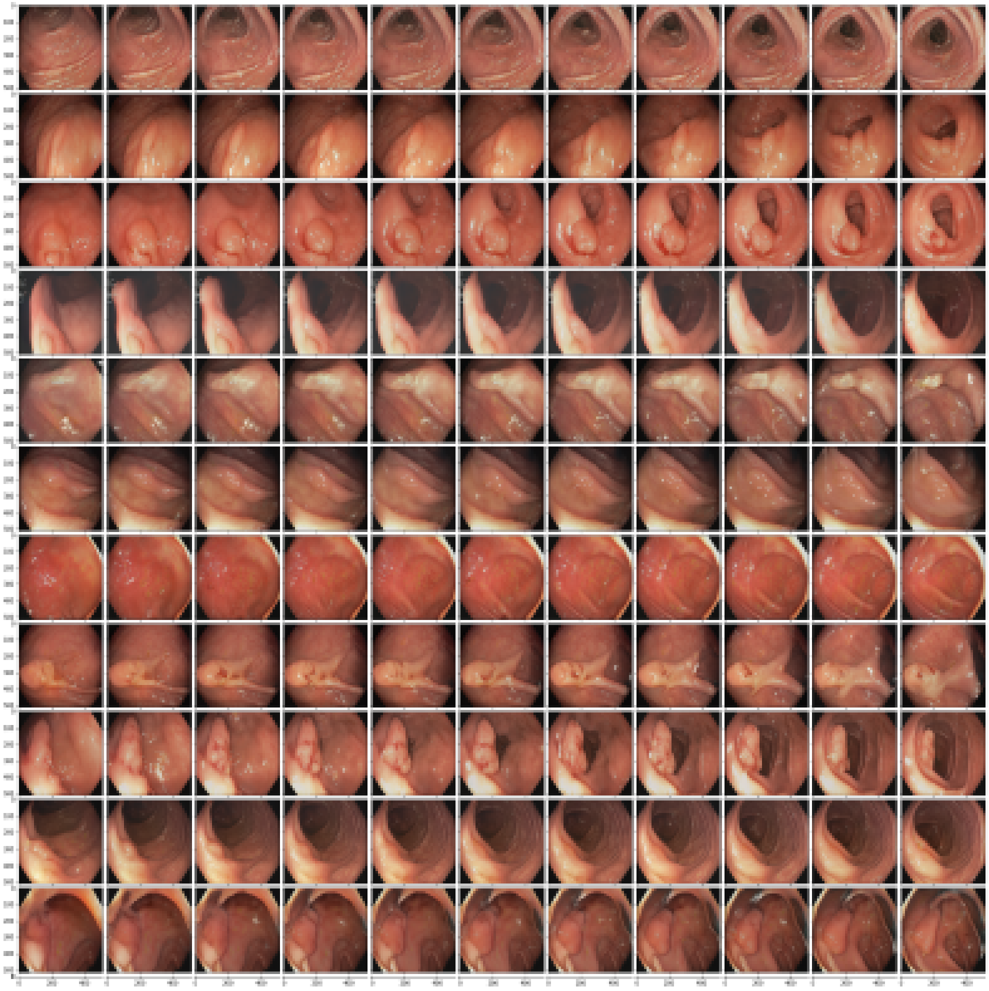


Figure 39: Example of modifications caused by changing principal component 5 in layers 3-6.

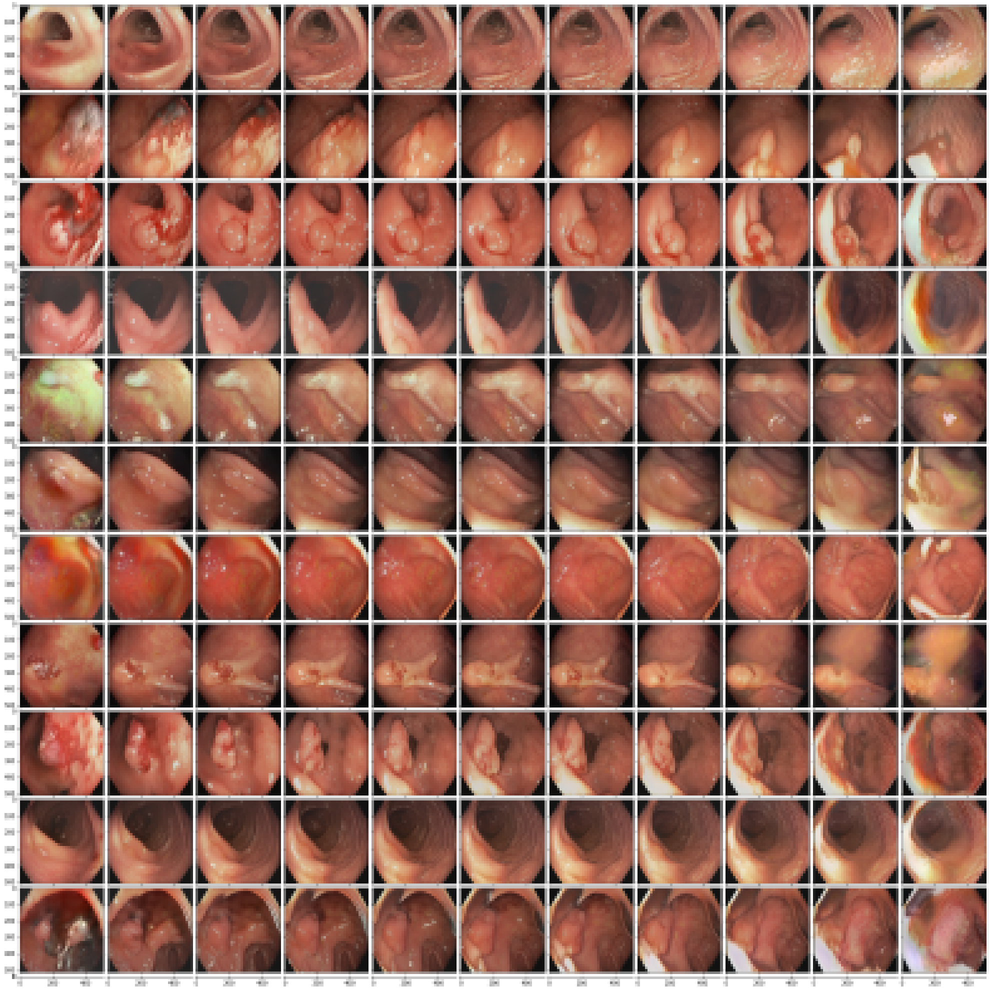


Figure 40: Example of modifications caused by changing principal component 0 in layers 6-16.