Transformation from Synthetic Colonoscopy Videos into Realistic Ones via Adversarial Training

Jiabo Xu

A report submitted for the course
COMP8755 Individual Computing Project
Supervised by: Mohammad Ali Armin, Nick Barnes
The Australian National University

November 2020

Except where otherwise indicated, this report is my own original work.

Jiabo Xu
2 November 2020

Acknowledgments

I would like to thank my supervisor Mohammad Ali Armin for his help thorough out my project, and thank Saeed Anwar for his instructive suggestion. Thanks doctor Florian Grimpen for providing the real colonoscopy data. Finally thank Nick Barnes for giving me this chance and providing me abundant CSIRO computation resources.

Abstract

The aim of this project is to create a deep-based model to transform synthetic colonoscopy videos into more realistic ones. It is an domain transformation task between our synthetic colonoscopy dataset and the real ones. The video transformation is done via continuous frames transformation with optical flow. To achieve this, a temporal cycle-consistent generative adversarial network (TCGAN) is designed and implemented. The model uses a temporal cycle-consistent constraint which consists of co-operations among three different losses. In terms of the temporal consistent loss, during the forward cycle the flow generator tries to transform synthetic frame into its real-alike next frame. Then, the reverse generator will transform the generated real frame back to its synthetic domain which should be exactly same as the corresponding next source frame. The backward cycle starts from the real frame and end on the next synthetic frame, which is the reverse process of forward cycle. In the addition, we use identity loss to restrict each generator on temporal level and optical flow loss to preserve the annotation of source video. We also implement two other existing methods and compare and analyze the performance among these models. Our TCGAN achieves smaller domain distance and better temporal consistency. Both qualitative and quantitative evaluations are performed to show the advantage and limitation of our TCGAN.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

For some machine learning projects, such as medical image segmentation it is difficult to obtain labelled data. This is due to that fact that annotating real data is a challenging task and sometimes need years of experience. Luckily, it is possible to generate labels for synthetic data. As a result, a good solution is to train a model by synthetic data, then generalize the its performance to real data domain. This is called domain adaptation. In this project, we propose a model to transform synthetic colonoscopy videos into more realistic ones with same annotation. Thus future supervised learning on synthetic data can be transferred to real domain. In this chapter, we will describe problem, motivation and existing challenges. It also describes, our research project scope and the contribution.

1.1 Problem Statement

It is known that labeled data is very useful for machine learning tasks, especially for supervised learning. Although there are a large amount of data nowadays, only few of them are labeled. High-quality human labeled datasets are even less. For some data, such as optical flow even human is unable to provide an accurate ground-truth. In medical images, obtaining labeled data is difficult and challenging. Besides, the ethical concerns and patients privacy prevent the data distribution and manipulation.

1.2 Motivations

Considering the above problem, it is beneficial to generate annotated synthetic data with non-learnable algorithms. However, model trained by simulated data may not be generalized on real data. If a model is trained on a synthetic dataset which has a similar data distribution to real one then it can indirectly increase the model performance on real data. This project aims at proposing a model to transform simulated colonoscopy video frames to real data while it preserve the colon structure in each frame.

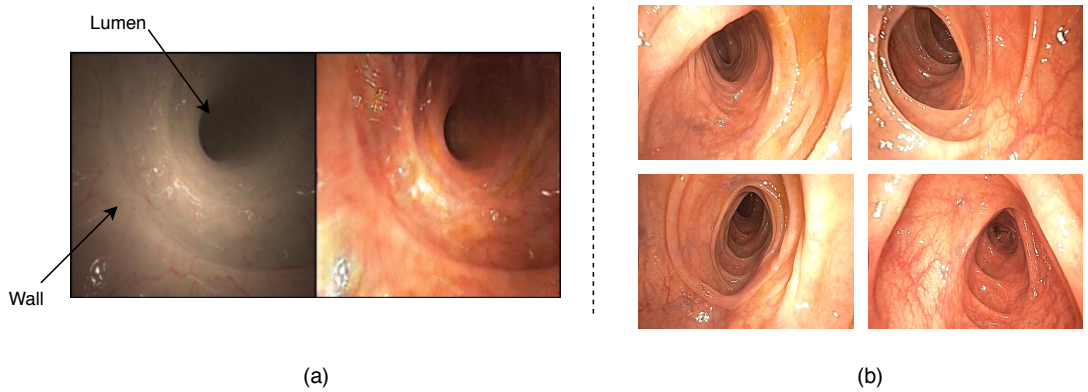

Figure 1.1: Example of our result and dataset. (a) One frame of transformed colonoscopy image(Left) and its original image(Right). (b) Four sample real colonoscopy images, which belongs to our target domain.

1.3 Challenges

1. The first challenge derives from the domain difference. As shown in Figure 1.1, we have to transform frames from synthetic domain to real domain. There are much differences between these two domains. The first one is the color difference. In synthetic domain, frames are in gray color while there are blood vessels in red color. However, in real domain, frames are in red, yellow and orange, depending on the light condition. The second one is the structure difference. There are all smooth and regular wall in the synthetic domain but complicated and irregular in real domain. Finally, its the light difference. The light is the bright spot existing in both domain. They are reasonable in human view, but it is usually interpreted as noise by the model.

2. The consistency of the video is also challenging. In general, single-frame domain transformation only makes assure whether the transformed frame looks like a frame in the target domain. As a result, feeding a pair of consecutive frames to these models may output two frames with different appearance, even thought they still look like images in the target domain. This is necessary to add innovative constraints to force the frame-base consistency

3. As the final generated video will be regarded as labelled data for other machine learning tasks, and they should preserve the annotation of the source input. The annotation in our case is the optical flow. For each pair of generated consecutive frame, their optical flow should be very close to the corresponding synthetic source frames.

1.4 Project Scope

Computer Vision. The project is a high-level computer vision task on medical imaging. We have to propose a method to understand the high-level(e.g. object) and

low-level(e.g. texture) information inside the data. Our target data is colonoscopy video frames. The simulated colonoscopy dataset in our study has different texture and structure from general images, which makes the task much more challenging.
Domain Adaptation. Domain adaptation is a branch of transfer learning. It has many representations, for this project, it can be interpreted as mapping data from its original domain to a target domain. Generative adversarial network(Goodfellow et al. [2014]) is currently one of the best models for domain adaptation.
Deep Learning. Using deep learning techniques, in computer vision low level and high level feature can be extracted from images. In terms of feature abstraction, it is unavoidable to mention Convolutional Neural Network(Krizhevsky et al. [2012]). CNN is a best tool for extracting feature maps from images. It also makes complex image-to-image mapping possible. However, there are many different CNN structures. Determining the most suitable CNN structure for a specific task is challenging.

1.5 Contributions

The contributions of this project are two-fold.

1. Recent approaches have not yet targeted to solve the video domain transformation problem on medical image. Based on the standard cycle-consistent structure, we create and implement the temporal-consistent GAN which is able to transform the domain meanwhile keep the temporal consistent of colonoscopy videos. The model is evaluated on our colonoscopy dataset both qualitatively and quantitatively.

2. We exploit the task on our dataset with two more existing models. By comparing the performance, we further understand the speciality of our data and task. It is meaningful for future research on the dataset and relevant tasks.

1.6 Report Outline

The report content includes Chapter 2 which describes the works related to the project task, Chapter 3 describes the theory of methods which are proposed to achieve our aim. Chapter 4 is about implementation details and results analysis for each proposed methods. Chapter 5 concludes the research work and leads to the potential future researches related to this project.

Chapter 2

Background and Related Work

Section 2.1 briefly explains the background knowledge to support this report. The recent related work corresponding to the background and technologies of the project introduced in Section 2.2.

2.1 Background

Machine Learning and Optimization. The project is a machine learning task, which can be described as training a mathematical model with the data we have to fit the specific task as good as possible. For the idea of machine learning, please refer to Machine Learning and Pattern Recognition (Bishop [2006]). More specifically, "training" is to find a set of variables of the mathematical model. With those variables, the model performs the best on the specific task. After training, the model does not need those data anymore. For most cases of machine learning, "specific task" is defined by loss functions, so-called optimization objective. Optimization is the method to find the relatively best variables for the model. For the idea of non-discrete optimization, please refer to Convex Optimization (Boyd and Vandenberghe [2004]).

Artificial Neural Network and Deep Learning. The most famous term "Neural Network" is one set of a successful machine learning models. In early stages, it was a composition of linear transformation. Each linear transformation is a network layer. Later on, it becomes useful when researchers added non-linear activation function to each layer. According to Hornik et al. [1989], theoretically 3-layer network is able to fit any kinds of function. However, in practice, the network size and training complexity make it infeasible. Hence, Deep Learning appears to tackle this problem. By adding more layers to the network in a reasonable way, learning complex problem becomes possible. Since then, deep-based method becomes a universal framework for function fitting tasks. For more knowledge of Deep Learning, please refer to Deep Learning LeCun et al. [2015].

Adversarial training. The idea of adversarial training introduced by Goodfellow et al. [2014]. It is able to generate high-quality data from noise. The overwhelming performance and perfect theory basis makes it famous very quickly. The adversarial training contains two networks; the generator and the discriminator. The generator

tries to generate data that is able to fool the discriminator, while the discriminator tries to identify if the input data is generated or real. Training them successively until these two players reach the Nash Equilibrium. More details and varieties of it will be explained in 2.2.

The above three fields are the top-level knowledge related to this report. More research-oriented knowledge will be introduced in the related work.

2.2 Related Works

Convolutional Neural Network Convolutional Neural Network obtains outstanding performance on different supervised learning tasks in Computer Vision. Well-known CNN structure such as VGG (Simonyan and Zisserman [2014]), shows a powerful feature extraction capability. Residual block (He et al. [2016]) mitigates the gradient vanishing problem for very deep networks. Fully Convolutional Network (Long et al. [2015]) has refreshed idea towards perception layer. U-Net (Ronneberger et al. [2015]) has outstanding performance on pixel-wise prediction tasks. Miscellaneous structures, such as skip-connection (Huang et al. [2017]), dilate convolution (Yu and Koltun [2015]) and pixel shuffling (Shi et al. [2016]) are proposed to increase efficiency and performance of the CNN models.

Generative Adversarial Network The GAN framework as introduced above contains two adversarial models, the generator and the discriminator. The generator is the target model to be finally utilized while the discriminator plays the role as the loss function. Different from general loss function which restricts the output to a fix target, it dynamically changes during training and represents an abstract target which is hard to be digitalized. It surely brings about training difficulty and extend the training duration. But the advantage is also appealing. The abstract loss, such as real or fake and belong to same group or not, can be interpreted by the discriminator. Further, many ill-posed problems such as super-resolution and style transfer can be better solved with GAN. The application of GAN includes generating data from latent space, super-resolution, de-noise, colorization, image transformation, video generation, text generation. In terms of training, it converges when the gaming between generators and discriminators finally reach to the Nash Equilibrium. Hence, the generator and discriminator should have equal capability which is also known as balanced training, Otherwise the training will be much unstable and non-convergent. Famous models, such as LSGAN (Mao et al. [2017]), WGAN (Arjovsky et al. [2017]) and GP-WGAN (Gulrajani et al. [2017]) are proposed to deal with this problem by modifying the loss function. Conditional GAN (Isola et al. [2017]) and cycle-consistent GAN(Zhu et al. [2017]) try to deal with the problem by adding additional constraints. Historical buffer (Shrivastava et al. [2017]) and PatchGAN (Li and Wand [2016]) deal with unstable training problem by modifying the training condition of the discriminator. StackGAN (Zhang et al. [2017]) and PG-GANs (Karras et al. [2017]) generate high quality results by utilizing novel model and training strategies.

Domain Adaptation Domain adaptation is a sub-topic under transfer learning.

One of the basic idea of it is to map two or more domains (data distribution) to one feature space and make their distance inside this feature space as close as possible. Under such feature space, performing machine learning task on the source domain can be easily generalized to target domains. GAN plays an important role on achieving this task. Liu and Tuzel [2016] proposed a CoGAN method which combines two generators and discriminators into one coupled system. Inside the system, both generator share the weight of first few layers and both discriminator share the weight of last few layers. By such weight sharing strategy, given same latent vector as input to theses generators, they will generate image with shared high level semantics with different low level features. For example, both generator will generate digit one but in different color or both generators will generate a smile face but on different person. In this case, the model tries to find the shared feature space of both domains so that it can transfer the share semantic from one domain to another. S+U GAN (Shrivastava et al. [2017]) tries to refine the synthetic images to real-like images through a direct transformation. In this network, the input is the synthetic image and the output is its corresponding image in real domain. However, it only performs well on specific conditions. For instance, the images should be in grayscale, the complexity of target domain should be low and the difference between both domain sould be small. Otherwise direct transformation will be less effective. In terms of domain adaptation in medical images, Mahmood (Mahmood et al. [2018]) applies the idea of S+U GAN and create reversed domain transformation network which transforms real endoscopy images into synthetic ones to remove patient-specific features. Rau (Rau et al. [2019]) extended pix2pix model by adding real domain image to the conditional discriminator so that it is able to predict the depth of real colonoscopy image after training on synthetic data.

Image-to-Image Transformation Image-to-image translation is like language translation. Given an input image (sentence) from a specific domain (language), the task is to translate it into one of the corresponding image in another domain. In general, the input domain and the output domain are restricted by training data (pairs). Hence, it is somehow like domain adaptation on images. The main difference is that domain adaptation focus on the performance of other tasks on target domain, while image-to-image translation concentrates on the quality of output images. For example, fashion to cloth image translation (Yoo et al. [2016]) achieves the task by adding one more discriminator to identify if the generated cloth image is associated to the input fashion image. The detailed behavior like conditional GAN (Mirza and Osindero [2014]) concatenates the source image with a sample from three-image set including the generated image, the ground truth and the irrelevant ones. By only output true for source and ground truth pairs, the discriminator will be trained to learn if the cloth image is paired with the input fashion image. Pix2pix (Isola et al. [2017]) also applies the idea of conditional GAN but it makes it simpler. Different from fashion-to-cloth translation, Pix2pix generalizes the performance on any paired data sets and achieves state of art performance on paired data sets. It only uses one discriminator whose input is the generated image conditioning on its source images. However, both introduced methods are limited by paired data set and, in practice,

it is difficult to obtain paired/pixel-wise labeled image sets. CycleGAN (Zhu et al. [2017]) is created to deal with such limitation. By importing cycle-consistent loss, CycleGAN is able to translate unpaired images from one domain to another. The idea of cycle-consistent is that when transforming an image from one domain to the other and then transform it back to its original domain, the final output should be exactly same as the original one. The detail of it will be explained in detail in chapter 3. Oda et al. [2019] successfully transforms endoscopy CT image to real domain using CycleGAN with deep residual U-Net (Ronneberger et al. [2015]). However, same as standard CycleGAN, their work is not capable of generating frame-consistent video.

Video Generation The very first model VGAN (Vondrick et al. [2016]) simply uses 3D-convolution layers to generate small and short videos from scratch. Later on, temporal GAN (Saito et al. [2017]) separates the video generation into frames generation. Even though it can generate better result than VGAN, most generated results are still irrational. The latest model (Lu et al. [2018]) uses optical flow and divide the generation into optical flow generation and texture generation. However, same as previous models, it can only generate one second video with 64 times 64 frame size. It is obvious that video generation from scratch is too hard to be achieved. However, video generation conditioning on the input video(s) got much bigger success. Ruder (Ruder et al. [2016]) generalizes image style transfer (Gatys et al. [2016]) to video. Based on content loss, the main loss of style transfer, they introduce a temporal consistency loss, which uses optical flow warping and forces the model to make stylized video consistent along frames. Instead of the style transfer way, Wang (Wang et al. [2018]) applies GAN on the same task. They use optical flow as a warping factor of generator and a condition of the discriminator. Acquiring the advantage of GAN, their model generates videos in multi-style meanwhile persistent the style of each single video. Compared with video style transfer, video-to-video synthetic requires the corresponding target video of the source video rather than only one more style frame. Hence, the style applied on the source video can be as realistic as patterns from real world, like semantic to city street. For our work, we do domain adaptation, hence there is no frame-to-frame correspondence but domain correspondence on our training set. Furthermore, instead of only using predicated optical flow, we have ground truth optical flow on our synthetic source videos.

2.3 Summary

This chapter introduces and discusses the background and related works of the research. The background defines the basic technology condition meanwhile the related work are the nutrition to support the research. Most of the related work are inspirational, even though they may not be directly applied during the research work. The work of GANs is usually task and data sensitive. Hence, it is necessary to find the underlying attributes of specific problems and solve it contrapuntally. The next chapter is about the description and explanation of existing methods we used for experiments and our proposed methods for solving the challenges.

Chapter 3

Methodology

This chapter covers the three main models used for this research. The content includes the intuition and the support of each model. The first model is inspired by Shrivastava et al. [2017] and Mahmood et al. [2018], which tries to directly transform the synthetic domain into real domain using adversarial training. Due to the poor result of the first model, we proposed a model based on cycleGAN structure Zhu et al. [2017]. Finally, we create a temporal-consistent GAN which generalizes the image-to-image cycle structure into the video-to-video version. This model uses optical flow as a constraint which is an extension of our previous model. In 3.1 and 3.2, most of methods are applied directly from the original work. In 3.3, we developed our model based on previous methods, this is elaborated in details for the temporal-consistent GAN.

3.1 Direct Domain Transformation (Model1)

The model is a version of Shrivastava's S+U GAN (Shrivastava et al. [2017]) extended with optical flow. Similar to traditional adversarial training, it has a generator and a discriminator. Given a synthetic colonoscopy image z, the generator tries to learn the mapping $G : z \rightarrow y$ where y is the corresponding real image of z. The discriminator tries to distinguish if the input image is real or fake. The best generation quality can be obtained by the optimization objective, adversarial loss(which is a function of the generator G) and the discriminator D as formulated below:

$$L_{adv}(G, D) = E_{y \sim Pdata(y)}[\log D(y)] + E_{x \sim Pdata(x)}[1 - \log D(G(x))] \tag{3.1}$$

where G and D represent the set of weights of generator and the discriminator respectively, which can be regarded as mapping functions. The generator tries to minimize the adversarial loss when the discriminator reach its maximum.

$$G^* = \arg \min_G \max_D L_{adv}(G, D) \tag{3.2}$$

When training the discriminator, the loss is decoupled as

$$L_D(D) = -E_{y\ Pdata(y)}[\log D(y)] - E_{x\ Pdata(x)}[1 - \log D(G(x))] \tag{3.3}$$

Training generator and discriminator is alternate in one iteration. In general, one training of generator follows by one training of discriminator. However, this is not always the best setting. In S+U GAN, generator is trained fifty times followed by only one time training for discriminator per epoch. In other cases, for example, GP-WGAN trains generator one time while critics (discriminator) five times per epoch. In our research, we consider this as a hyper-parameter and adjust it based on the training result.

3.1.1 Buffer and Historical Buffer

Since the discriminator may be trained more than once while the weight of generator is frozen, to prevent over-fitting, it is better not to only train it with the same fake images generated from the latest epoch. A possible method is to temporarily store the output fake mini-batches into a buffer and each time sample required amount of data to train the discriminator. S+U GAN provides an extended method of the above idea. They use a history buffer not only to store the generated fake image of the latest epoch but also to store the generated fake image along the history. Regarding the batch size as B. In our case, the detailed process is that each time a generated mini-batch of the fake image is fed into the buffer until the amount reaches the maximum buffer size. When the buffer is full, $\frac{B}{2}$ next incoming fake images will be randomly sampled and randomly replaced $\frac{B}{2}$ images in the history buffer. Before training the discriminator, $\frac{B}{2}$ images are sampled from both the history buffer and the buffer of generated image of latest epoch. Hence, the final batch size is still maintained as the original size B. The intuition of using the history buffer is straight-forward. The discriminator should not only be able to identify the latest generated image, but also the previous generated ones. Using history buffer prevents the discriminator from over-fitting to the latest generated images.

3.1.2 Self-regularization Loss

In addition to the adversarial loss, there is a self-regularization loss for generator to preserve the structure of the source image. It is the L1 loss between the input and the generated image as:

$$L_{reg}(G) = E_{x\ Pdata(x)}[||G(x) - x||_1] \tag{3.4}$$

Importing the self-regularization loss has two benefits. Firstly, preserves the structure of the source image and helps to preserve the its annotation. Secondly, it restricts the complexity of target distribution which leads to simplification of training. The drawback of latter former benefit id that if the source domain is vastly different from the target domain, preserving the structure of source domain can cause a conflict in the domain transformation. In other word, the training will either be inclined to identity mapping or non-convergence. The analysis of our results prove the idea,

which will be discussed in the next chapter. The overall objective for this model is:

$$G^* = \arg\min_G \max_D L_a dv(G, D) + \lambda L_{reg} \tag{3.5}$$

3.1.3 Optical Flow Loss

Up to now, it is the direct domain transformation on single frame. As we have the ground truth optical flow, we extend the original model to double frames version. We modify the data loader to randomly take two continuous frames from the synthetic data set. For each iteration, the first frame will be fed into the generator to make the first real-alike image. Then obtain the next real-alike image from the second synthetic frame. As the two input frames are continuous, the output should be continuous. Therefore, we use L1 loss as the flow loss between the ground truth optical flow and the predicted optical flow between these consecutive generated frames. The loss formula is:

$$L_{flow}(G) = E_{x\ Pdata(x)}[||Op(G(x_1), G(x_2)) - GT_{flow}||_1] \tag{3.6}$$

where Op represents a pre-trained optical flow network whose weights are fixed through out the training. We simply use end-point-error as the loss function. More details of the pre-trained network, such as the accuracy will be discussed in the next chapter.

The intuition of using the optical flow loss is that the generator should generate continuous frames if input frames are under same frame interval as the training data and preserve the structure based on the ground truth optical flow. This constraint may not guaranty the continuity of the generated frames. For example, if the model is not perfectly trained, there may be some noise on some generated frames. Since generating new frames does not based on the previous generated frame, there may be no noise on the next frame or the noise is on different place

The overall loss function is:

$$\mathcal{L}(G) = \mathcal{L}_{adv} + \lambda \mathcal{L}_{reg} + \sigma \mathcal{L}_{flow} \tag{3.7}$$

where λ and σ are the weight of adversarial loss and optical flow loss respectively. Except other hyper-parameters, such as learning rate and batch size, which do not significantly impact the generated image quality, λ and σ are the main parameter for tuning. For example, we can ignore the loss by setting corresponding weight to zero.

3.2 Cycle-consistent GAN (CycleGAN) (Model2)

The cycle structure is fully based on the idea of CycleGAN (Zhu et al. [2017]). Standard GAN only has one mapping $G : X \to Y$ transforming source domain X to a target domain Y. In contrast, CycleGAN uses one more mapping $F : Y \to X$ which transforms target domain back to the source domain. As there is no enough con-

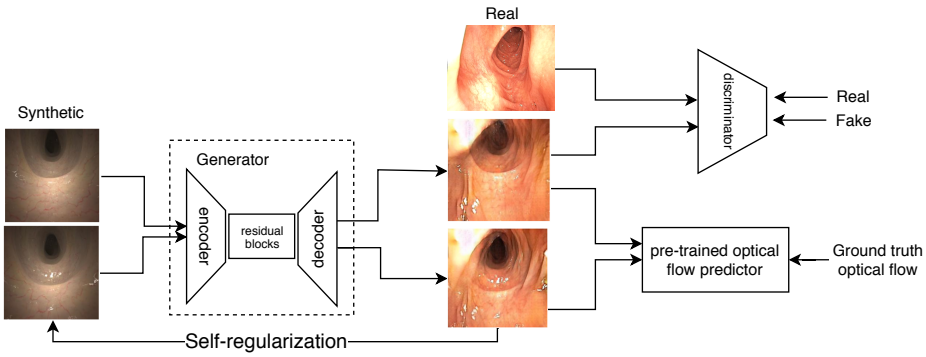

Figure 3.1: The overall model structure for direct domain transformation. For each iteration, a pair of consecutive synthetic colonoscopy frames are fed into the generator. The generated real-alike frames feed into the optical flow prediction network. The predicted optical flow should be close to the ground truth

straint on unpaired domain transformation, inverse training is introduced as a new constraints.

3.2.1 Cycle-consistent Loss

Given any image x from source domain, $G(x)$ transforms it to target domain. Then, using $F(G(x))$ to transform it back to the source domain, the target image can be the input x itself. The idea of CycleGAN is to construct a target pair for each unpaired images. As the logic of transforming source to target domain is not different from transforming target to source domain, the other constraint is the inverse version which starts from image y in the target domain to $G(F(y))$ supposed to be itself. Both constraints are together called cycle-consistent loss. The mathematical expression is

$$\mathcal{L}_{cyc}(G,F) = \mathrm{E}_{x\ Pdata(x)}[||F(G(x)) - x||_1] + \mathrm{E}_{y\ Pdata(y)}[||F(G(y)) - y||_1] \tag{3.8}$$

where P_data represents the data distribution of specific domain, also known as current data set.

3.2.2 Adversarial Loss

In terms of the adversarial loss, refer to LSGAN (Mao et al. [2017]) average mean square error (MSE) is used to replace cross entropy loss for more stable training. The total adversarial loss for generators is:

$$\mathcal{L}_G(G,F) = \mathrm{E}_{x\ Pdata(x)}[(D_g(G(x)) - 1)^2] + \mathrm{E}_{y\ Pdata(y)}[||D_f(F(y)) - 1||_2] \tag{3.9}$$

The adversarial loss for the discriminator D_g of domain Y is:

$$\mathcal{L}_{Dg}(D_g) = \mathrm{E}_{y\ Pdata(y)}[(D_g(y) - 1)^2] + \mathrm{E}_{x\ Pdata(x)}[(D_g(G(x)))^2] \tag{3.10}$$

and for the discriminator D_f of domain X is:

$$\mathcal{L}_{Df}(D_f) = \mathrm{E}_{x\ Pdata(x)}[(D_f(x)-1)^2] + \mathrm{E}_{y\ Pdata(y)}[(D_f(F(y)))^2] \tag{3.11}$$

The overall discriminator loss is the average of $\mathcal{L}_{Dg}(D_g)$ and $\mathcal{L}_{Df}(D_f)$

3.2.3 Identity Loss

In addition to the cycle-consistent loss, there is another loss called identity loss (Taigman et al. [2016]) which helps the generated image preserve the structure as the source image. For both generators, if the input image already belongs to the target domain, then the generator will do identity mapping $G : Y \to Y$ and $F : X \to X$. The formula is:

$$\mathcal{L}_{idt}(G,F) = \mathrm{E}_{y\ Pdata(y)}[\mathcal{L}(G(y),y)] + \mathrm{E}_{x\ Pdata(x)}[[\mathcal{L}(F(x),x)] \tag{3.12}$$

where $\mathcal{L}$ represents similarity metric for one image to the other one. Candidate measurements are L1 loss, Structure Similarity Index (SSIM) and Perceptual Loss (Johnson et al. [2016]).

The function of this loss is to restrict the real-alike image close to the source. However, it somehow influences the adversarial loss, even though there is no domain conflict, especially when the source domain is too much different from the target domain. For example, if the source domain is frog and target domain is horse, using this loss will make the generated image not like both. However, if use self-regularization loss like the previous model in this case, the training will either be closed to identical mapping or non-converge, which makes the drawback even worse.

3.3 Temporal-consistent GAN (TCGAN) (Model3)

The previous cycle structure is used on pure single image transformation. We improve it to generate continuous frames. We call the domain transformation with two generators the double-generative model. The general double-generative model can be presented as Figure 3.2(a), there are two generative models G and F. The task of G is to transform data from domain X to domain Y while the task of F is to transform data in reverse manner. There is no limitation, which means the generated data of both model can be either one in the other domain. In terms of CycleGAN Figure 3.2(b), cycle-consistent constraint is applied so that there are one-to-one correspondences between both domains. When transforming the generated data back to its original domain, it must be same as the source data. For our model Figure 3.2(c), we modify the constraint to frame-level by breaking one-to-one relationship into chain relationship based on the video data. In this way, the transform-backed version is no longer the source frame but the next frame. The constraint is same as the cycle-consistent in spatial level, meanwhile it is also able to restrict the generated frame to be temporal consistent with the previous generated one. We call it temporal-consistent constraint.

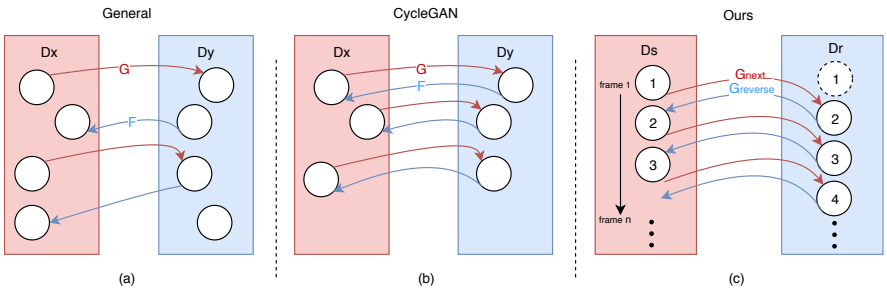

Figure 3.2: The logic of double-generative models, such as cycleGAN and our TCGAN. In general (a), there are two generative models G and F. The task of G is to transform data from domain X to domain Y while the task of F is to transform data in reverse manner. There is no limitation, which means the generated data of both model can be either one in the other domain. In terms of CycleGAN (b), cycle-consistent constraint is applied so that there are one-to-one correspondence between both domains. When transforming the generated data back to its original domain, it must be as same as the source data. For our model (c), we modify the constraint to frame-level by breaking the one-to-one into chain relationship due to the video data. In this way, the transform-backed version is no longer the source frame but the next frame.

3.3.1 Temporal-consistent Loss

More specifically, CycleGAN adds new constraints by transforming the generated image back to its original domain using another generator, $x \rightarrow F(G(x)) \rightarrow x'$. It is valid if the x' looks like other images in its original domain rather than x. Thus, we make the supervised pair to be the next frame of the synthetic video. In this way, we have $G_{flow} : S^n \rightarrow R^{n+1}$ where the subscript indicates the frame number, S indicates the synthetic colonoscopy domain and R represents the real colonoscopy domain. It plays the role as the original G but transforms the domain and frame at the same time. Furthermore, we have $G_{self} : R^n \rightarrow S^n$ which is exactly same as the original F. Therefore, the forward cycle finally transforms S^n to $S^{n+1\prime}$ which is supposed to be same as the ground truth S^{n+1}. In terms of backward cycle, the process becomes $R^n \rightarrow G_{flow(}G_{self}) \rightarrow R^{n+1\prime}$ 3.3. We call the new loss temporal-consistent loss. The formula for temporal-consistent loss is:

$$\mathcal{L}_{tc}(G_{flow}, G_{self}) = \mathrm{E}_{s\ Pdata(S)}[||G_{self}(G_{flow}(x)) - s^{n+1}||_1] + \mathrm{E}_{r\ Pdata(R)}[||G_{flow}(G_{self}(r)) - r^{n+1}||_1] \tag{3.13}$$

Nevertheless, the temporal-consistent loss is insufficient to force G_{flow} to generate the next frame. Even though there are both forward cycle and backward cycle, it is still possible that $S^n \rightarrow G_{flow} \rightarrow R^n \rightarrow G_{self} \rightarrow S^{n+1\prime}$. The function of G_{flow} and G_{self} may keep exchanging during training, which makes it difficult to converge. Therefore, we use identity loss to force each generator to obey their order. In term of temporal consistency, the order of G_{flow} is to predict the next frame. Therefore, its identity

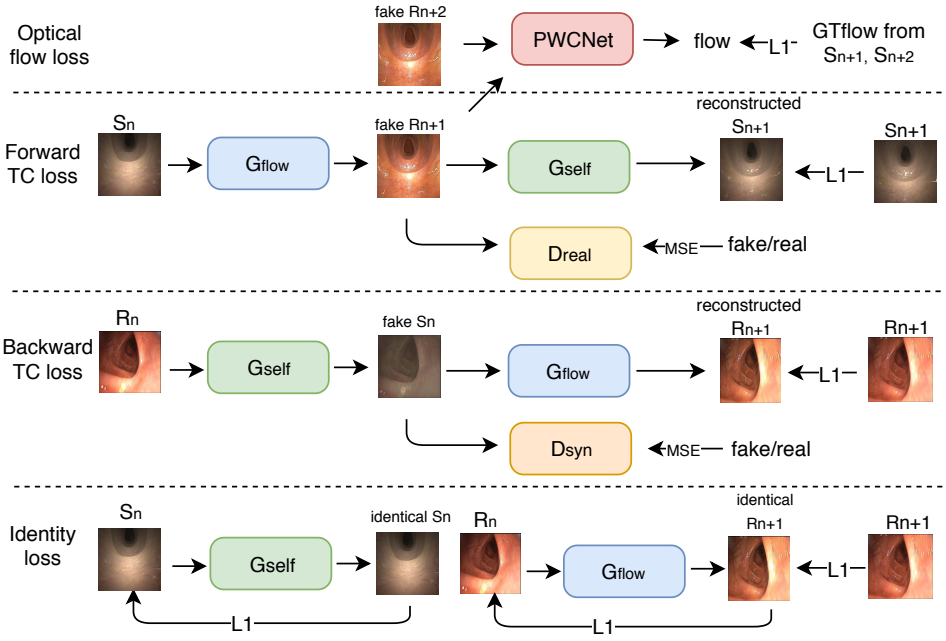

Figure 3.3: The overall model structure for TCGAN. The first generator of forward cycle will not only transform the domain but also estimate the optical flow at the same time. Indeed, it transforms the input to the next frame of the target domain. It is trained by the cycle-consistent loss, and forces the reserved generation (G_{self}) to be close to the ground truth synthetic frame.

loss only trains it to do the prediction. For G_{self}, its duty is to preserve the frame unchanged in temporal space. The identity loss encourages the identity mapping. The formula is as following:

$$\mathcal{L}_{idt}(G_{flow}, G_{self}) = \mathrm{E}_{r\ Pdata(R)}[\mathcal{L}(G_{flow}(r^n), r^{n+1})] + \mathrm{E}_{s\ Pdata(S)}[\mathcal{L}(G_{self}(s^n), s^n)] \tag{3.14}$$

In other word, we can regard the identity loss as a loss to release the training burden of the cycle training step, especially for the G_{flow}. In this way, G_{flow} does not have to learn the domain transformation and frame prediction at the same time only from the temporal-consistent loss. Further, it restricts the function of G_{self} so that it cannot influence the training of G_{flow}. The overall loss function up to this stage can be presented as:

$$\mathcal{L}(G_{flow}, G_{self}) = \mathcal{L}_{adv}(G_{flow}, G_{self}) + \lambda\mathcal{L}_{cyc}(G_{flow}, G_{self}) + \beta\mathcal{L}_{idt}(G_{flow}, G_{self}) \tag{3.15}$$

where adversarial loss $\mathcal{L}_{adv}$ is same as the standard cycle structure.

Moreover, as G_flow has already been trained to predict its next frame, it is not necessary to introduce ground truth optical flow into the model. Importing the ground truth optical flow into the model is still an option. Similar to the direct domain transformation model, we minimise the difference between the predicted optical flow from generated frames with the ground truth.

$$L_{flow}(G_{flow}) = E_{s\ Pdata(S)}[||Op(G_{flow}(s^n), G_{flow}(x^{n+1})) - GT_{flow}||_1] \tag{3.16}$$

Referring to Figure 3.2(c), the temporal consistent is achieved frame by frame for each video. However, it is not necessary to train the model video by video. We break videos into two or three (for optical flow loss) continuous segments. In this way, training will not over-fit to any specific video but generalize on the given underlying optical flow distribution, hence also stabilizes the training. The pseudo-code for implementing temporal-consistent model is in the Appendix 7 2.

3.4 Network Structures

The network structures are fixed setting for all three models. There are three structures being used, which are encoder-decoder with residual blocks, encoder-decoder with skip-connection and PatchGAN (Li and Wand [2016])

The first one comes from real-time image transfer (Johnson et al. [2016]). It uses an encoder-decoder structure (Hinton and Salakhutdinov [2006]) mainly including a stack of residual blocks (He et al. [2016]). The encoder-decoder network structure helps preserve the high-level information of the image, especially helpful for image-to-image translation tasks. It extracts the high-level feature of the input image and expand low-level features to approach to the target domain. For example, in our case the basic high-level information of colonoscopy images is spatial direction of the lumen. From human's perception, it is easy to roughly identify the direction of

the lumen, even though the image may be under different lighting condition. The encoder-decoder structure can indirectly help the neural network mimic such human capability by encoding it into low dimensional space. This admits our idea that we want the generated image looks like an image of the target domain but it should still preserve its semantic. The encoder-decoder structure contains several downsampling layers with same amount of upsampling layers. The downsampling is achieved by two striding convolution layers, whereas the upsampling uses $\frac{1}{2}$ striding. The bottleneck layers are a stack of residual blocks with stride of one and same output channel size (256).

The second structure applies the skip-connection between the shallow and deep layers. FCNs (Long et al. [2015]) obtains successful results on pixel-wise semantic segmentation with the help of skip-connections. In addition, FCNs has been improved by U-Net (Ronneberger et al. [2015]), especially on medical image. In detail, skip-connections here connect the information of layer i and layer $n - i$ where n represents the total amount of layers. In FCNs, the connection is simply adding the output of both layers whereas in U-Net, it is a concatenation. We use U-net as the second possible structure for the generator.

The third structure is for the discriminator. We use PatchGAN. Instead of identifying true or false in image-level, patchGAN provides the value for identifying each overlapped 70×70 patches. It is achieved simply by removing the fully-connection (FC) layers and make sure the perception field of the output feature map is 70×70. In terms of the performance, changing from the image-level to the patch-level identification indirectly helps the generator generates fine details. Moreover, binary classification, such as the real or fake classification does not necessarily require complex model. Hence removing the heavy FC layers have small effect on the performance of discriminator and meanwhile it makes the model light. Furthermore, avoiding the size restriction from FC layers, the discriminator can take any size of input intrinsically.

3.5 Summary

This chapter describes and analyzes three models we have used for this research. It includes the intuition, loss function and algorithm details. The direct domain transformation model is theoretically possible, but very difficult to be trained on our dataset in practice. We reduce the training complexity by using the cycle structure. Then we extend the single-frame (standard) cycle structure to temporal-consistent version (ours) so that it performs better on video data. In the addition, network structures and detailed training conditions are discussed and universally used in all three models. Other implementation and experiments details are discussed in the next chapter. Moreover, in the next chapter performance of each model is discussed in both qualitative and quantitative ways.

Chapter 4

Experiments

This chapter mainly explains the experiment details and analyzes the result of each model. The data set used in our experiments for each model 3 are identical . The result evaluation metrics are consist of both qualitative and quantitative analysis. The qualitative analysis basically compares the image quality among generated images of each model, the synthetic image and the real image in a subjective manner. The quantitative analysis is two-fold. One uses end point error (EPE) (Dosovitskiy et al. [2015]) to evaluate the temporal consistency. The other one uses FrÃl'chet Inception Distance (FID) (Heusel et al. [2017]) to measure the image quality and the domain distance.

4.1 Data Set and Pre-processing

We used 8000 synthetic colonoscopy and 2741 real frames for training, and 2000 unknown synthetic frames from two long videos were used for testing. The synthetic data is simulated by a colon model and haptic device (De Visser et al. [2010]). The real data captured from patients by our specialist which are confidential. The appearance of synthetic and real frames are shown in 1.1. To balance the training between the generator and the discriminator, we randomly sample synthetic data as much as real data for training per epoch. The ground truth for optical flow is available for simulated data from simulator. The original data size is $3 \times 540 \times 676$ for both domain and $2 \times 540 \times 676$ for optical flow. Our real data is collected by Olympus 190 colonoscopy, which is equipped by a fish-eye lens camera. As a result, the real frames are distorted. We do the fisheye correction (Scaramuzza et al. [2006]) to remove fish-eye distortion. A sample of corrected real frame is shown in Figure 4.1(a) bottom. We clean the real data. By observing our synthetic data and real data, we find the diversity of the synthetic data is much less than real data. The diversity mostly reflects on the camera motion and the light condition. We remove real data with extreme lighting condition, wall-only image and blur image. Besides, some real image contains water jet which is used for cleaning the colon and some contains patient-specific features. These two types of real data are also eliminated. The sample images are presented in Figure 4.1. The data cleansing strategy aims at shortening the domain distance and stabilizing the training. After data cleansing, 1472 real

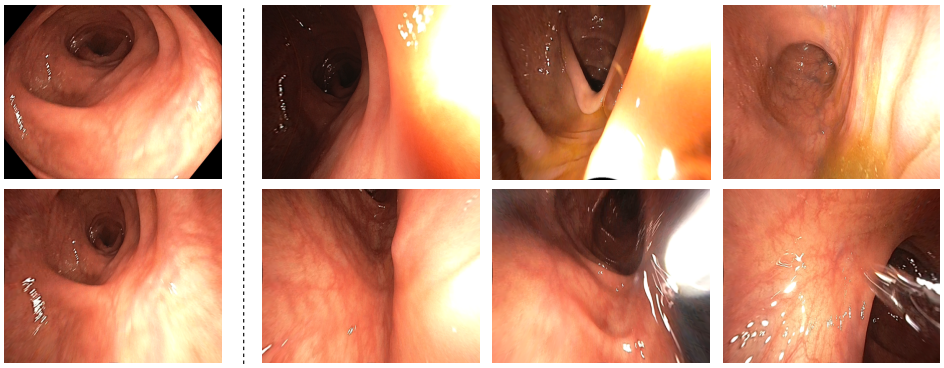

Figure 4.1: Data pre-processing for real domain. (a) The top image is original image collected by fish eye camera. The bottom one is the result after fish correcting. (b) Six types of real image to be eliminated. From the top left to right bot, they are too dark and over-exposure, special structure, special lumen, wall only, blurred by motion and water jet.

frames are left. In the addition, we resize the data into size 256×256 to reduce the computation burden. Another option is to randomly crop the data from its original size into 256×256. Intuitively, it is more suitable for our data, as there is no object, what we focus on is only structures. We performed same data augmentation on the corresponding optical data. Then re-scale the value of the optical flow by $\frac{w/h_{current}}{w/h_{original}}$, where w/h means the width or height of the optical flow data.

4.2 Common Training Conditions

We use the common training condition for all of the three models. Firstly, we use history buffer with size equals to 50 mini-batches. Each mini-batch size contains 16 groups of data. Moreover, we use Instance Normalization (Ulyanov et al. [2016]) on generators and Batch Normalization (Ioffe and Szegedy [2015]) on the discriminator. We do not use neither dropout layer nor weight decay. We use Adam (Kingma and Ba [2014]) optimizer with beta equals to $(0.5, 0.999)$ and the learning rate equals to $2e^{-4}$. We decay the learning rate linearly after 200 epochs and training up to 400 epochs depending on the specific setting. The program is implemented in PyTorch (Paszke et al. [2017]). The models are trained on four Nvidia P100 GPUs on a SLURM cluster.

4.3 Unsuccessful Results

The direct domain transformation is not successful. Hence, we only analyze the single-frame quality and ignore the temporal consistency. From Figure 4.2, we find the generated image either cannot learn the target distribution or stay far from the target distribution and never converges. Increasing the weight of self-regularization loss will make the model converge to identity mapping. If decrease it, the adversarial loss will not be able to lead the model to do any domain transformation. As a result,

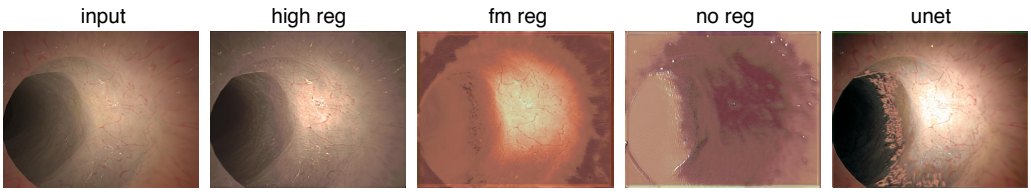

Figure 4.2: Results from direct domain transformation model after the model converge or a large amount of epochs. The first column is the input synthetic colonoscopy image and the second column is the generated image corresponding to the input from the model trained with high weight for self-regularization loss. The third column is trained with perceptual self-regularization loss. The fourth column does not use self-regularization. The last column bases on the model of the first column and replaces the residual generator with U-Net.

it will generate unrealistic features on images. In terms of optical flow loss, it mainly controls the temporal consistency and has fewer benefits for domain transformation. We adjust the weight of the optical flow loss and observes less impact to the generated results. There are two possible reasons for such poor performance. Optimistically, it because of the training complexity of GAN. Transforming one image to another is even harder than generate image from latent vector. Because the generator should firstly encode the source distribution to much lower dimensional space. However, it is avoided when directly generate image from latent space. As the encoding layers are the first few lowers of the generator, the gradient vanish problem makes the learning for encoding much hard, which even enlarges the training complexity. We have trained a GP-WGAN on our data set to prove this. The result (Figure 4.3) is closer to the real domain. From this point of view, it is still possible to achieve successful results by discovering the best hyper-parameters. Those hyper parameters include the learning rate, the number of training time for generator and discriminator per epoch, the weight for each loss, etc. Hence, it is a discrete global optimization problem, which require a huge amount of computation. Pessimistically, such structure is not able to do domain transformation on our data set due to the large distribution gap. The direct domain transformation model derives from S+U GAN (Shrivastava et al. [2017]) which got successful results on refining greyscale eye and hand gesture image. In their data set, the synthetic image is much close to the real one, especially in the greyscale space. However, there is no research on quantifying the domain transformation difficulty. Hence, from pessimistic view, we can only subjectively think direct domain transformation does not work on our data set.

4.4 Successful Results

We infer successful results from both cycle-consistent GAN and our temporal consistent GAN. We train our baseline model under the following parameters. The weight of cycle-consistent loss $\lambda = 150$ and the weight of identity loss $\beta = 75$. Data is randomly cropped into 256×256. The generator is U-Net, since we observe more realistic results from it. Other hyper-parameters refer to section 4.2. It is trained

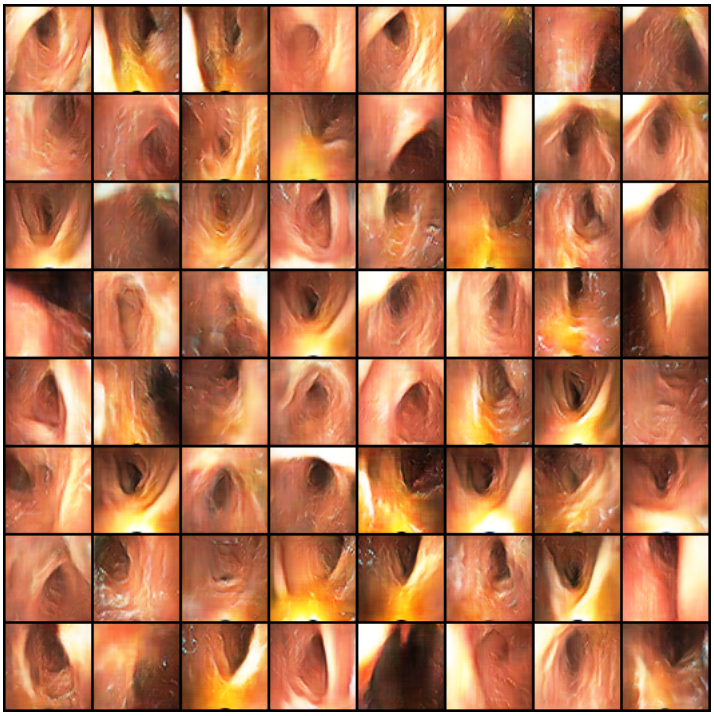

Figure 4.3: Results from using GP-WGAN to generated real-alike image from 128-d noises. Each generated image are in 64×64 size. In total there are 64 generated images from 64 different latent vectors.

for 350 epochs, which takes around 24 hours. The other model only modify specific parameters based on the baseline model.

4.4.1 Qualitative Analysis

Sample results are shown in Figure 4.4. The baseline model is able to generate very consistent videos. There are almost no noise in the generated video. However, the realness is quite shiftless. It is like colorizing the synthetic ones into orange. Hence it only captures one safest color of the real domain. Reducing the weight of identity loss and decreasing the weight of cycle-consistent loss force the model to learn more about the real domain. Nevertheless, it not only loses the consistency but generates some deformed structures. More noises emerge on single frame and disappear or become larger on the later frames. Consequently, the cycle-consistent model achieves meaningful results, but it cannot narrow the distance from the real domain.

Our TCGAN achieves better temporal consistency and much better realness. We test the TCGAN with three more different settings by changing the identity loss function and adding optical flow loss. The result from TCGAN with optical flow loss is very consistent, we can find light black spot moving together with the view. The spot and blur derives from the inaccurate optical flow predictor. TCGAN and TCGAN with perceptual loss generates more realistic results with less flaws than the standard CycleGAN. The consistency of them are not poor, but mask-like noise with

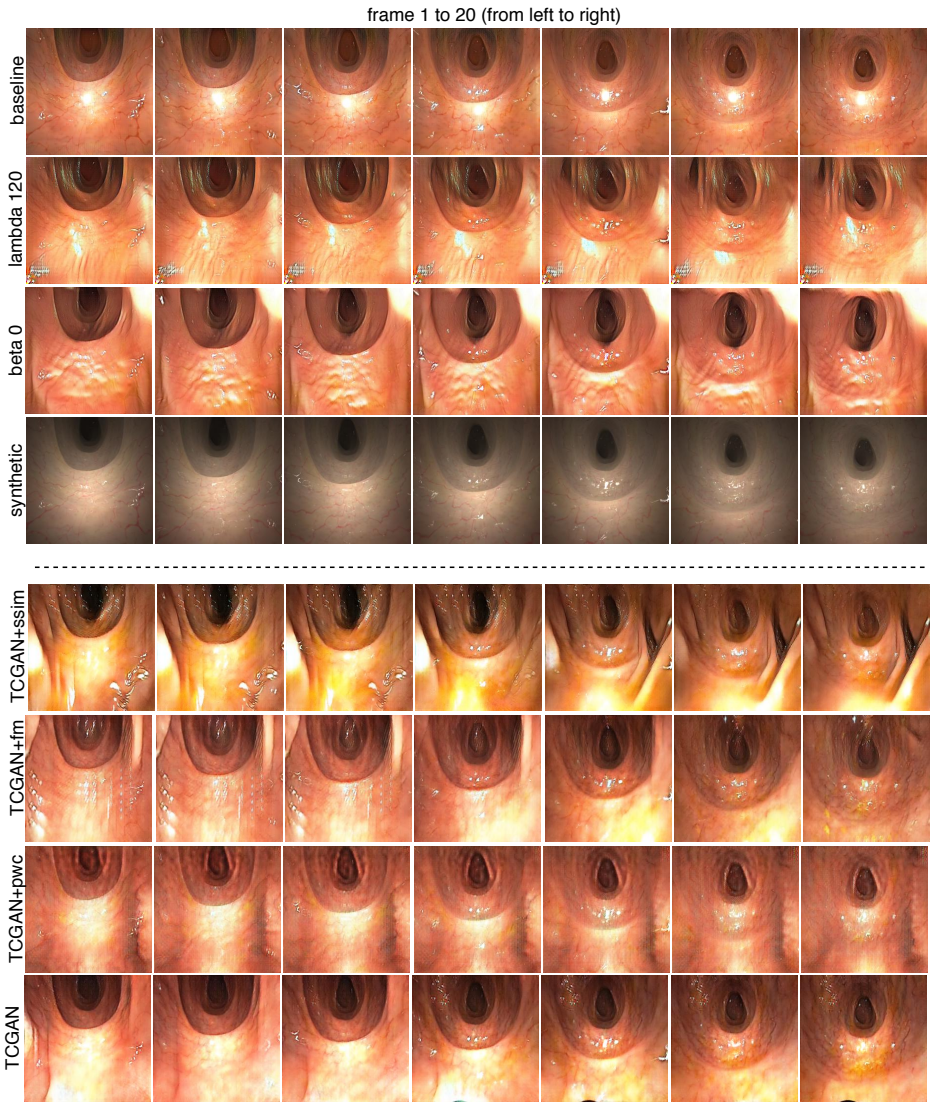

Figure 4.4: A sequence of sample results from our baseline model (top 3) and TCGAN (bottom 4). From the left to right they are the 1st frame to the 20th frame. To make the motion obvious, we display one for every three frames. The top blocks are results from standard CycleGAN with different training conditions. Results in the bottom blocks come from our TCGAN. More specifically, the first row shows the baseline results. The second row is the model trained with $\lambda = 120$. The third row is the model trained with $beta = 0$. The fourth row is the corresponding synthetic frames. With regard to the bottom four rows, "+ssim" means using SSIM as the identity loss and remain others unchanged. "+fm" means use perceptual loss as the identity loss. The "+pwc" version uses optical flow loss and SSIM identity loss. The weight σ of the optical flow loss is set to be 10. The pure version (TCGAN) uses the hyper-parameter referring to the baseline model. Other settings are in Section 4.2

identical shapes occasionally appear on continuous few frames.

4.4.2 Quantitative Analysis

Quantitative analysis are performed separately for temporal consistency and domain distance. In terms of temporal consistency, we measure the end point error (EPE) between the optical flow of synthetic and generated frames. As for domain distance, we use a modified FrÃl'chet Inception Distance (FID) to evaluate how well our generated frames fits the real domain.

Temporal Consistency

Approach	EPE-GT	EPE-Pred	real-alike	no mask-noise	no bright-spot	FID-768
real	-	-	T	T	T	0.44
synthetic	1.01	0	F	-	-	*
l1	1.29	0.42	T	F	T	1.74
*x1						
*x3						
*x4						
*x5						
*unet						
res9	1.38	0.50	T	F	F	1.57
sig10	1.17	0.49	F	-	-	1.91
sig5	1.19	0.36	T	T	F	1.61
*sig0.01						
cyclegan	1.28	0.24	F	-	-	1.75
cyclegan+op	1.19	0.55	F	-	-	1.91
*TCGAN	2.85	2.34	T	F	T	1.49
TCGAN+op	1.22	0.31	T	T	F	1.44

Models	EPE-GT	EPE-Pred
synthetic	3.84	0
baseline	4.41	**1.13**
TCGAN+ssim	1.24	2.53
TCGAN+fm	1.79	2.11
TCGAN+pwc	4.72	1.51
TCGAN	2.97	1.38

Table 4.1: Average EPEs for models we trained on our dataset. EPE-GT means regarding the ground truth as the target optical flow. EPE-Pred means regarding the predicted optical flow on synthetic frames as the target optical flow. We use EPE-Pred to reduce the impact of the inaccuracy of the optical flow predictor (PWCNet).

Approach	EPE-Pred	real-alike	no mask-noise	no bright-spot
synthetic	**0**	F	-	-
l1	1.38	T	F	T
unet	1.30	T	F	T
res9	0.59	T	F	F
sig10	0.40	F	-	-
sig5	0.38	T	T	F
cyclegan	**0.27**	F	-	-
cyclegan+op	0.61	F	-	-
TCGAN	2.41	T	F	T
TCGAN+op	0.35	T	T	F

The temporal consistency can be indirectly measured by the optical flow similarity between source videos and generated videos. We use EPE to measure the optical flow misalignment. In our case, EPE describes the difference between the optical flow of our generated frames and the optical flow of their corresponding source frames. Same as training the model with optical flow loss, the estimated optical flow between two generated frames is obtained from the pre-trained PWCNet. To mention that, worse EPE does not necessarily indicate the worse consistency, since consistent frames may have different optical flow from the synthetic pairs. In our task, besides the consistency, we also want to preserve the annotation (optical flow). Thus, EPE is the most suitable measurement. The formula of EPE is $E = ||Op(r'_n, r'_{n+1}), Target_{flow}||_1$ where r'_n means the generated real-alike frame n and Op represent the pre-trained PWCNet. Because PWCNet is pre-trained on general datasets, directly using it onto our medical image will lead to accuracy problem. Sample prediction accuracy of PWCNet is shown in Figure 4.5. We predicted optical flows from our synthetic test set by the PWCNet and make the predicted result as the target to corresponding generated frames. We finally obtain the average EPE on our test set for each model. Results are shown in Figure 4.1. Results of the second column (EPE-GT) regards the ground truth as the target.

Figure 4.5: Sample inaccurate optical prediction results from PWCNet. The dense optical flow is visualized in RGB by color wheel model (Baker et al. [2011]). The EPE between the ground truth and the predicted optical flow is 0.96. The value of the optical flow data is in range 2 to 15, which means, for a pixel in one frame, its corresponding pixel in the next frame is at least 2 pixel and at most 15 pixel distance from its original position.

Models	FID-2048	FID-768
real	110.27	0.44
basline	313.34	1.33
TCGAN+ssim	189.61	**0.77**
TCGAN+fm	**181.74**	0.78
TCGAN+pwc	329.68	1.47
TCGAN	231.04	0.93

Table 4.2: FID scores for models we trained on our dataset. The meaning of the model name referring to previous figure caption. FID takes image quality into consideration. Hence it may not always align with human's feeling

If the optical flow predictor is accurate, EPE of synthetic (first row) should be very close to one. However, it is relatively large (as around 3.8). Besides, there should not be value less than 3.8, but there are. Hence, it is not rigorous to refer to the EPE-GT column. Instead, we change the target to the predicted optical flow. By measuring the distance in the predicted space, the EPE Pred score turns to be stable and reliable. Results of EPE-Pred shows that videos generated from the baseline model have the closest annotation with source videos, as it is like simply painting frames orange. Our unsupervised TCGAN is the second best one in terms of EPE-Pred, which illustrates that the temporal consistent model learns and generalizes the optical flow of training data even without the supervised information. Theoretically, TCGAN with PWCNet is supposed to generate the best EPE-Pred due to the supervised information it uses. However, it is the third best. The most possible reason is also the inaccuracy of the optical flow predictor. Potential solutions to tackle such inaccuracy problem are introduced in the future work section **??**.

Domain Distance

We use a modified FID to evaluate how well our generated frames fits the real domain. FID uses pre-trained Inception V3 (Szegedy et al. [2016]) on ImageNet as a feature extraction component. By applying adaptive pooling on the last pooling layer of Inception V3, we can obtain a 2048 dimensional vector for each given image. Then calculate the mean and covariance of vectors extracted from same data distribution. Regard them as Multivariate Guassian distribution and using FrÃl'chet Distance to measure the distribution gap. Given two Multivariate Guassian distribution, the FrÃl'chet Inception Distance can be calculated by the following equation:

$$FID = ||\mu_s, \mu_r||^2 + Tr(\Sigma_1 + \Sigma_2 - 2(\Sigma_1\Sigma_2)^{\frac{1}{2}}) \tag{4.1}$$

where μ_i and Σ_i are the mean and covariance of the feature distribution of each image domain. Two distributions are much closer to each other when their FID is close to zero. On the contrary, the FID will be much larger. Due to the feature difference between image from ImageNet and our medical image, the feature extractor is not much accurate in this way. We verify the accuracy by splitting our real data into two segment and calculating the FID for them. The score is around 110 but the correct score should be close to zero. Hence, we make a little bit changes on the output of Inception V3. We change the feature output to the second late pooling layer whose dimension is 768. The score between the real domains now becomes 0.4. However, we can use the FID score to only measure the relative distance among our results. Comparative results on both FID-2048 and FID-768 are displayed in Table 4.2. Together with EPE-Pred, we find the baseline result achieve the best consistency but almost the largest domain distance. Because the baseline results are like simply colorization. The best domain generated results derive from TCGAN with SSIM and Perceptual identity loss. However, for these two models mask-like noise occurs frequently in the generated videos, which decreases the EPE-Pred. In terms of TCGAN with PWCNet, we think it acquires the shortest domain distribution. However, as it becomes more blur than others results, the Inception network dramatically rises the distance.

4.5 Summary

We implement three proposed models in methodology section and train them with different hyper-parameters and training conditions. In the result section, typical results from eight successful trained models are selected for evaluations. We show the trade-off of domain distance and temporal consistency of standard CycleGAN. By comprehensive consideration on both qualitative and quantitative evaluations the TCGAN achieves the best performance. However, mask-like noises occasionally emerge on results. TCGAN with optical flow loss successfully removes the noise but blur and spots are introduced due to the inaccurate result from PWCNet. In next section we conclude the overall research work and discuss potential solutions to improve our proposed model as future work.

Chapter 5

Conclusion

In this project we firstly investigate the difficulty of the video domain transformation task on our colonoscopy dataset with two existing methods. One we called direct domain transformation model which is mostly inspired by S+U GAN. The other one is the famous unpaired image-to-image translation method, cycleGAN. Experiments results show that the domain transformation and temporal consistency are trade-off when simply using these methods. As a result, we proposed temporal-consistent GAN by combining the benefits of cycle-consistent GAN and optical flow loss. It achieves relatively good temporal and spatial performance on video domain transformation on our colonoscopy data set. Moreover, our transformed data also preserves the annotation of source data, which is beneficial for domain adaptation. Both qualitative and quantitative evaluations are performed to determine the performance of the proposed methods. The remaining challenges, such as the image blur of much consistent results and mask-like noise of much realistic results, will be investigated in the future work.

Chapter 6

Future Works

There are important future works need to be considered based on our research. Firstly, as we already have the ground truth optical flow, it is not necessary to predict the optical flow once again. Even though we use the state of art optical flow prediction network, there are inaccuracy on the predicted results due to the special feature on our colonoscopy data. If we can directly use the ground truth optical flow, we may remove the impact of the prediction inaccuracy. An immature idea to handle it is to use conditional discriminator. The conditional discriminator D_{op} exists simultaneously with the normal discriminator D_{real} as in our TCGAN. D_{op} tries to maximize the conditional probability $P(real|s_n, s_{n+1}, flow_{n,n+1})$ and $P(fake|I_j, I_k, flow_{n,n+1})$ where the I_j and I_k indicates any other image pairs whose optical flow is not same as $flow_{n,n+1}$. The intuition is to train a discriminator which is able to determine if the optical flow can be generated from a pair of consecutive frames. To prevent over-fitting of D_{op} for the synthetic frames, augmented data must be used for training the discriminator. However, there is a problem for this idea. The discriminator performs image-level classification, but the optical flow prediction is a pixel-level task. Hence, it may be difficult to use image-level model to mimic pixel-level capacity. Experiments and verification about it can be done in the future.

Using another pre-trained network(PWCNet) as one component of our model is not always an ultimate solution. It will be better, if we can internalize the function of the pre-trained network into our model. Our TCGAN is created just because of this motivation. However, TCGAN also needs the optical flow loss to enhance the performance. We expect a better end-to-end structure which uses less loss functions and auxiliary models.

With regard of our model itself, there are also space for improvement. In our project hyper-parameters are chosen only depends on intuition and observation due to time limitation. The hyper-parameter we have tried is not likely to be the best. There are essential explicit hyper-parameters including optimizer parameters, batch size, weights of each loss and etc. Finding the best hyper-parameters is a discrete global optimization problem which requires either exhausted search or meta learning. Further, there are also implicit hyper-parameters such as the network structure and the type of optimizer. In our project, we only exploit the effect of the model structure. The selection of those factors are left to be investigated.

Furthermore, domain adaptation on colonoscopy images can be exploited based

on our domain transformation task. After transforming the synthetic domain to real domain meanwhile preserving the annotation, we can train other tasks, such as polyps detection, depth prediction and optical flow prediction, on the real-alike space. Compared with model only trained on synthetic data set, this model may have better generalization ability on real domain. Therefore, more efforts are still required in this field.

Chapter 7

Appendix

Appendix 1: Final Project Description

Final project description:
This is a computer vision research project involving in medical image and domain transformation. The aim of the project is to use or create a deep-based method to transform synthetic colonoscopy videos into real-alike ones. It is a very challenge task. There are very limited related works specially targeting the medical image in this field. Moreover, the dataset used in this project is very special. In addition, the model training step is unavoidable and very time-consumption. Computation resources will be provided to the student who works on this project.

Detailed tasks:

1. The student should fully understand relevant technology on tackling this problem.
2. The student should create a model that is able to refine single synthetic colonoscopy images into realistic ones.
3. The student should extent the model on refining continuous video frames.

Expected outcomes:
The final model should take the synthetic colonoscopy video as its input and output the video that have same annotation as the input and looks as realistic as possible.

Appendix 2: Research Project Contract

Australian National University

INDEPENDENT STUDY CONTRACT PROJECTS

Note: Enrolment is subject to approval by the course convenor

SECTION A (Students and Supervisors)

UniID: _____u6435287_____

SURNAME: _____Xu_____ FIRST NAMES: _____JiaBo_____

PROJECT SUPERVISOR (*may be external*): _____Ali Armin_____

FORMAL SUPERVISOR (*if different, must be an RSSCS academic*): _____Nick Barnes_____

COURSE CODE, TITLE AND UNITS: _____COMP8755 individual projects 12units_____

COMMENCING SEMESTER ☒ S1 ☐ S2 YEAR: _____ _ Two-semester project (12u courses only): ☐

PROJECT TITLE:

Transformation from Synthetic Colorectal Video into Realistic Colorectal Video via Adversarial Training

LEARNING OBJECTIVES:

Practice research skills by doing literature reviews. Deeply study and understand GAN series, especially on medical images and videos. Be able to program all relevant neural network with PyTorch.

PROJECT DESCRIPTION:

Lack of labelled data leads to lots of troubles on machine learning tasks, especially in medical imageology. In terms of colorectal videos, labelled synthetic video is relatively easy to be generated. Based on these truths, this project aims at generating realistic colorectal videos from synthetic ones via a specific deep learning model.

There are several tasks required to be achieved.

First, the student should fully understand relevant technology on tackling this problem.

Second, the student should create a model that is able to refine single synthetic colorectal images into realistic ones.

Australian National University

Third, the student should extent the model on refining continuous image frames.
The final model should take the synthetic colorectal video as its input and output the video that have same annotation as the input and looks as realistic as possible.

ASSESSMENT (as per the project course's rules web page, with any differences noted below).

Assessed project components:	**% of mark**	**Due date**	**Evaluated by:**
Report: style: _____research report_____ (e.g. research report, software description...,)	80		(examiner)
Artefact: kind: _____software_____ (e.g. software, user interface, robot...,)	10		(supervisor)
Presentation :	10		(course convenor)

MEETING DATES (IF KNOWN):

STUDENT DECLARATION: I agree to fulfil the above defined contract:

许家博 ..... 01/03/2019 .....
Signature Date

SECTION B (Supervisor):

I am willing to supervise and support this project. I have checked the student's academic record and believe this student can complete the project. I nominate the following examiner, and have obtained their consent to review the report (via signature below or attached email)

..... 24-03-19 .....
Signature Date

Examiner:
Name: Nick Barnes Signature .....
(Nominated examiners may be subject to change on request by the supervisor or course convenor)

REQUIRED DEPARTMENT RESOURCES:

Research School of Computer Science *Form updated Jun 2018*

Appendix 3: Description of software

- List of all program code: "./dataloader/*" contains scripts of data loaders for different tasks. They are all implemented by myself. "./model/*" contains scripts of neural network structures. Only the "xxxborrow.py" scripts are borrowed from github for test the correctness of my work. "./sample_results" stores some sample generated results of the model. "./pwcnet", "./pytorch_ssim" and "./wgantest" are all third party packages from github with small modification to fit our implementation. All the other scripts are implemented by myself. "configc.py" stores the training and testing configuration details, which should be modified every time before training. "functions.py" stores the useful functions which are necessary in the training. "run.py" is for running the direct domain transformation model. "runc.py" is for running cycleGAN and "runc_v2.py" is for running our TCGAN. However, the dataset is extremely large and the real data is confidential. We cannot provide the data here.

- Test for the code: testing scripts for the code correctness is in jupyter notebook files, the suffix is ".ipynb" the file name corresponds to the test item. The correctness of training the neural network can be indirectly verified by the results. Hence, it is not necessary for writing testing scripts.

- Experiments detail. The training and implementation details are in section 4. The version control details is in the README file.

Appendix 4: README file

The original README is a markdown file inside the code directory. We suggest you to read that one.
Transform synthetic colonoscopy video into realistic ones with TCGAN.
Package requirements
pytorch 1.1.0 with cuda 10.0
tensorboardx (tensorflow 1.13)
cupy 6.0
pillow
matplotlib
All the hyper-parameters can be modified in ‘config.py‘.
Modify the ‘train_name‘ variable inside ‘config.py‘ to specify the place for storing temporary files (models, images, logs). Run the following command to generate ‘config.json‘

    python config.py

Train the model by specific config file

    python run.py -c yourtrain_name ‘run.py‘ is for direct domain transformation model

‘runc.py‘ is for cycleGAN

'runc_v2.py' is for TCGAN
You can trace the training information inside ./preservation. Observing generated images in ./preservation/images. Tracing the loss by running
tensorboard –logdir ./preservation/logs
The sustainable models are stored in ./preservation/temp_models.
To test the model, modify specific items inside config.py and run the program as above.
If training it on slurm, you can use the script submit.sh after generating 'config.json' e.g. Training TCGAN with setting in train_name for 24 hours
bash slurm.sh runc_v2.py train_name 24h

Appendix 5: Algorithm

The detailed adversarial training logic for direct domain transformation is shown in the pseudo-code. Given the input data and specific hyper-parameters, the model will output the trained generator G_θ and discriminator D_ϕ. From the line 2 to 9, the generator is trained for T_g times, from line 10 to 14, the discriminator is trained for T_d times.

Algorithm 1 adversarial training for direct domain transformation network G_θ

Input: synthetic colonoscopy images $x_i \in \mathcal{X}$, real colonoscopy images $y_i \in \mathcal{Y}$, total required epochs E, number of training of generator per iteration $\mathcal{T}_\}$, number of training of discriminator per iteration $\mathcal{T}_\lceil$, the batch size B

Output: G_θ and D_ϕ

1: **for** $e = 1$ to E **do**
2: **for** $t = 1, \ldots T_g$, **do**
3: 1. Sample a mini-batch of two continuous frames x_i, x_{i+1} from $\mathcal{X}$
4: 2. obtain $r_i = G_\theta(x_i)$ and $r_{i+1} = G_\theta(x_{i+1})$
5: 3. sample $\frac{B}{2}$ of r_i to the history buffer H and put all r_i to normal buffer H'
6: 4. obtain $f_i = P(r_i, r_{i+1})$
7: 5. calculate loss $\mathcal{L}$ on single frame and optical flow loss in (3.7)
8: 6. update θ by using optimizer on overall loss $\mathcal{L}$
9: **end for**
10: **for** $t = 1, \ldots T_d$, **do**
11: 1. Sample a mini-batch of single real frames y from $\mathcal{Y}$ and sample $\frac{B}{2}$ single generated frames r from history buffer H and sample $\frac{B}{2}$ from normal buffer H'
12: 2. calculate loss $\mathcal{L}_\mathcal{D}$ in (3.3)
13: 3. update ϕ by using optimizer on discriminator loss $\mathcal{L}_\mathcal{D}$
14: **end for**
15: **end for**

The detailed implementation logic for extended cycle structure(without optical flow) is shown in the pseudo-code. Given the input data and specific hyper-

parameters, the model will output the trained generator G_{next}, $G_{reverse}$ and discriminator D_{real} and D_{syn}. From the line 2 to 14, both generators and discriminators are trained for E epochs. From line 3 to 11, both generators are trained, including the forward cycle(line 3-5) and backward cycle(line 6-8). From line 12 to 14, both discriminators are trained.

Algorithm 2 adversarial training frame-consistent model without optical flow.

Input: synthetic colonoscopy images $s^i, s^{i+1} \in \mathcal{S}$, real colonoscopy images $r^i, r^{i+1} \in \mathcal{R}$, total required epochs E, number of training of generator per iteration $\mathcal{T}_\}$, number of training of discriminator per iteration $\mathcal{T}_\lceil$, the batch size B

Output: G_{next}, $G_{reverse}$, D_{real}, D_{syn}

1: **for** $e = 1$ to E **do**
2: Sample a mini-batch of two continuous frames s^i, s^{i+1} from $\mathcal{S}$ and two continuous frames r^i, r^{i+1} from $\mathcal{R}$
3: obtain $r^{i+1}_{fake} = G_{next}(s^i)$ ▷ Forward cycle
4: obtain $s^{i+1}_{rec} = G_{reserve}(r^i_{fake})$
5: obtain $r^{i+1}_{idt} = G_{next}(r^i)$
6: obtain $s^i_{fake} = G_{reverse}(r^i)$ ▷ Backward cycle
7: obtain $r^{i+1}_{fake} = G_{next}(s^i_{fake})$
8: obtain $s^i_{idt} = G_{reverse}(s^i)$
9: sample $\frac{B}{2}$ of r^{i+1}_{fake} to the history buffer H_r and s^i_{fake} to history buffer H_s
10: calculate frame-consistent loss $\mathcal{L}_{fc}$ (3.13), identity loss $\mathcal{L}_{idt}$ (3.14) and adversarial loss $\mathcal{L}_{adv}$ (3.9)
11: update G_{next} and $G_{reverse}$ by using optimizer on overall loss $\mathcal{L}$ (3.15)
12: Sample a mini-batch of single real frames y from $\mathcal{Y}$ and sample $\frac{B}{2}$ single generated frames r from history buffer H_r and s from history buffer H_s
13: calculate adversarial loss $\mathcal{L}_{adv}$ in (3.10) and (3.11)
14: update D_{real} and D_{syn}
15: **end for**

Bibliography

Arjovsky, M.; Chintala, S.; and Bottou, L., 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, (2017). (cited on page 6)

Baker, S.; Scharstein, D.; Lewis, J.; Roth, S.; Black, M. J.; and Szeliski, R., 2011. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92, 1 (2011), 1–31. (cited on pages x and 26)

Bishop, C. M., 2006. *Pattern recognition and machine learning*. springer. (cited on page 5)

Boyd, S. and Vandenberghe, L., 2004. *Convex optimization*. Cambridge university press. (cited on page 5)

De Visser, H.; Passenger, J.; Conlan, D.; Russ, C.; Hellier, D.; Cheng, M.; Acosta, O.; Ourselin, S.; and Salvado, O., 2010. Developing a next generation colonoscopy simulator. *International Journal of Image and Graphics*, 10, 02 (2010), 203–217. (cited on page 19)

Dosovitskiy, A.; Fischer, P.; Ilg, E.; Hausser, P.; Hazirbas, C.; Golkov, V.; Van Der Smagt, P.; Cremers, D.; and Brox, T., 2015. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, 2758–2766. (cited on page 19)

Gatys, L. A.; Ecker, A. S.; and Bethge, M., 2016. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2414–2423. (cited on page 8)

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y., 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680. (cited on pages 3 and 5)

Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; and Courville, A. C., 2017. Improved training of wasserstein gans. In *Advances in neural information processing systems*, 5767–5777. (cited on page 6)

He, K.; Zhang, X.; Ren, S.; and Sun, J., 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778. (cited on pages 6 and 16)

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S., 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, 6626–6637. (cited on page 19)

Hinton, G. E. and Salakhutdinov, R. R., 2006. Reducing the dimensionality of data with neural networks. *science*, 313, 5786 (2006), 504–507. (cited on page 16)

Hornik, K.; Stinchcombe, M.; and White, H., 1989. Multilayer feedforward networks are universal approximators. *Neural networks*, 2, 5 (1989), 359–366. (cited on page 5)

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q., 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4700–4708. (cited on page 6)

Ioffe, S. and Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, (2015). (cited on page 20)

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A., 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1125–1134. (cited on pages 6 and 7)

Johnson, J.; Alahi, A.; and Fei-Fei, L., 2016. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, 694–711. Springer. (cited on pages 13 and 16)

Karras, T.; Aila, T.; Laine, S.; and Lehtinen, J., 2017. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, (2017). (cited on page 6)

Kingma, D. P. and Ba, J., 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, (2014). (cited on page 20)

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E., 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105. (cited on page 3)

LeCun, Y.; Bengio, Y.; and Hinton, G., 2015. Deep learning. *nature*, 521, 7553 (2015), 436. (cited on page 5)

Li, C. and Wand, M., 2016. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision*, 702–716. Springer. (cited on pages 6 and 16)

Liu, M.-Y. and Tuzel, O., 2016. Coupled generative adversarial networks. In *Advances in neural information processing systems*, 469–477. (cited on page 7)

Long, J.; Shelhamer, E.; and Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3431–3440. (cited on pages 6 and 17)

Lu, K.; You, S.; and Barnes, N., 2018. Deep texture and structure aware filtering network for image smoothing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 217–233. (cited on page 8)

Mahmood, F.; Chen, R.; and Durr, N. J., 2018. Unsupervised reverse domain adaptation for synthetic medical images via adversarial training. *IEEE transactions on medical imaging*, 37, 12 (2018), 2572–2581. (cited on pages 7 and 9)

Mao, X.; Li, Q.; Xie, H.; Lau, R. Y.; Wang, Z.; and Paul Smolley, S., 2017. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2794–2802. (cited on pages 6 and 12)

Mirza, M. and Osindero, S., 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, (2014). (cited on page 7)

Oda, M.; Tanaka, K.; Takabatake, H.; Mori, M.; Natori, H.; and Mori, K., 2019. Realistic endoscopic image generation method using virtual-to-real image-domain translation. *Healthcare Technology Letters*, (2019). (cited on page 8)

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A., 2017. Automatic differentiation in pytorch. (2017). (cited on page 20)

Rau, A.; Edwards, P. E.; Ahmad, O. F.; Riordan, P.; Janatka, M.; Lovat, L. B.; and Stoyanov, D., 2019. Implicit domain adaptation with conditional generative adversarial networks for depth prediction in endoscopy. *International journal of computer assisted radiology and surgery*, 14, 7 (2019), 1167–1176. (cited on page 7)

Ronneberger, O.; Fischer, P.; and Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 234–241. Springer. (cited on pages 6, 8, and 17)

Ruder, M.; Dosovitskiy, A.; and Brox, T., 2016. Artistic style transfer for videos. In *German Conference on Pattern Recognition*, 26–36. Springer. (cited on page 8)

Saito, M.; Matsumoto, E.; and Saito, S., 2017. Temporal generative adversarial nets with singular value clipping. In *Proceedings of the IEEE International Conference on Computer Vision*, 2830–2839. (cited on page 8)

Scaramuzza, D.; Martinelli, A.; and Siegwart, R., 2006. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, 45–45. IEEE. (cited on page 19)

Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A. P.; Bishop, R.; Rueckert, D.; and Wang, Z., 2016. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1874–1883. (cited on page 6)

Shrivastava, A.; Pfister, T.; Tuzel, O.; Susskind, J.; Wang, W.; and Webb, R., 2017. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2107–2116. (cited on pages 6, 7, 9, and 21)

Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, (2014). (cited on page 6)

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826. (cited on page 27)

Taigman, Y.; Polyak, A.; and Wolf, L., 2016. Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200*, (2016). (cited on page 13)

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V., 2016. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, (2016). (cited on page 20)

Vondrick, C.; Pirsiavash, H.; and Torralba, A., 2016. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, 613–621. (cited on page 8)

Wang, T.-C.; Liu, M.-Y.; Zhu, J.-Y.; Liu, G.; Tao, A.; Kautz, J.; and Catanzaro, B., 2018. Video-to-video synthesis. *arXiv preprint arXiv:1808.06601*, (2018). (cited on page 8)

Yoo, D.; Kim, N.; Park, S.; Paek, A. S.; and Kweon, I. S., 2016. Pixel-level domain transfer. In *European Conference on Computer Vision*, 517–532. Springer. (cited on page 7)

Yu, F. and Koltun, V., 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, (2015). (cited on page 6)

Zhang, H.; Xu, T.; Li, H.; Zhang, S.; Wang, X.; Huang, X.; and Metaxas, D. N., 2017. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 5907–5915. (cited on page 6)

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 2223–2232. (cited on pages 6, 8, 9, and 11)