

© 2021 Zengming Shen

LEARNING TO MAP BETWEEN DOMAINS

BY

ZENGMING SHEN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Nuclear, Plasma and Radiological Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2021

Urbana, Illinois

Doctoral Committee:

Professor Rizwan Uddin, Chair
Professor Emeritus Thomas S. Huang, Director of Research
Professor Zhi-Pei Liang
Doctor Bogdan Georgescu, Siemens Healthineers
Assistant Professor Shiva Abbaszadeh
Assistant Professor Angela Di Fulvio

ABSTRACT

Humans consume visual content avidly. We are in the midst of an imaging revolution enabled by inexpensive digital cameras and the internet. Almost everyone's cell phone has a camera. The photos taken by these cameras are shared massively and rapidly on the internet. However, there is an asymmetry: Each individual can consume only limited visual content in his limited lifetime, such that only a chosen few are talented enough to both express and understand something unseen visually and effectively. The rest of us try to understand and express something unseen by translating them to something seen before. Similarly, in the medical image field and radiological science, tens of thousands of medical images (MRI, CT etc) of patients are taken. These medical images need to be studied and interpreted. In this dissertation, we investigate a number of data-driven approaches for mapping from an 'unseen' or hard to understand domain to a 'seen' or easy to understand domain. Our work includes mapping between two image domains and mapping from an image domain to a language domain, which in computer vision are called, respectively, image-to-image translation and image captioning. The presented methods not only help users to easily and accurately synthesize useful photos, but also enable new visual and linguistic effects not possible before this work. In the clinical diagnosis, these approaches can improve the accuracy and efficiency of the diagnosis process for the experienced radiologist. What's

more, the approach of mapping from image domain to text domain can mimic the work of the experienced radiologist for automatic medical report generation.

Part I: This part describes image segmentation, which can be treated as a special case of image-to-image translation. This part includes two works. The first work solves the anisotropic resolution problem for 3D medical image semantic segmentation in the Appendix A. The second work describes our US patented cross-domain medical image segmentation. The first domain has labels while the second domain has no labels; by designing a special domain mapping, we enable image semantic segmentation on the second domain. Both of these works can improve computer aided medical image interpretation and help the radiologist read the medical images more efficiently and accurately.

Part II: In the clinical diagnosis, in order to combine the advantages of multiple medical imaging modalities together, medical image registrations or cross domain image translation is needed. A crucial requirement for both is one to one correspondence. Because the medical images from multiple image modalities (such as MRI, CT) are from the same patients. This part presents learning a self-inverse network to realize one-to-one mapping for both paired and unpaired image-to-image translation.

Part III: In the clinical diagnosis, the final output of the diagnosis is in text domain (such as medical report, medical prescriptions etc). Since medical report writing based on medical image can be error-prone for inexperienced physicians, and time-consuming and tedious for experienced physicians, automatic generation of medical image report can make this tedious and difficult task efficient. This part expands to learn the mapping from the image domain to the language domain. Specifically, the mapping is done by learning a language representation

to form the language domain.

To my advisor, parents, lover and colleagues and friends for their love and support.

ACKNOWLEDGMENTS

First, let me express my deep sorrow over the recent passing of my adviser Thomas S. Huang and his wife Margret Huang. I still feel they are with me and I will always remember having dinner or lunch with them. May they rest in peace together.

Alyosha sometimes says, "If you wait until the last minute, it will only take a minute." I think he might have got it wrong this time. It actually took me more than one minute to write these acknowledgments. Nevertheless, I would like to thank Professor Thomas S. Huang for his guidance, inspiration, enthusiasm, and support throughout my years at UIUC. If a student is a generator and an advisor a discriminator, he is probably the best discriminator a student could have. First, he uses all the researchers, mentors, and students with whom he has interacted as valuable training data. Second, during each optimization step, he not only tells me the difference between those scholars and myself, but also provides helpful suggestions on how to close the gap. Third, his guidance is always encouraging and constructive, preventing me from crashing due to gradient explosion. Perhaps this adversarial mentorship sounds a bit complicated. In the end, it is probably just some nearest neighbor mumbo jumbo. I hope I have become closer to one of his training points throughout the years.

As we all know, initialization matters in learning a non-convex function. For that, I would

like to thank Honghui Shi, Jianping Wang and Thomas Paine for bringing a sophomore to the wonderful world of computer vision and computer graphics. Without their fantastic work, I would not have been attracted to these fields in the first place. BE I have had the privilege of conducting and discussing my research with many UIUC faculty members, including my doctoral committee members—Prof. Rizwan Uddin, Zhi-Pei Liang, Shiva Abbaszadeh, and Angela Di Fulvio. It has also been a privilege to work with committee member Dr. Bogdan Georgescu of Siemens. Over the summers, I have had the good fortune to intern with many researchers at Siemens, including Dr. Georgescu, S. Kevin Zhou, Siqui Liu, Haofu Liao, and Zizhao Zhang. I would especially like to thank Dr. Georgescu, who has shared not only his view of vision and graphics, but also invaluable practical knowledge. His daily support helped push me through the ups and downs during my PhD, especially when experiments were not working out. I was extremely fortunate to be able to treat him as a second advisor.

I would like thank Becky Meline for her advice and help throughout my PhD years.

For my final exam scheduling and writing and endorsement, I would like to thank Prof. Yang Zhang and Prof Honghui Shi.

Finally, I am grateful to my parents for their love and support during this wonderful journey. The unbroken bonds between us made me the person I am today.

TABLE OF CONTENTS

LIST OF TABLES	x
LIST OF FIGURES	xi
LIST OF ABBREVIATIONS	xv
CHAPTER 1 INTRODUCTION	1
CHAPTER 2 LEARNING A SELF-INVERSE NETWORK FOR BIDIRECTIONAL MRI IMAGE SYNTHESIS	7
2.1 Introduction	7
2.2 Benefits of Learning A Self-inverse Network	9
2.3 Related Work	12
2.4 Method	15
2.5 Experimental Results	18
2.6 Model Sensitivity Analysis	21
2.7 Conclusion	23
CHAPTER 3 ONE-TO-ONE MAPPING FOR UNPAIRED IMAGE-TO-IMAGE TRANSLATION	29
3.1 Introduction	29
3.2 Literature Review	33
3.3 Unsupervised Learning of One-to-one Mappings between Domains	35
3.4 Self-inverse Learning for Unpaired Image-to-image Translation	37
3.5 Experiments	42
3.6 Conclusions	45
3.7 Acknowledgment	46
CHAPTER 4 TEXT EMBEDDING BANK MODULE FOR DETAILED IMAGE PARAGRAPH CAPTIONING	52
4.1 Introduction	52
4.2 Related Works	53
4.3 Approach	59
4.4 Implementation	59
4.5 Experiments	60
4.6 Conclusion	65

CHAPTER 5 FUTURE WORK: TOWARDS EXTRACTING AND LEARNING VECTOR SPACE REPRESENTATIONS OF WORDS FOR IMAGE CAPTIONING	68
5.1 Introduction	68
5.2 Problem Description	69
5.3 Data	69
5.4 Models and Algorithms	70
5.5 Experiments and Results	75
5.6 Conclusion	82
APPENDIX A DC-DENSEUNET: 2D-3D DENSELY COUPLED, DENSELY CON- NECTED UNET FOR AUTOMATIC LIVER LESION SEGMENTATION FROM CT VOLUMES	84
A.1 Introduction	84
A.2 Related Work	87
A.3 Method	88
A.4 Experiments and Results	95
A.5 Conclusions	98
REFERENCES	99

LIST OF TABLES

2.1	Quantitative performance of labels \leftrightarrow photo on cityscapes dataset.	11
2.2	Quantitative performance of map \leftrightarrow aerial on google maps.	13
2.3	Model sensitivity performance of labels \leftrightarrow photo on cityscapes.	15
2.4	Model sensitivity performance of aerial \leftrightarrow map on Maps dataset.	17
2.5	(a) Image synthesis performance and (b) model sensitive analysis on MRI T1 and T2 images from BraTs dataset [1]. Smaller L1 is better than larger. The difference between PSNR and SSIM increases with sensitivity. All the metrics are averaged on 10230 1-channel 2D images.	20
3.1	Results of Photo \leftrightarrow Label translation on the Cityscapes dataset.	42
3.2	Results of user study on the horse to zebra dataset.	43
3.3	Evaluation of cross-modal medical image synthesis on the BRATS database.	44
3.4	Results of user study on the summer to winter Yosemite dataset.	45
4.1	Our result compared with prior results on Stanford Visual Genome dataset	61
5.1	Model validation loss and BLEU scores on the validation dataset	77
5.2	Leaderboard of various methods on the online MS-COCO test server	77
A.1	DC-DenseUNet architecture	90
A.2	Segmentation results on the test dataset (from LiTS 2017 leaderaboard and publications)(Dice: %).	96
A.3	Segmentation results by ablation study of our methods on the test dataset.(Dice: %).	97

LIST OF FIGURES

2.1	Our self-inverse network learns a bijective mapping $f : x_i \leftrightarrow y_i$. Here we illustrate the concept using the CityScapes dataset [2] for bidirectional photo-to-label translation.	8
2.2	Comparison of our self-inverse network and other CNNs for image-to-image translation. The f and f^{-1} are the two generator networks for the tasks A and B , respectively. The D_Y and the D_X are the associated adversarial discriminators. (a) Pix2pix [3]: Two separate generator networks f and f^{-1} for the tasks A and B , respectively. (b) Cycle GAN [4]: Two jointly trained but different generator networks f and f^{-1} for the tasks A and B , respectively. (c) Self-inverse network: Only one generator network for both tasks.	9
2.3	Function space. Blue area: the whole function space; White area: the function space of a CNN; Purple area: the function space of f ; Green area: the function space of f^{-1} ; and Overlap area: the function space of $f = f^{-1}$	10
2.4	Illustrations of the self-inverse network using the U-Net architecture [5]. Each block represents the Convolution-BatchNorm-LeakyReLU layers in the encoder part and the Convolution-BatchNorm-ReLU layers in the decoder. Alternative training: In the training stage, for a batch of image pairs (x_i, y_i) , at the step j , the input and label are x_i and y_i , respectively, and at the step $j + 1$, the input and label are y_i and x_i , respectively.	12
2.5	Qualitative result on labels \leftrightarrow photo bidirectional image-to-image translation on cityscapes dataset. Upper: photo \rightarrow label. Bottom: label \rightarrow photo.	24
2.6	Qualitative result on Google maps. Upper: aerial \rightarrow map. Bottom: map \rightarrow aerial.	25
2.7	Model sensitivity performance of labels \leftrightarrow photo on cityscapes. Upper: photo \rightarrow labels. The input is generated by inputting the groundtruth to pix2pixB. Bottom: labels \rightarrow photo. The input is generated by inputting the groundtruth to pix2pixA.	26
2.8	Model sensitivity performance of aerial \leftrightarrow map on google maps. Upper: aerial \rightarrow map. The input is generated by inputting the groundtruth to pix2pixB. Bottom: map \rightarrow aerial. The input is generated by inputting the groundtruth to pix2pixA.	27

2.9	Examples of generated images. Column 1 depicts the original images for T_1 . Column 2 depicts the original images for T_2 . Generated T_2 images from T_1 with pix2pix and one2one models are in columns 3 and 4 respectively. Generated T_1 images from T_2 with pix2pix and one2one models are in columns 5 and 6 respectively. Generated T_2 images from column 5 with pix2pix and one2one models are in columns 7 and 8, respectively. Generated T_1 images from column 3 with pix2pix and one2one models are in columns 9 and 10, respectively. In columns 3-6, the score under each image is its PSNR and SSIM score compared with the original image. In column 7-10, the scores under each image are the PSNR and SSIM score differences between input x and $x + dx$ for both models. For example, to compare model sensitivity in the $T_1 \rightarrow T_2$ direction, x is column 1 and $x + dx$ is column 5. The model sensitivity for the pix2pix model is the score difference between columns 3 and 7. The model sensitivity for the one2one model is the score difference between columns 4 and 8.	28
3.1	Comparison of our one2one CycleGAN with the original CycleGAN [4] for the mapping between two domains X and Y. (a) Original CycleGAN model. It contains two separate mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$. (b) Our one2one CycleGAN. We propose to realize one-to-one mapping by learning ONLY one self-inverse function G for the mapping between two domains bidirectionally. It contains only one mapping function $G : X \leftrightarrow Y$	30

3.2 (a) The mapping routes of CycleGAN. The limitation of the CycleGAN model is that it allows biased and non-unique unpaired image translation. For the mapping route $x \rightarrow x'$, the mapping $G : x \rightarrow y'$ is a one-to-many mapping with the result that x can be mapped to infinity possible y' . Let us denote the unique target as y'_t and the actually mapped result as y'_k ; the mapping $F : y'_k \rightarrow x'$ is a many-to-one mapping. As a result, there is allowable bias between the target y'_t and the prediction y'_k . Similarly, for the mapping route $y \rightarrow y'$, the mapping $F : y \rightarrow x'$ is a one-to-many mapping with the result that y can be mapped to infinity possible x' . Let us denote the unique target as x'_t and the actually mapped result as x'_k ; the mapping $G : x'_k \rightarrow y'$ is a many-to-one mapping. As a result, there is allowable bias between the target x'_t and the prediction x'_k . (b) The mapping routes of one2one CycleGAN. The motivation of one2one CycleGAN is to realize unique and accurate unpaired image translation. The mapping function G is a self-inverse function with the one-to-one mapping property. For the mapping route $x \rightarrow x'$, the mapping $G : x \rightarrow y'$ is a one-to-one mapping with the result that x is only mapped to the unique target y'_t . The mapping $F : y'_t \rightarrow x'$ is also a one-to-one mapping. As a result, there is no bias between the target and the prediction. Similarly, for the mapping route $y \rightarrow y'$, the mapping $F : y \rightarrow x'$ is a one-to-one mapping with the result that y can only be mapped to the unique target x'_t . The mapping $G : x'_t \rightarrow y'$ is also a one-to-one mapping. As a result, there is no bias between the target and the prediction. 47

3.3 Visual comparison for horse \leftrightarrow zebra. 48

3.4 Visual comparison for summer \leftrightarrow winter on yosemite. 49

3.5 Visual comparison for apple \leftrightarrow orange. 50

3.6 Visual comparison for photo \leftrightarrow label on the Cityscapes. 51

3.7 Qualitative comparison for T1 \leftrightarrow T2 on BRATS datasets. 51

4.1	Integration of the paragraph vector framework as a TEB module to an existing deep learning-based image captioning model. There are three interconnected components divided into three dashed rectangular boxes. In the green box on the top left, the image encoder extracts visual features through a CNN model. In the yellow box on the bottom, an RNN-based language model decoder is used to generate paragraphs. Existing deep learning-based models only contain these two components. The red box on the top right box is the TEB module: In the training stage, for a image, paragraph pair, the varied-length paragraph is mapped to a fixed-length vector which is called TEB through the paragraph vector framework. The visual features from the image encoder are converted to the predicted TEB (called TEB') through several fully connected layers. The TEB' is supervised by the TEB through an L1 loss, which acts as global deep supervision to regularize the visual feature extraction for the image encoder. The visual features and TEB' are concatenated and fed into the RNN as input. The generated paragraph is supervised by the ground truth paragraph through a word-level loss. In the inference stage, the TEB is not available and the TEB' acts as the distributed memory to provide the semantic features of the whole paragraph to alleviate the long-term dependency problem for the language model.	66
4.2	Qualitative result comparison of paragraph outputs of our model (Diversity with TEB) and the baseline Diversity model [6]	67
5.1	(left) Successful hypothesis from the baseline model. (right) Incorrect hypothesis from the baseline model.	78
5.2	(left) Decent hypothesis from the GloVe model. (right) Incorrect hypothesis from the GloVe model.	78
5.3	Accurate captions by the BERT model.	79
5.4	Direct comparison of the three main models proposed.	81
5.5	Failed attempt to visualizing attention.	83
A.1	Example of contrast-enhanced CT scans showing the large variations of shape, size, and location of liver lesions. The red regions denote the liver and the green ones denote the tumors.	86
A.2	DC-DenseUNet pipeline	91
A.3	Examples of liver and tumor segmentation results of DC-DenseUNet from the test dataset. The red regions denote the liver and the green ones denote the tumors.	93

LIST OF ABBREVIATIONS

ML	Machine Learning
MTL	Multi-Task Learning
CT	Computed Tomography
3D	3-dimension

CHAPTER 1

INTRODUCTION

In our daily life, we consume many natural images. Similarly, the hospital 'consume' medical images, such as radiology and pathology images, for the diagnosis and treatment of many diseases, such as pneumonia and pneumothorax etc. The reading, interpretation and understanding of medical images are usually conducted by specialized medical professionals. For example, radiology images are read by radiologist. In this process of interpretation and understanding, they examine each area of body, determine whether each area was found to be normal, abnormal or potentially abnormal. Finally they write their findings to a medical report.

Due to the limited resources and manpower, there is a huge gap between the patient demand and the medical image diagnosis quality. Especially for the radiologist and pathologist who are working in the rural area the resources of healthcare is limited, computer aided diagnosis and writing medical imaging reports is demanding. For instances, to efficiently and correctly diagnose a chest X-ray image, these key skills are necessary: 1. the ability to examine the normal and abnormal of the thorax and the necessary physiology of chest diseases; 2. ability of interpreting the radiograph through a fixed pattern; 3. skills of evaluating the development over time; 4. experience in understanding the clinical history and records; 5. skills of comprehensive diagnosis by correlating with other diagnosis results such

as electrocardiogram, respiratory function tests and laboratory results etc.

On the other hand, for experienced radiologists and pathologists, examining abnormality correctly and efficiently is challenging and writing imaging reports are tedious and time-consuming. Especially, in countries such as India, China which have high population density, the number of radiologist per capita is very low. Hundreds of radiology images need to be read by a radiologist per day. It is hard to maintain a high diagnosis accuracy without computer aided tool. Besides, reading and typing the findings of every image into computers typically takes the radiologist about 10 minutes. This accounts for a large portion of the diagnosis process.

In sum, no matter whether the medical professionals are experienced or inexperienced, an automatic computer aided medical image interpretation and automatic medical report generation are needed.

I investigate the learning of mapping between domains for the specific problem of medical image segmentation. Image segmentation is the process of pixel-wisely labelling a digital image. We call the processed image a mask of the original image. So to map an image to its mask is an instance of mapping between domains. Here, the two domains are the image domain and its mask domain.

To explore more general problems of the mapping between domains, more tasks are explored. In Appendix A, image segmentation is treated as a special case of image-to-image translation. This work solves the anisotropic resolution problem for 3D image semantic segmentation. Chapters 2 and 3 address learning a self-inverse network to realize one-to-one mapping for both paired and unpaired image-to-image translation. Chapters 4 and 5 ad-

dress learning the mapping from the image domain to the language domain. Specifically, this mapping is done by learning a language representation to form the language domain.

- Appendix A: 3D Anisotropic Resolution Medical Image Segmentation

To address the anisotropic resolution issue in the 3D medical image, a novel coupled densely connected UNet(C-DenseUNet) is proposed. This model consists of a 2D DenseUNet for intra-slice feature extraction and a 3D counterpart for inter-slices feature extraction. These two DenseUNets are coupled by concatenating the encoding features of the 2D DenseUNet to the decoding features of its 3D counterpart. The following convolution fuses these 2D and 3D features in a learnable way. We designed the C-DenseUNet architecture and learning process in an end-to-end manner, where the intra-slice and inter-slices features are jointly optimized through this concatenation fusion layer. We evaluate our method on the dataset of the MICCAI 2017 Liver Tumor Segmentation (LiTS) Challenge. Our method achieved very competitive performance for liver and tumor segmentation even with a single model.

- Chapter 2: Learning a Self-Inverse Network for Bidirectional MRI Image Synthesis

The one-to-one mapping is necessary for MRI image synthesis as MRI images are unique to the patient. State-of-the-art approaches for image synthesis from domain X to domain Y learn a convolutional neural network that meticulously maps between the domains. A different network is typically implemented to map along the opposite direction, from Y to X . In this chapter, we explore the possibility of only wielding one network for bi-directional image synthesis. In other words, such an autonomous learning network implements a *self-inverse function*. A self-inverse network shares

several distinct advantages: only one network instead of two, better generalization and more restricted parameter space. Most importantly, a self-inverse function guarantees a *one-to-one mapping*, a property that cannot be guaranteed by earlier approaches that are not self-inverse. The experiments on MRI T1 and T2 images show that, compared with the baseline approaches that use two separate models for the image synthesis along two directions, our self-inverse network achieves better synthesis results in terms of standard metrics. Finally, our sensitivity analysis confirms the feasibility of learning a one-to-one mapping function for MRI image synthesis.

- Chapter 3: One-to-one Mapping for Unpaired Image-to-image Translation

Recently image-to-image translation has attracted significant interest, starting from the successful use of the generative adversarial network (GAN), to the introduction of cyclic constraint, to extensions to multiple domains. However, in existing approaches, there is no guarantee that the mapping between two image domains is unique or one-to-one. Here we propose a self-inverse network learning approach for unpaired image-to-image translation. Building on CycleGAN, we learn a self-inverse function by simply augmenting the training samples by swapping inputs and outputs during training and with separated cycle consistency loss for each mapping direction. The outcome of such learning is a proven one-to-one mapping function. Our extensive experiments on a variety of datasets, including cross-modal medical image synthesis, object transfiguration, and semantic labeling, consistently demonstrate clear improvement over the CycleGAN method both qualitatively and quantitatively. In particular, our proposed method reaches the state-of-the-art result on the cityscapes benchmark

dataset for the label-to-photo unpaired directional image translation.

- Chapter 4: Text Embedding Bank Module for Detailed Image Paragraph Captioning

Image paragraph captioning is the task of automatically generating multiple sentences for describing images through coherent text. Existing deep learning-based models for image captioning typically consist of an image encoder to extract visual features and a language model decoder, an architecture that has shown promising results in single high-level sentence generation. However, only the word-level guiding signal is available when the image encoder is optimized to extract visual features. The inconsistency between the parallel extraction of visual features and sequential text supervision limits its success when the generated text is long (more than 50 words). In this chapter, we propose a new module, called the Text Embedding Bank (TEB) module, to address this problem for image paragraph captioning. This module uses the paragraph vector model to learn fixed-length feature representations from a variable-length paragraph. We refer to the fixed-length feature as the TEB. This TEB module plays two roles to benefit paragraph captioning performance. First, it acts as a form of global and coherent deep supervision to regularize visual feature extraction in the image encoder. Second, it acts as a distributed memory to provide features of the whole paragraph to the language model, which alleviates the long-term dependency problem. Adding this module to two existing state-of-the-art methods achieves a new state-of-the-art result on the paragraph captioning Stanford Visual Genome dataset.

- Chapter 5: Learning Vector Space Representations of Words for Image Captioning

Image captioning is the task of automatically generating a sentence for describing images through coherent text. Existing deep learning-based models for image captioning typically consist of an image encoder to extract visual features and a language model decoder, an architecture that has shown promising results in single high-level sentence generation. However, only the word-level guiding signal is available when the image encoder is optimized to extract visual features. The inconsistency between the parallel extraction of visual features and sequential text supervision limits its success. In this chapter, we propose extracting and learning vector space representations of words for image captioning. This vector space representation of words acts as a form of global and coherent deep supervision to regularize visual feature extraction in the image encoder. Second, it acts as a distributed memory to provide features of the whole sentence to the language model, which alleviates the long-term dependency problem. By integrating the BERT embedding to the caption, we achieve a new state-of-the-art result on the MS COCO Image Captioning Challenge.

CHAPTER 2

LEARNING A SELF-INVERSE NETWORK FOR BIDIRECTIONAL MRI IMAGE SYNTHESIS

2.1 Introduction

Magnetic resonance imaging (MRI) is one of the widely used medical image modalities due to its non-invasiveness and its ability to clearly capture soft tissue structures using multiple acquisition sequences. However, its disadvantage lies in its long acquisition time and high cost. Therefore, there is a lack of large scale MRI image databases needed for learning-based image analysis. MRI image synthesis or image-to-image translation [7, 8] is able to fill such a gap by generating more images for training purpose. Also, a generated MRI image can be helpful to cross-sequence image registration, in which an image is first synthesized for the target sequence and then used for registration [9].

In language translation, if we treat the translation from one language A to another language B as a forward process f , then the translation from language B to A is its inverse problem f^{-1} . Similarly, in computer vision, there is a concept of image-to-image translation [3, 4, 10, 11] that converts an image to another one. In medical imaging, there are image reconstruction problems. Traditionally, each of these problems uses two different functions, one for the forward task f and the other for its inverse f^{-1} . In this chapter, our goal is to demonstrate that, for MRI image synthesis and other tasks, we are able to *learn the above*

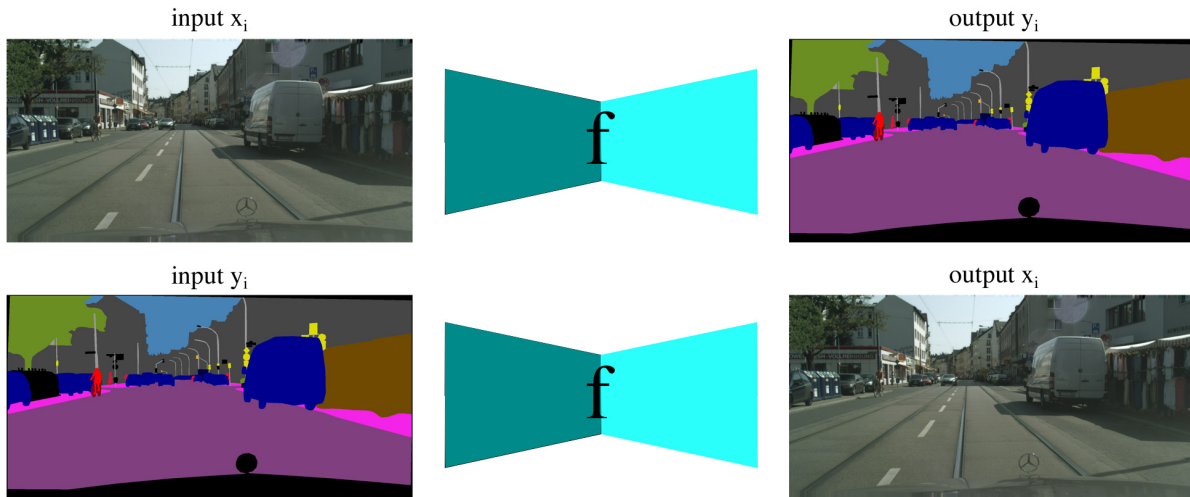


Figure 2.1: Our self-inverse network learns a bijective mapping $f : x_i \leftrightarrow y_i$. Here we illustrate the concept using the CityScapes dataset [2] for bidirectional photo-to-label translation.

two tasks simultaneously using only one function (see Figure 2.1), that is, $f = f^{-1}$.

Many problems found in computer visioning, computer graphics, image processing, and natural language processing stem from the inverse problem. In language translation, if we treat the translation from one language A to another language B as a forward process, then the translation from language B to A is its inverse problem. Similarly, in computer visioning, there is a concept of image-to-image translation [3, 4] that converts an image to another one. In medical imaging, there are image reconstruction problems. Traditionally, each of these problems uses two different functions, one for the forward task and the other one its inverse. In this chapter, our goal is to demonstrate that, in the context of image-to-image translation, only one function is able to learn the above two tasks simultaneously, as shown in Figure 2.1.

The community has explored the power of CNN in various tasks in computer visioning,

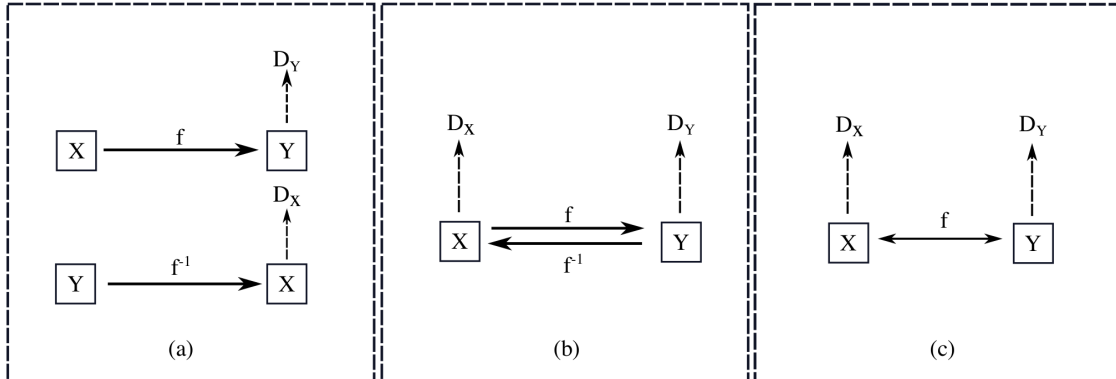


Figure 2.2: Comparison of our self-inverse network and other CNNs for image-to-image translation. The f and f^{-1} are the two generator networks for the tasks A and B , respectively. The D_Y and the D_X are the associated adversarial discriminators. (a) Pix2pix [3]: Two separate generator networks f and f^{-1} for the tasks A and B , respectively. (b) Cycle GAN [4]: Two jointly trained but different generator networks f and f^{-1} for the tasks A and B , respectively. (c) Self-inverse network: Only one generator network for both tasks.

as well as within several other fields. But so far, to the best of our knowledge, no one has explored the learning capability of a self-inverse function using CNN, and its potential use in applications. Our aim in this chapter is to bridge this gap. We refer to the mapping from a domain X to a domain Y as task A and the mapping from the domain Y to X as task B . Additionally, the proposed CNN that learns a self-inverse function is referred to as the self-inverse or one-to-one network.

2.2 Benefits of Learning A Self-inverse Network

There are several advantages in learning a self-inverse network in addition to the one-to-one mapping property.

(1) From the perspective of the application, only one self-inverse function can model both tasks A and B and it is a novel way of multi-task learning. As shown in Figure 2.4, the

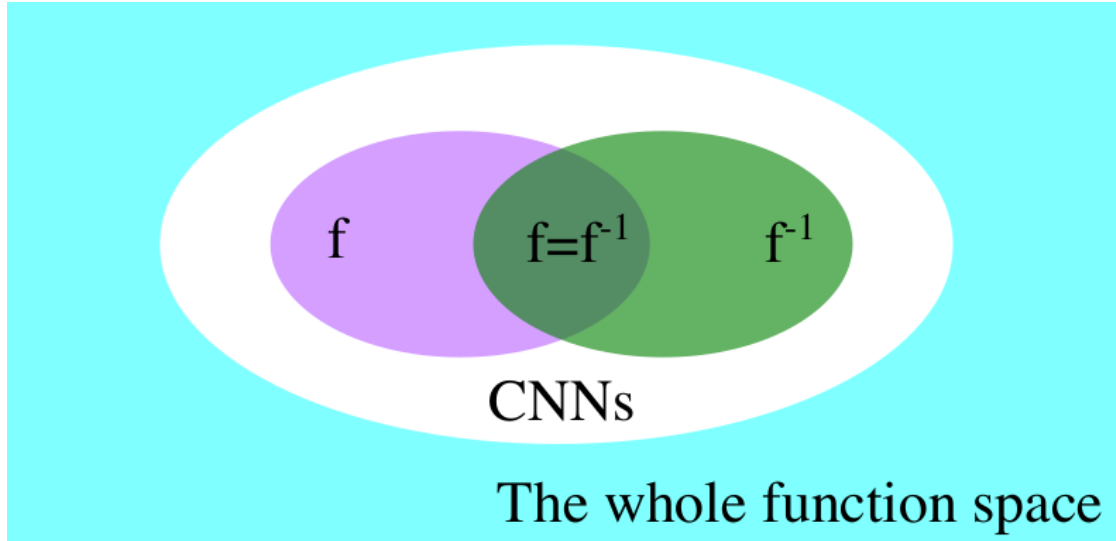


Figure 2.3: Function space. Blue area: the whole function space; White area: the function space of a CNN; Purple area: the function space of f ; Green area: the function space of f^{-1} ; and Overlap area: the function space of $f = f^{-1}$.

self-inverse network generates an output given an input, and vice versa, with only one CNN and without knowing the mapping direction. It is capable of doing both tasks within the same network, simultaneously. Rather than separately assigning two CNNs for tasks A and B , the self-inverse network halves the necessary parameters, assuming that the self-inverse network and the two CNNs share the same network architecture as shown in Figure 2.2.

(2) It automatically doubles the sample size, a great feature for any data-driven model, thus becoming less likely to over-fit the model. The self-inverse function f has the co-domain $Z = X \cup Y$. If the sample size of either domain X or Y is N , then the sample size for domain Z is $2N$. As a result, the sample sizes for both tasks A and B are doubled, making this a novel method for data augmentation to mitigate the over-fitting problem.

(3) It implicitly shrinks the target function space. As shown in Figure 2.3, the blue area is the whole function space, which is unlimited. Given a CNN with its architecture fixed,

Table 2.1: Quantitative performance of labels \leftrightarrow photo on cityscapes dataset.

Direction	Method	p. acc. \uparrow	c. acc. \uparrow	IOU \uparrow
photo \rightarrow label	pix2pix	0.80	0.35	0.29
photo \rightarrow label	one2one	0.83	0.35	0.29
label \rightarrow photo	pix2pix	0.73	0.25	0.19
label \rightarrow photo	one2one	0.74	0.25	0.20
labels \rightarrow photo	GT	0.80	0.26	0.21

its function space (Figure 2.3, white area) is enormous, with millions of parameters. When the CNN is trained for the task A , the target function space f is the purple area. When the CNN is trained for the task B , the target function space f^{-1} is the green area. When it is trained to learn a self-inverse function for both tasks A and B , the target function space is the overlapping area, which is a subset of the function space of f and f^{-1} . For a fixed neural network architecture, its function space is large enough to have the overlapping area in Figure 2.3. For a fixed data set, the trained model is a function within the blue area or the purple area for each direction, since the overlap area is always the subset of the blue or purple areas. If the network is trained as a self-inverse network, the trained model is a function within the overlapping area, which is always smaller than that of the network trained separately in each direction. A smaller function space means a smaller bias between the true function and the trained model, so the self-inverse network likely generalizes better. Another interpretation of this shrinking behavior is to regard the inverse f^{-1} as a regularization condition when learning the function f , and vice versa.

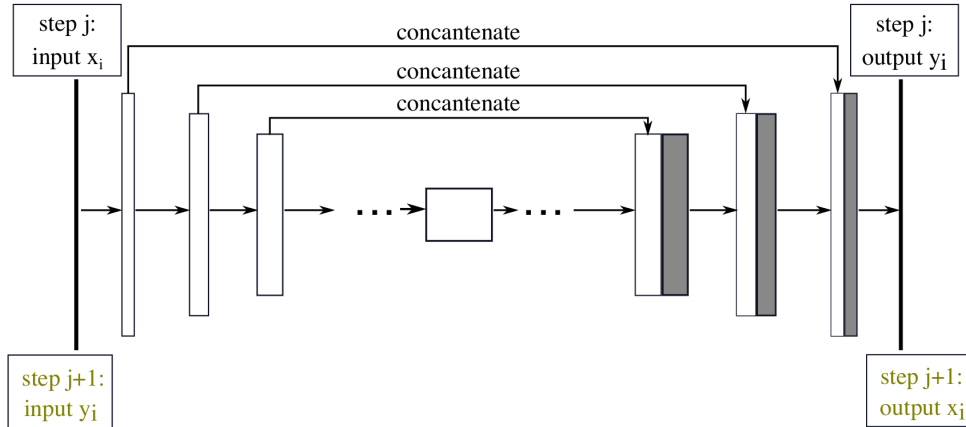


Figure 2.4: Illustrations of the self-inverse network using the U-Net architecture [5]. Each block represents the Convolution-BatchNorm-LeakyReLU layers in the encoder part and the Convolution-BatchNorm-ReLU layers in the decoder. Alternative training: In the training stage, for a batch of image pairs (x_i, y_i) , at the step j , the input and label are x_i and y_i , respectively, and at the step $j + 1$, the input and label are y_i and x_i , respectively.

2.3 Related Work

Inverse problem with neural networks The loss of information is a big problem that affects the performance of CNNs in various tasks. Several works such as [12, 13] show that essential information concerning the input image is lost as the network traverses to deeper layers in well-known ImageNet-based CNN classifiers. To recover and understand the loss of information, the above works use learned or hand-crafted methods prior to inverting the representation. An example of ‘compensating’ the lost information for performance improvement involves the segmentation task approach [14], which proposes the use of prior anatomical information from the latent space within a pre-trained decoder.

Building an invertible architecture is difficult due to the local inversion being ill-conditioned. Several works demonstrate that building an invertible CNN is a difficult challenge, hence not much progress has been made in solving it. Multiple works only allow invertible repre-

Table 2.2: Quantitative performance of map↔aerial on google maps.

Direction	Method	L1↓	PSNR↑	SSIM ↑
aerial→map	pix2pix	0.0696	19.36	0.505
aerial→map	one2one	0.0635	19.93	0.558
map→aerial	pix2pix	0.270	9.091	0.144
map→aerial	one2one	0.270	9.101	0.148

sentation learning under certain conditions. Parseval network [15] increases the robustness of learned representation with respect to adversarial attacks. In this work, the linear operator is bijective under the condition that the spectrum of the convolutional operator is constrained to norm 1 during learning. [16] introduces a signal recovery method conditioned on pooling representation to design invertible neural network layers. [17] makes the CNN architecture invertible by providing an explicit inverse. In this work, the reconstruction of the linear interpolations between natural image representations is achieved. This gives empirical evidence to the notion that it is possible to learn invertible representations that do not discard any information concerning their input on large-scale supervised problems. But the work from [17] cannot provide bi-directional mapping and is not self-invertible. Ardizzone et al. [18] prove the invertibility of neural network theoretically and verify experimentally for artificial data and real data in inverse problems using invertible neural networks. More specifically, Kingma [19] uses the invertible 1X1 convolution for the generative flow. Following this previous work, our self-inverse network realizes the inevitability between two domains by learning a self-inverse function.

Image-to-image translation The concept of image-to-image translation is broad, including image style transfer, translation between image and semantic labels, gray-scale to

color, edge-map to photograph, super-resolution [20] and many other types of image manipulations. It dates back to image analogies by [21], which employs a non-parametric texture model [22] from a single input-output training image pair. More recent approaches use a data set of input-output examples to learn a parametric translation function using CNN [23]. Our approach builds on the pix2pix framework of [3], which uses a conditional generative adversarial network [24] to learn a mapping from input to output images. CycleGAN [4] contributes to the unpaired image-to-image translation with a cycle consistency loss. In this framework, CycleGAN addresses exactly the same issue of learning a bijective mapping, albeit without the self-inverse property. CycleGAN can be seen as BiGAN [25] where the latent variable is like an image in the co-domain and the loss is augmented with an L1 loss. Similar ideas have been applied to various tasks such as generating photographs from sketches [26] or from attribute and semantic layouts[27]. Recently, [28] used multi-scale loss and conditional GAN to realize high-resolution image synthesis and semantic manipulation. One direction toward diversifying image translation is to allow many-to-many mapping, like augmented CycleGAN [29, 30, 31, 32, 10, 33]. The other direction towards accurate image translation is to restrict output image variance, like instance-level image translation [34]. Our method falls into the latter case and learns both tasks A and B with one generator network in a bidirectional way instead of using two generator networks (see Figure 2.2). Unlike [10], we encourage the invertibility of our model as a self-inverse function to realize bijection.

Neural style transfer Neural style transfer can be treated as a special category of image-to-image translation as well. [35] proposes to use image representation derived from CNN, optimized for object recognition, to make high-level image information explicit. [36] intro-

Table 2.3: Model sensitivity performance of labels \leftrightarrow photo on cityscapes.

Direction	Method	d(Class IOU) \uparrow
labels \rightarrow photo	pix2pix	0.0168
labels \rightarrow photo	one2one	0.0178
photo \rightarrow labels	pix2pix	0.0199
photo \rightarrow labels	one2one	0.0190

duced a cascade refinement network for photographic image synthesis. [37] highlights the power and flexibility of generative feed-forward models trained with complex and expressive loss functions for style transfer. [38] contributes the perceptual losses, which works very well.

2.4 Method

Our goal is to learn a self-inverse mapping function or bidirectional mapping function f for pairs (x_i, y_i) . This means $f : x_i \leftrightarrow y_i$. It also can be illustrated in this way: The function $f : x_i \rightarrow y_i$ and its inverse function $f^{-1} : y_i \rightarrow x_i$ satisfy $f = f^{-1}$, where samples $\{x_i\}_{i=1}^N \in X$ and $\{y_i\}_{i=1}^N \in Y$, the symbol ' \leftrightarrow ' means bijection, the symbol ' \rightarrow ' means one-directional mapping, and the symbol '=' means the two functions on both sides are exactly the same function.

Mathematically, it boils down to solving the following minimization problem:

$$\min_W \sum_{i=1}^N l_A(f_W(x_i), y_i) + l_B(x_i, f_W(y_i)) + \lambda r(W), \quad (2.1)$$

where W denotes the neural network parameters, l_A and l_B the loss function for tasks A and

B , respectively, and $r(W)$ is the regularizer. In this work, we use L_1 norm as the loss and GAN discriminator as the regularizer. The model pipeline is illustrated in Figure 2.2(c). It consists of two networks. The generator network f and the discriminator network D_x or D_y . Here D_x and D_y are the same network, while the D_x and D_y are two different networks for the baseline pix2pix model (see Figure 2.2(a)). The generator f is trained to translate the image as realistically as possible to fool the discriminator network D_x or D_y , which is trained to detect as well as possible the ‘fake’ examples generated by f .

Detailed network architecture. We adopt the architecture from [3] for our self-inverse network implementation. Let C_k denote a Convolution-BatchNorm-LeakyReLU layer with k filters in the encoder and Convolution-BatchNorm-ReLU layer with k filters in the decoder. All convolutions are 4×4 spatial filters applied with a stride 2. Convolutions in the encoder are down-sampled by a factor of 2. Convolutions in the decoder are up-sampled by a factor of 2.

The encoder-decoder architecture consists of an encoder, $C_{64} - C_{128} - C_{256} - C_{512} - C_{512} - C_{512} - C_{512} - C_{512}$, and an decoder, $C_{512} - C_{512} - C_{512} - C_{512} - C_{512} - C_{256} - C_{128} - C_{64}$. After the last layer in the decoder, a convolution is applied to map according to the number of output channels, which is 1, followed by a Tanh function. Following the convention, the C_{64} is not applied with batch-normalization. All LeakyReLUs in the encoder are with a slope of 0.2. For the U-Net skip connection, the skip connection is to concatenate feature maps from layer i to layer $n - i$, where i is the layer index and n is the total number of layers. Compared to the decoder above without skip connection, the number of feature maps doubles due to the use of a U-Net decoder, $C_{512} - C_{1024} - C_{1024} - C_{1024} - C_{1024} - C_{512} - C_{256} - C_{128}$. It

Table 2.4: Model sensitivity performance of aerial↔map on Maps dataset.

Direction	Method	dL1↓	dPSNR↑	dSSIM↑
aerial→map	pix2pix	.0007	0.87	0.029
aerial→map	one2one	.0008	0.89	0.029
map→aerial	pix2pix	0.0140	0.447	0.023
map→aerial	one2one	0.0144	0.458	0.024

is $C_{64} - C_{128} - C_{256} - C_{512}$. Following the C_{512} layer is a convolution layer to map the feature map channel number to 1. Then a sigmoid function follows the above layers to generate the output. Similar to the generator, the first convolution layer C_{64} is without batch normalization. All LeakyReLU are with a slope of 0.2.

Loss function. The objective of a conditional GAN [39] can be expressed as

$$\mathcal{L}_{cGAN}(G, D) = E_{x,y}[\log D(x, y)] + E_{x,z}[\log(1 - D(x, G(x, z)))] \quad (2.2)$$

We use L1 distance rather than L2 as L1 encourages less blurring:

$$\mathcal{L}_{L1}(G) = E_{x,y,z} [||y - G(x, z)||_1] \quad (2.3)$$

Our final objective is

$$(G^*, D^*) = \arg \min_{\mathbf{G}} \max_{\mathbf{D}} \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \quad (2.4)$$

With z , the net could learn a mapping from x to y in terms of any distribution instead of just a delta function.

Bi-directional Training To train a CNN as a self-inverse network, we randomly sample

a certain-sized batch of pairs (x_i, y_i) and (y_i, x_i) alternatively and iteratively. This is shown in Figure 2.4) The baseline is without alternative training, which means that we are training two separated generator networks for the tasks A and B , respectively (see Figure 2.4). For a fair comparison with the baseline, with the same data set, we use the same batch size and the same number of epochs. In other words, except for the alternative part, everything is the same as the baseline. We resize the 256×256 input images to 286×286 , add a random jitter, and then randomly crop them back to size 256×256 . All networks are trained from scratch. The weights are initialized from a Gaussian distribution with mean 0 and standard deviation of 0.02.

2.5 Experimental Results

The term ‘pix2pix’ refers to the result obtained by the model we retrained from scratch following exactly the same training details as in the pix2pix paper [3]. The term ‘one2one’ refers to our results by training the same networks as a self-inverse function. In all the tables, all of the results are averaged across the whole validation partition which follows the same dataset split in [3]. In Figure 2.5-2.8, the number below each image corresponds to the image above it individually.

We conducted the experiments using three paired image data sets:

Semantic label \leftrightarrow **photo**, trained on the Cityscapes dataset [2]. Our model is one2one and the baseline is pix2pix.

Table 2.1 and Figure 2.5 show the model performance comparison between one2one model and pix2pix model on bidirectional label and photo image translation. The evaluation met-

rics are pixel accuracy (p.acc.), class accuracy(c.acc.) and class IOU(IOU). In the direction photo \rightarrow labels, our one2one model performs better than pix2pix model by 3.75% in pixel accuracy. In the direction labels \rightarrow photo, the evaluation metric is "FCN score". Our one2one model increases the class IOU by 5.3% compared with the pix2pix model. Note that the FCN score for ground truth is 0.21. The FCN score of the one2one model is 0.20 which is very close to the score of the ground truth.

Map \leftrightarrow aerial photo, trained on data scraped from Google Maps [3].

Table 2.2 and Figure 2.6 show the model performance comparison between one2one model and pix2pix model on bidirectional aerial and map image translation.

In the direction aerial photo \rightarrow map image translation is a many-to-one mapping. As shown in Table 2.2 and the upper part of Figure 2.6, pix2pix produces a better result than one2one by 3%, 10.5%, 9,6% in PSNR, SSIM and L1, respectively.

In the direction map \rightarrow aerial photo, as shown in Table 2.2 and the bottom part of Figure 2.6, one2one model outperforms the pix2pix model by 3% in SSIM and 2% in PSNR.

MRI image synthesis on BRATS We conducted the experiments based on the BraTS 2018 dataset [1], which contains ample multi-institutional routine clinically-acquired pre-operative multimodal MRI scans of glioblastoma (GBM/HGG) and lower-grade glioma (LGG). There are 285 3D volumes for training and 66 3D volumes for testing. The T_1 and T_2 images are selected for our bi-directional image synthesis. All the 3D volumes are preprocessed to one channel image of size 256 x 256 x 1.

In all tables, all results are averaged across all splits as in [1].

As shown in Table 2.5(a), in the $T_1 \rightarrow T_2$ image synthesis direction, our one2one model

Table 2.5: (a) Image synthesis performance and (b) model sensitive analysis on MRI T1 and T2 images from BraTs dataset [1]. Smaller L1 is better than larger. The difference between PSNR and SSIM increases with sensitivity. All the metrics are averaged on 10230 1-channel 2D images.

Direction	Method	(a)	L1↓	PSNR↑	SSIM↑	(b)	d PSNR ↑	d SSIM ↑
$T_1 \rightarrow T_2$	pix2pix		0.042	26.53	0.871		2.17	0.018
$T_1 \rightarrow T_2$	one2one		0.039	29.23	0.875		3.01	0.020
$T_2 \rightarrow T_1$	pix2pix		0.051	27.78	0.872		4.51	0.034
$T_2 \rightarrow T_1$	one2one		0.048	30.99	0.876		4.93	0.036

outperforms the pix2pix model on PSNR by 13.6%. The qualitative result is shown in columns 3 and 4 in Figure 2.9. In the $T_2 \rightarrow T_1$ image synthesis direction, our one2one model outperforms the pix2pix model on PSNR by 11.6%. The qualitative result is shown in columns 5 and 6 in Figure 2.9.

2.5.1 Evaluation metrics

- Cityscapes data set[2].

For fair comparison with the baseline, which is pix2pix [3], we follow the same evaluation metric as that in the tpix2pix [3] paper. We use the released public evaluation code from the pix2pix GitHub repository https://github.com/phillipi/pix2pix/tree/master/scripts/eval_cit.

For the photo→labels direction, we use IOU as the evaluation metric. For the labels→photo direction, we use the "FCN score" [40, 23, 41, 42, 43].

- Map data scraped from Google Maps [3] and Brats [1]

To quantify the image quality distance between the generated image and the ground truth objectively and to have a metric to do the model sensitivity analysis, we use the

SSIM[44], PSNR[45], and L1 distance as the evaluation metric for both directions.

2.6 Model Sensitivity Analysis

To measure the model sensitivity, we add a perturbation dx to the input image x , then measure the change of the output, dy . In our experiment on the BraTs dataset shown in Figure 2.9, in the $T_1 \rightarrow T_2$ direction, the input image with perturbation $x + dx$ is the generated T_1 images from T_2 with the pix2pix model (see column 5 in Figure 2.9). In the $T_2 \rightarrow T_1$ direction, the input image with perturbation $x + dx$ is the generated T_2 images from T_1 with the pix2pix model (see column 3 in Figure 2.9).

In order to compare the performance of pix2pix and one2one on both tasks A and B , we need to train 3 models in total: pix2pix for task A (pix2pixA), pix2pix for task B (pix2pixB) and a one2one model for both tasks A and B (one2one). To compare the model sensitivity between pix2pixA and one2one for task A , we follow four steps.

1. For an image pair $(x_i, y_i)/(T_1, T_2)$, we pass y_i/T_2 to pix2pixB as input to generate $x_i + dx_i/T_1'(pix2pix)$, which adds a perturbation to x_i/T_1 .
2. We input x_i/T_1 to the pix2pixA and one2one models, obtaining the corresponding outputs $y_i'/T_2'(pix2pix)$ and $y_i'/T_2'(one2one)$, respectively.
3. We input $x_i + dx_i$ to the pix2pixA and one2one models obtaining the corresponding outputs $(y_i + dy_i)'/T_2''(pix2pix)$ and $(y_i + dy_i)'/T_2''(one2one)$, respectively.
4. For both models, we use a predefined evaluation metric E (for example PSNR and SSIM) to evaluate y_i' and $(y_i + dy_i)'$ and get the scores Ey_i' and $E(y_i + dy_i)'$, respec-

tively. So, the change of the output is measured by $d\|E\| = |E(y_i + dy_i)' - Ey_i'|$.

The model with a larger change of the output due to perturbation dx_i is more sensitive, and vice versa. Similarly, we can compare the model sensitivity between pix2pixB and one2one for task B by swapping the x_i and y_i in the above steps.

As shown in Table 2.5(b) on the $T_1 \rightarrow T_2$ image synthesis direction, our one2one model is more sensitive than pix2pix model, improving PSNR by 38.7%! The qualitative result is shown in columns 7 and 8 in Figure 2.9. In the $T_2 \rightarrow T_1$ image synthesis direction, our one2one model is more sensitive than pix2pix, improving PSNR by 9.3%. The qualitative results are shown in columns 9 and 10 in Figure 2.9.

For the cityscapes dataset, we use the mean class IOU to measure the change of output for the photo \rightarrow labels direction and "FCN score" to measure the change of output for the labels \rightarrow photo direction. In Table 2.3 and Figure 2.7, D(CLASS IOU) is the absolute value difference between the IOU score for the photo \rightarrow labels direction and FCN score for label \rightarrow photo direction between one2one and pix2pix.

For the Google Maps data set, we use the structural similarity index (SSIM), peak signal to noise ratio (PSNR) and L1 distance to measure the change of output from both directions. In Table 2.4 and Figure 2.8, the dL1, dPSNR and dSSIM are the absolute value of the difference between one2one and pix2pix.

For the cityscapes dataset, according to Table 2.3, one2one model is more sensitive than pix2pix by 6% in the label \rightarrow photo direction and by 5% in the photo \rightarrow label direction, and Figure 2.8 illustrates qualitative sensitivity analysis.

For the maps dataset, according Table 2.4, one2one model is more sensitive than pix2pix

by 2% in PSNR and 14% in L1 for the aerial \rightarrow map direction. The one2one model is more sensitive than pix2pix by 3% in L1, 2% in PSNR and 4.3% in SSIM in the map \rightarrow aerial direction. Figure 2.8 illustrates qualitative sensitivity analysis.

In summary, one2one model is more sensitive than pix2pix on all the three datasets.

2.7 Conclusion

We have presented an approach for learning one U-Net for both forward and inverse image-to-image translation. The experiment results and model sensitivity analysis results are consistent to verify the one-to-one mapping property of the self-inverse network.

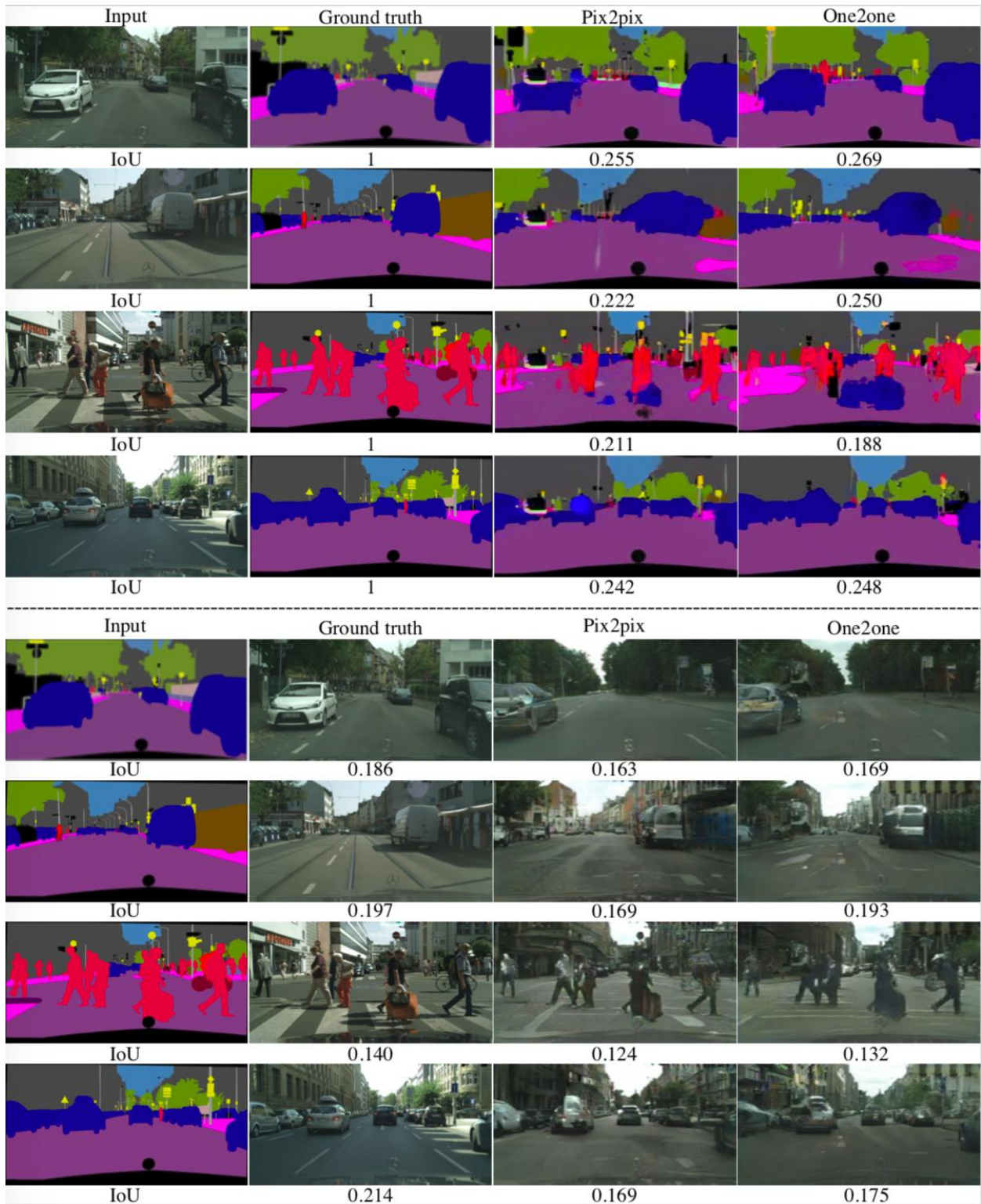


Figure 2.5: Qualitative result on labels \leftrightarrow photo bidirectional image-to-image translation on cityscapes dataset. Upper: photo \rightarrow label. Bottom: label \rightarrow photo.

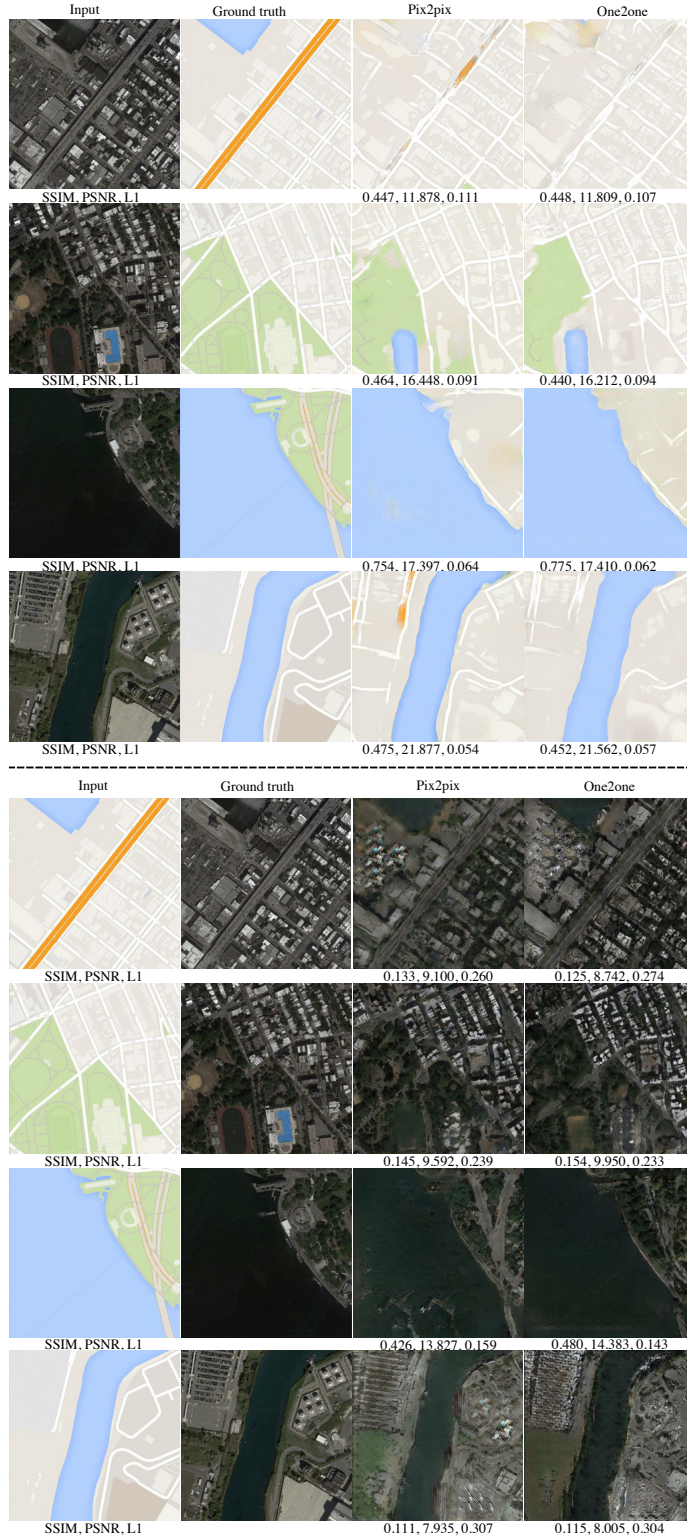


Figure 2.6: Qualitative result on Google maps. Upper: aerial \rightarrow map. Bottom: map \rightarrow aerial.



Figure 2.7: Model sensitivity performance of labels \leftrightarrow photo on cityscapes. Upper: photo \rightarrow labels. The input is generated by inputting the groundtruth to pix2pixB. Bottom: labels \rightarrow photo. The input is generated by inputting the groundtruth to pix2pixA.



Figure 2.8: Model sensitivity performance of aerial \leftrightarrow map on google maps. Upper: aerial \rightarrow map. The input is generated by inputting the groundtruth to pix2pixB. Bottom: map \rightarrow aerial. The input is generated by inputting the groundtruth to pix2pixA.

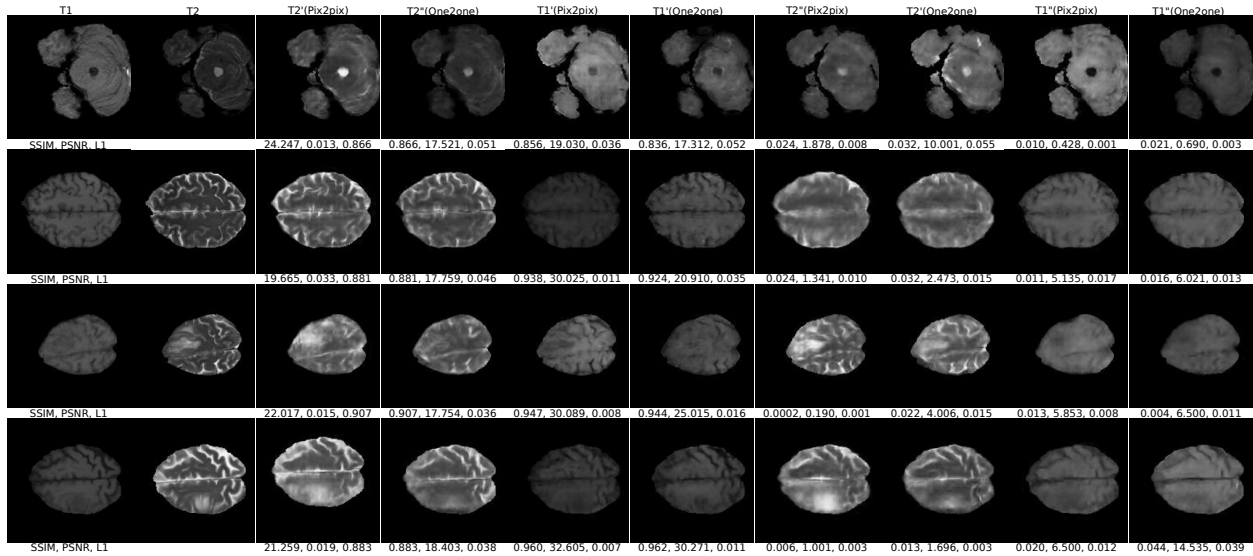


Figure 2.9: Examples of generated images. Column 1 depicts the original images for T_1 . Column 2 depicts the original images for T_2 . Generated T_2 images from T_1 with pix2pix and one2one models are in columns 3 and 4 respectively. Generated T_1 images from T_2 with pix2pix and one2one models are in columns 5 and 6 respectively. Generated T_2 images from column 5 with pix2pix and one2one models are in columns 7 and 8, respectively. Generated T_1 images from column 3 with pix2pix and one2one models are in columns 9 and 10, respectively. In columns 3-6, the score under each image is its PSNR and SSIM score compared with the original image. In column 7-10, the scores under each image are the PSNR and SSIM score differences between input x and $x + dx$ for both models. For example, to compare model sensitivity in the $T_1 \rightarrow T_2$ direction, x is column 1 and $x + dx$ is column 5. The model sensitivity for the pix2pix model is the score difference between columns 3 and 7. The model sensitivity for the one2one model is the score difference between columns 4 and 8.

CHAPTER 3

ONE-TO-ONE MAPPING FOR UNPAIRED IMAGE-TO-IMAGE TRANSLATION

3.1 Introduction

Image-to-image translation (or cross-domain image synthesis) learns a mapping function from an input image to an output image or vice versa. It can be grouped into two categories: supervised [46] vs unsupervised (or unpaired) [47].

The task of learning mappings between two domains from unpaired data has attracted a lot of attention, especially in the form of unpaired image-to-image translation [4, 10, 48, 30]. Thanks to the pioneering work of GAN[24] and cycleGAN [47], recent works [49, 50, 51, 52, 11, 32, 31, 29, 53, 34] have shown promising results for unpaired image-to-image translation. This task is very important because paired data are not available in many cases and the paired information is difficult or time-consuming to get. For example, in the medical field of cross-domain medical image segmentation [52, 51], with the brain CT image semantic label and without the brain MRI semantic label, the goal is to generate the semantic label for the brain MRI image. Impressive works [52, 51] like this cross-domain image segmentation task in the medical image application could be further improved if unpaired image translation can be unique and more accurate. In many cases the information source, such as a patient, is unique. For example, there is only a brain MRI image for a patient, but there should

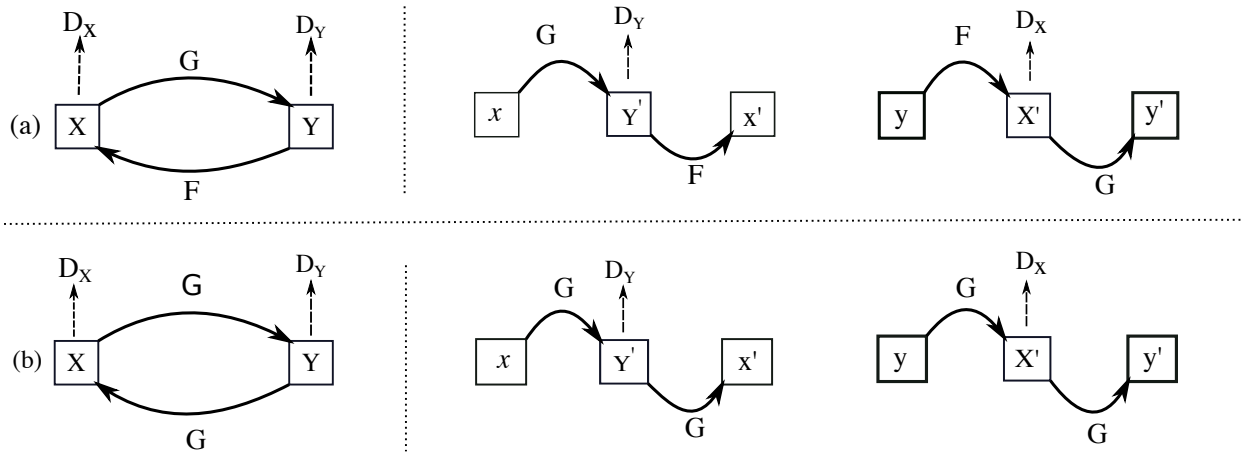


Figure 3.1: Comparison of our one2one CycleGAN with the original CycleGAN [4] for the mapping between two domains X and Y . (a) Original CycleGAN model. It contains two separate mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$. (b) Our one2one CycleGAN. We propose to realize one-to-one mapping by learning ONLY one self-inverse function G for the mapping between two domains bidirectionally. It contains only one mapping function $G : X \leftrightarrow Y$.

be a unique CT brain image for the same patient. This uniqueness requirement can be called one-to-one mapping of the brain CT and the brain MRI image from the same patient. If the unpaired image-to-image translation can achieve this one-to-one mapping, the cross-domain medical image segmentation performance can be further improved. However, existing methods cannot meet this requirement.

As mentioned above, the major limitation of existing methods for unpaired image-to-image translation, like CycleGAN, is that they cannot realize one-to-one mapping, which is necessary in many cases such as when the information source of the unpaired image is unique. Without the pairing information, CycleGAN using the distribution constraint allows many-to-many mappings. To reduce the space of possible mappings and improve in finding a more unique mapping, their models add an essential cycle-consistency constraint. The

cycle-consistency constraint enforces a stronger connection across domains by requiring the input image and the output image to be close. The output image is generated by first mapping from source domain to target domain, then mapping back to the source domain. But this only reduces the many-to-many mapping to many-to-one mapping or one-to-many across-domain mapping, which will be illustrated in detail in Sections 3.2 and 3.3.

With the success of image generation [24, 54] model generative adversarial networks (GANs) and unsupervised mapping methods like CycleGAN [47], and motivated by the recent works [55, 17] of exploring invertibility of convolutional neural networks (CNNs), we propose to learn a one-to-one mapping between domains from unpaired data to compensate for the limitation of the existing methods such as CycleGAN.

Specifically, we enforce the generator of the CycleGAN as a self-inverse function to realize a one-to-one mapping. So we call our proposed method One2one CycleGAN. When a function G is self-inverse, illustrated as

$$G = G^{-1}, \tag{3.1}$$

it guarantees a one-to-one mapping. We use the CycleGAN [47] as the baseline framework for image-to-image translation. To impose the self-inverse property, we implement *a simple idea* of augmenting the training samples by switching inputs and outputs during training. However, as we will demonstrate empirically, *this seemingly simple idea makes a genuinely big difference!*

The distinct feature of our self-inverse network is that it learns one network to perform both forward ($X \rightarrow Y$: from X to Y) and backward ($Y \rightarrow X$: from Y to X) translation tasks. It differs from the state-of-the-art approaches which typically learn two separate

networks, one for forwarding translation and the other for backward translation. As a result, it enjoys several benefits. First, it halves the necessary parameters, assuming that the self-inverse network and the two separate networks share the same network architecture. Second, it automatically doubles the sample size, a great feature for any data-driven model, thus becoming less likely to over-fit the model.

One key question arises: Is it feasible to learn such a self-inverse network for image-to-image translation? We cannot theoretically prove this existence; however, we experimentally demonstrate it. Intuitively, such an existence is related to the redundancy in the expressive power of the deep neural network. Even given a fixed network architecture, the function space for a network that translates an image from A to B is large enough; that is, there are many neural networks with different parameters capable of doing the same translation job. The same holds for the inversion network. Therefore, the overlap between these two spaces, in which the self-inverse network resides, does exist.

Our contributions are as follows: (i) We introduce the one2one CycleGAN model for learning one-to-one mappings across domains in an unsupervised way. (ii) We show that our model can learn mappings that generate a more accurate output for each input. (iii) We evaluate our method in extensive experiments on a variety of datasets, including cross-modal medical image synthesis, object transfiguration, and semantic labeling, consistently demonstrating clear improvement over the CycleGAN method both qualitatively and quantitatively. Especially, our proposed method reaches the state-of-the-art result on the Cityscapes benchmark dataset for the label-to-photo unpaired directional image translation.

3.2 Literature Review

Iosla et al. [46] presented the seminal work of image-to-image translation that offered a general-purpose solution, and Goodfellow et al. proposed to use the generative adversarial network (GAN) [24] for the first time in the literature. While paired data are assumed in [46], later Zhu et al. [47] proposed the CycleGAN approach for addressing the unpaired setting using the so-called cyclic constraints. There are many recent advances that use guidance information [49, 50], impose different constraints [56, 57, 58], or deal with multiple domains [10, 11, 32, 31], etc. In this chapter, we study unpaired image-to-image translation.

In addition to using the GAN that essentially enforces similarity in image distribution, other guidance information is used such as landmark points [49], contours [59], sketches [60], anatomical information [50], etc. In addition to cyclic constraint [47], other constraints like ternary discriminative function [56], optimal transport function [57], and smoothness over the sample graph [58] are used as well.

Also, extensions were proposed to deal with video inputs [61, 62], to synthesize images in high resolution [63], to seek for diversity [64] and to handle more than two image domains [10, 11, 32, 31]. Furthermore, there are methods that leverage attention mechanism [65, 66, 67] and mask guidance [68]. Finally, disentangling is a new emerging direction [32, 31].

In terms of works about inverse problems with neural networks, [17] makes the CNN architecture invertible by providing an explicit inverse. Ardizzone et al. [18] prove the invertibility theoretically. More specifically, Kingma [19] shows the benefit of an invertible 1×1 convolution.

Different from a one-to-one mapping function are one-to-many, many-to-one, and many-

to-many [29]¹ mapping functions. In [46], the well-studied scenarios of labels-to-scenes and edge-to-photo are more likely one-to-many mapping as it is possible that multiple photos (scenes) have the same edge (label) information. The colorization example is also one-to-many. From an information theory perspective, the entropy of the edge map (label) is low while that of the photo is high. When an image translation goes in an information-gaining direction, that is, from low-entropy to high-entropy, its mapping leans toward one-to-many. Similarly, if it goes in an information-losing direction, then its mapping leans toward many-to-one. If the information level of both domains is close (or information-similar), then the mapping is close to one-to-one. In [46], the examples of Monet-to-photo and summer-to-winter are closer to one-to-one mapping as the underlying contents of both images before and after translation are regarded the same but the styles are different, which does not change the image entropy significantly. For image-to-image translation, much work has been done to diversify the output [29, 30, 31, 32, 10, 33], while relatively little work has been done to make the output unique [34]. Our work goes in the latter direction.

Although there are many research works on image-to-image translation, the perspective of learning a one-to-one mapping network has not been fully investigated, with the exception of [57]. In [57], Lu et al. show that CycleGAN cannot theoretically guarantee the one-to-one mapping property and propose to use an optimal transport mechanism to mitigate this issue. However, like GAN, the optimal transport method also measures the similarity in image distribution; hence the one-to-one issue is not fully resolved. By contrast, our self-inverse learning comes with a guarantee that the learned network realizes a one-to-one

¹It is worth noting that recently there are quite a few works focusing on addressing image-to-image translation among many domains, also the so-called one-to-many.

mapping.

3.3 Unsupervised Learning of One-to-one Mappings between Domains

3.3.1 Problem setting

For any two domains X and Y with only unpaired elements available, we assume there exists a mapping, potentially a one-to-one mapping, between the elements of each domain. The goal is to make sure there is a unique target element in the target domain to match an element in the source domain. The objective is to recover this mapping. Since there are only unpaired samples available, this goal is realized by matching the distributions $p_d(x)$ and $p_d(y)$ of each domain. This can be treated as a conditional generating task. The true conditionals $p(x|y)$ and $p(y|x)$ are estimated from the true marginals. To be able to uncover this mapping, the elements in both domain X and domain Y are highly dependent.

3.3.2 CycleGAN model

As shown in Figure 3.1, the CycleGAN model [47] solves this problem by estimating these two conditionals with two separated mappings functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$. Both of the mapping functions are parameterized with identical neural networks and constrained by the following:

- Distribution matching: The distribution of each mapping output should match the distribution of the target domain. This constraint allows many-to-many mappings between the source domain X and the target domain Y and vice versa.

- Cycle-consistency: Each element is mapped from the source domain to the target domain, then mapped back to source domain. The output should be close to the input element. This constrains one-to-many mapping from source domain and many-to-one mapping from target domain to source domain.

3.3.3 Limitations of CycleGAN for one-to-one mapping

The main weakness of the CycleGAN model is that it cannot realize one-to-one mapping for accurate and unique unpaired image translation. Based on the above constraints and the illustration in Figure 3.2, the CycleGAN model cannot meet our goal of one-to-one mapping. Next, we show how to modify CycleGAN to meet the goal.

The distribution matching is implemented by GAN[24]. The two mapping functions G and F implemented by neural networks are trained to fool the discriminators D_Y and D_X respectively. The adversarial loss [24] for mapping function G is

$$\begin{aligned} \mathcal{L}_{GAN}(G, D_X, X, Y) = & E_{x \sim p_{data}(x)}[\log D_Y(x)] \\ & + E_{y \sim p_{data}(y)}[\log(1 - D_X(G(y)))]. \end{aligned} \quad (3.2)$$

The cycle consistency loss is

$$\begin{aligned} \mathcal{L}_{cyc}(G, F) = & E_{x \sim p_{data}(x)}[||F(G(x)) - x||_1] \\ & + E_{y \sim p_{data}(y)}[||G(F(y)) - y||_1]. \end{aligned} \quad (3.3)$$

The final objective for the mapping functions G and F is

$$\begin{aligned} \mathcal{L}(G, F, D_X, D_Y) &= \mathcal{L}_{GAN}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{GAN}(F, D_X, Y, X) + \lambda \mathcal{L}_{cyc}(G, F) \end{aligned} \quad (3.4)$$

and we aim to solve

$$(G^*, F^*) = \underset{\mathbf{G}, \mathbf{F}}{\operatorname{arg\,min}} \underset{\mathbf{D}_X, \mathbf{D}_Y}{\operatorname{max}} \mathcal{L}(G, F, D_X, D_Y). \quad (3.5)$$

3.4 Self-inverse Learning for Unpaired Image-to-image Translation

In the section, we first show the property that the self-inverse function guarantees one-to-one (one2one) mapping. Then we discuss how to train a self-inverse CycleGAN network for image-to-image translation

3.4.1 One-to-one property

In image-to-image translation, we define a forward function as $Y = f_{X \rightarrow B}(X)$ that maps an image X on domain A to another image Y on domain B and, similarly, an inverse function as $X = f_{B \rightarrow A}^{-1}(Y)$. When there is no confusion, we will skip the subscript (e.g., $A \rightarrow B$).

Property: If a function $Y = f(X)$ is self-inverse, that is $f = f^{-1}$, then the function f defines a one-to-one mapping, that is, $Y_1 = Y_2$ if and only if $X_1 = X_2$.

Proof:

[\Rightarrow] If $X_1 = X_2$, then $Y_1 = f(X_1) = f(X_2) = Y_2$.

[\Leftarrow] If $Y_1 = Y_2$, then $X_1 = f^{-1}(Y_1) = f^{-1}(Y_2) = X_2$ as long as the inverse function exists, which is the case for a self-inverse function as $f^{-1} = f$. #

3.4.2 One-to-one benefits

There are several advantages in learning a self-inverse network to have the one-to-one mapping property.

(1) From the perspective of the application, only one self-inverse function can model both tasks A and B , and it is a novel way of multi-task learning. As shown in Figure 3.1, the self-inverse network generates an output given an input, and vice versa, with only one CNN and without knowing the mapping direction. It is capable of doing both tasks within the same network simultaneously. In comparison to separately assigning two CNNs for tasks A and B , the self-inverse network halves the necessary parameters, assuming that the self-inverse network and the two CNNs share the same network architecture as shown in Figure 3.1.

(2) It automatically doubles the sample size, an important feature for any data-driven models, thus it is less likely to over-fit the model. The self-inverse function f has the co-domain $Z = X \cup Y$. If the sample size of either domain X or Y is N , then the sample size of domain Z is $2N$. As a result, the sample sizes of both tasks A and B is doubled, making this a novel method for data augmentation to mitigate the over-fitting problem.

(3) As shown in Figure 3.2, in the unpaired image-to-image translation setting, the goal is to minimize the distribution gap between the two domains. The state-of-the-art methods can realize this but cannot guarantee an ordered mapping or bijection between the two domains. This results in variations for the generated images.

(4) The one-to-one mapping is a strict constraint. Therefore, forcing a CNN model as a self-inverse function can shrink the target function space.

3.4.3 One-to-one CycleGAN

We are inspired by the basic formulation of CycleGAN [47]. In CycleGAN, there are two generators $Y = F(X)$ and $X = G(Y)$, two discriminators D_x and D_y , and one joint object function. In our one2one CycleGAN, we have one shared generator G and still two discriminators D_x and D_y . Instead of having a joint objective for the dual-mappings, our proposed method has two separate objective functions, one for each of two mapping directions.

Separated loss functions

Compared to CycleGAN that uses a joint loss for both image transfer directions, our method has two separate losses, one for each image transfer direction. For the mapping function $G : X \rightarrow Y$ and its discriminator D_Y , the adversarial loss is

$$\begin{aligned} \mathcal{L}_{GAN}(G, D_Y, X, Y) &= E_{y \sim p_{data}(y)}[\log D_Y(y)] \\ &+ E_{x \sim p_{data}(x)}[\log(1 - D_Y(G(x)))]. \end{aligned} \quad (3.6)$$

The cycle consistency loss is

$$\mathcal{L}_{cyc}^x(G) = E_{x \sim p_{data}(x)}[\|G(G(x)) - x\|_1]. \quad (3.7)$$

For the mapping function $G : Y \rightarrow X$ and its discriminator D_X , the adversarial loss is

$$\begin{aligned} \mathcal{L}_{GAN}(G, D_X, X, Y) &= E_{x \sim p_{data}(x)}[\log D_Y(x)] \\ &+ E_{y \sim p_{data}(y)}[\log(1 - D_x(G(y)))]. \end{aligned} \quad (3.8)$$

The cycle consistency loss is

$$\mathcal{L}_{cyc}^y(G) = E_{y \sim p_{data}(y)}[\|G(G(y)) - y\|_1]. \quad (3.9)$$

So, the final objective for the mapping function $X \rightarrow Y$ is

$$\mathcal{L}(G, D_Y) = \mathcal{L}_{GAN}(G, D_Y, X, Y) + \lambda_x \mathcal{L}_{cyc}^x(G), \quad (3.10)$$

and the minimax optimization solves

$$(G^*, D_Y^*) = \arg \min_{\mathbf{G}} \max_{\mathbf{D}_Y} \mathcal{L}(G, D_Y). \quad (3.11)$$

Similarly, the final objective for the mapping function $Y \rightarrow X$ is

$$\mathcal{L}(G, D_X) = \mathcal{L}_{GAN}(G, D_X, X, Y) + \lambda_y \mathcal{L}_{cyc}^y(G), \quad (3.12)$$

and the minimax optimization solves

$$(G^*, D_X^*) = \arg \min_{\mathbf{G}} \max_{\mathbf{D}_X} \mathcal{L}(G, D_X). \quad (3.13)$$

3.4.4 Self-inverse implementation

We apply the proposed method based on the framework of CycleGAN [47]. To have a fair comparison with CycleGAN, we adopt the architecture of (Johnson et al., 2016) as the generator and the PatchGAN [46] as the discriminator. The log likelihood objective in the original GAN is replaced with a least-squared loss [38] for more stable training. We resize the input images to 256×256 . The loss weights are set as $\lambda_x = \lambda_y = 10$. Following CycleGAN, we adopt the Adam optimizer [69] with a learning rate of 0.0002. Similarly, we use a pool size of 50. The learning rate is fixed for the first 100 epochs and linearly decays to zero over the next 100 epochs on Yosemite and apple2orange datasets. The learning rate is fixed for the first 4 epochs and linearly decays to zero over the next 3 epochs on the BRATS dataset. The learning rate is fixed for the first 90 epochs and linearly decays to zero over the next 30 epochs on the Cityscapes dataset.

3.4.5 Training details and optimization

In our experiments, we use a batch size of 1. At each iteration, we randomly sample a batch of pair (x_i, y_i) , where samples $\{x_i\}_{i=1}^N \in X$ and $\{y_i\}_{i=1}^M \in Y$. At any iteration j , we perform the following three steps:

- First, we feed x_i as the input and y_i as the target, then forward G and back-propagate G .
- Second, we feed y_i as the input and x_i as the target, then forward G and back-propagate G .

Table 3.1: Results of Photo \leftrightarrow Label translation on the Cityscapes dataset.

Method	Label \rightarrow Photo			Photo \rightarrow Label		
	Pixel Acc. \uparrow	Class Acc. \uparrow	Class IoU \uparrow	Pixel Acc. \uparrow	Class Acc. \uparrow	Class IoU \uparrow
CycleGAN	52.7	15.2	11.0	57.2	21.0	15.7
DiscoGAN	45.0	11.1	7.0	45.2	10.9	6.3
DistanceGAN	48.5	10.9	7.3	20.5	8.2	3.4
UNIT	48.5	12.9	7.9	56.0	20.5	14.3
One2one(ours)	58.2	18.9	14.3	52.7	18.1	13.0

- Finally, we back-propagate D_Y and D_X individually.

3.5 Experiments

In order to test the effect of the proposed method, we evaluate it on an array of applications: cross-modal medical image synthesis, object transfiguration, and style transfer. Also we compare against several unpaired image-to-image translation methods: CycleGAN [47], DiscoGAN [48], DistanceGAN [70], and UNIT [30]. We conduct a user study when the ground truth images are unknown and perform quantitative evaluation when the ground truth images are present.

3.5.1 Datasets and results

Object transfiguration. We test our method on the horse \leftrightarrow zebra task used in the CycleGAN paper [47] with 2401 training images (939 horses and 1177 zebras) and 260 test images (120 horses and 140 zebras). This task has no ground truth for generated images and hence no quantitative evaluation is feasible. So we provide the qualitative results obtained in a user study. In the user study, we ask a user to rate his/her preferred image out of three

Table 3.2: Results of user study on the horse to zebra dataset.

Direction	Metric	Cycle	Distance	One2one
horse2zebra	Prefer pct. \uparrow	25%	0	75%
zebra2horse	Prefer pct. \uparrow	23%	0	77%

randomly positioned images, one obtained from CycleGAN, one from DistanceGAN, and the other from one2one CycleGAN. Figure 3.4 shows examples of input and synthesized images and Table 3.1 summarizes the use study results.

Figure 3.4 shows that one2one CycleGAN likely generates better quality images in an unsupervised fashion, especially in terms of the quality of zebra synthesis from the horse (refer to the first four rows). Our method generated more real and complete zebra content. From Table 3.1, it is clear that our one2one CycleGAN is the most favorable with a 75% (77%) preference percentage for the horse2zebra (zebra2horse) mapping direction, and DistanceGAN is the least favorable.

We test our method on the apple \leftrightarrow orange task [47] with 2014 training images (995 apples and 1019 orange) and 514 test images (248 apples and 266 oranges). This task has no ground truth for generated images and hence no quantitative evaluation is feasible. Figure 3.5 shows examples of input and synthesized images. There are failure cases in rows 1, 2, and 4 from CycleGAN while our model generates normal images.

Cross-modal medical image synthesis. This task evaluates cross-modal medical image synthesis. The models are trained on the BRATS dataset [71] which contains paired MRI data to allow quantitative evaluation. It contains ample multi-institutional routine clinically-acquired pre-operative multi-modal MRI scans of glioblastoma (GBM/HGG) and lower-

Table 3.3: Evaluation of cross-modal medical image synthesis on the BRATS database.

Direction	Method	PSNR \uparrow	SSIM \uparrow
T1 \rightarrow T2	CycleGAN	20.79	0.85
T1 \rightarrow T2	One2one CycleGAN	22.03	0.86
T2 \rightarrow T1	CycleGAN	17.47	0.81
T2 \rightarrow T1	One2one CycleGAN	18.31	0.82

grade glioma (LGG). There are 285 3D volumes for training and 66 3D volumes for the test. The T_1 and T_2 images are selected for our bi-directional image synthesis. All the 3D volumes are preprocessed to one channel image of size 256 x 256 x 1. We use the Peak signal-to-noise ratio (PSNR) and Structural Similarity Index Measure (SSIM) to evaluate the quality of generated images.

As shown in Table 3.2, in the $T_1 \rightarrow T_2$ image synthesis direction, our one2one model outperforms the CycleGAN model on PSNR by 6.0%. The qualitative result is shown in columns 3 and 4 in Figure 3.7. In the $T_2 \rightarrow T_1$ image synthesis direction, our one2one model outperforms the CycleGAN model on PSNR by 5.0%. The qualitative result is shown in columns 7 and 8 in Figure 3.7.

Semantic labeling. We also test our method on the labels \leftrightarrow photos task using the Cityscapes dataset [2] under the unpaired setting as in the original CycleGAN paper. For quantitative evaluation, in line with previous work, for labels \rightarrow photos we adopt the ‘‘FCN score’’ [46], which evaluates how interpretable the generated photos are according to a semantic segmentation algorithm. For photos \leftarrow labels, we use the standard segmentation metrics, including per-pixel accuracy, per-class accuracy, and mean class Intersection-Over-Union (class IoU). The quantitative result is shown in Table 3.3. Our model reaches the

Table 3.4: Results of user study on the summer to winter Yosemite dataset.

Direction	Metric	Cycle	One2one
summer2winter	Prefer pct. \uparrow	34%	66%
winter2summer	Prefer pct. \uparrow	41%	59%

state-of-the-art on the label \rightarrow photo direction image synthesis under this unpaired setting. The pixel accuracy outperforms the second best result by 10.4 %. The class accuracy outperforms the second best result by 24.3 %. The class IoU outperforms the second best result by 30.0 %. In the photo \rightarrow label direction, our model achieves comparable results.

The qualitative result is shown in Figure 3.6. Compared with CycleGAN which is the second best result in the label \rightarrow photo direction, our model has clearly better visual results. In the photo \rightarrow label direction, our model also achieves a comparable or better result.

Style Transfer. We also test our method on the summer \leftrightarrow winter style transfer task using the Yosemite dataset under the unpaired setting as in the original CycleGAN paper. As shown in Figure 3.4 for the qualitative result, our method has a better visual result in both directions of style transfer. We also do a similar user study by providing the generated image from the test set by our model and the CycleGAN to users. The result is in Table 3.4. The user study results show that our model has a higher preference than CycleGAN.

3.6 Conclusions

We have presented an approach for enforcing the learning of a one-to-one mapping function for unpaired image-to-image translation. The idea is to take advantage of representative redundancy in deep networks and realize self-inverse learning. The implementation is as

simple as augmenting the training samples by switching inputs and outputs. However, this seemingly simple idea brings a genuinely big difference, which has been confirmed by our extensive experiments on multiple applications including cross-modal medical image synthesis, object transfiguration, style transfer, etc. Using the CycleGAN as the base framework, the CycleGAN model learned using the one-to-one training strategy, which is one network only, has consistently outperformed the baseline models, consisting of two networks, in terms of various qualitative and quantitative metrics. In the future, we plan to investigate the effect of applying the self-inverse learning to natural language translation.

3.7 Acknowledgment

The work was supported by grants from Siemens Healthineers (Siemens 082387).

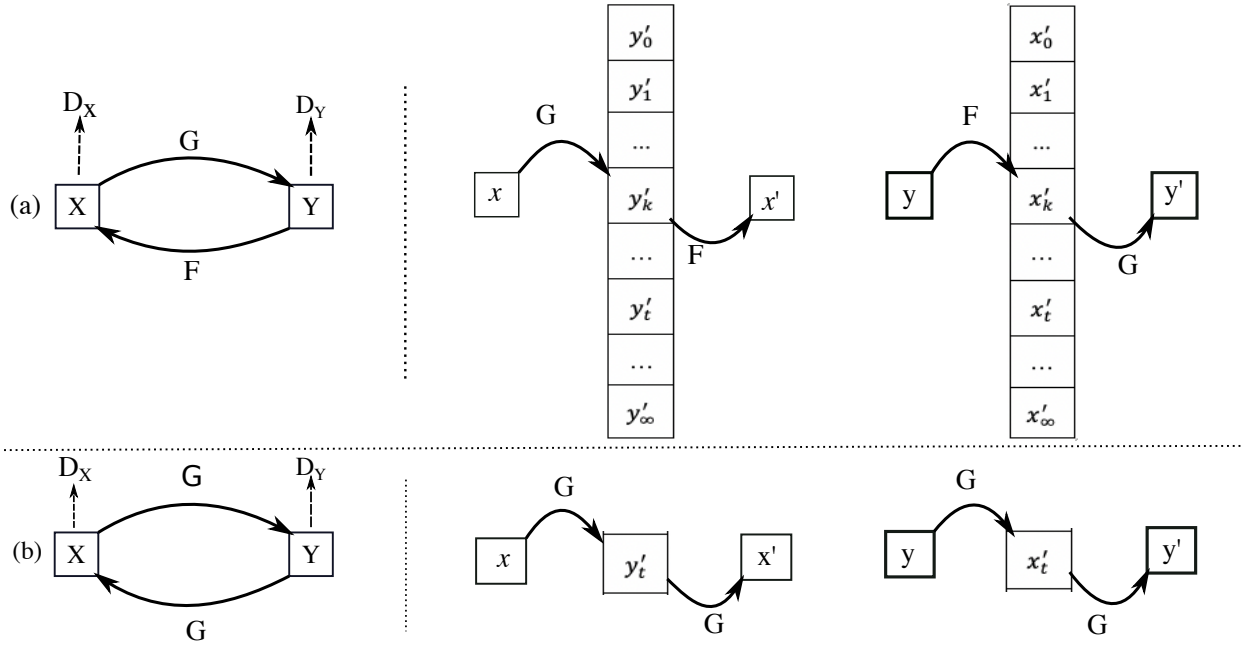


Figure 3.2: (a) The mapping routes of CycleGAN. The limitation of the CycleGAN model is that it allows biased and non-unique unpaired image translation. For the mapping route $x \rightarrow x'$, the mapping $G : x \rightarrow y'$ is a one-to-many mapping with the result that x can be mapped to infinity possible y' . Let us denote the unique target as y'_t and the actually mapped result as y'_k ; the mapping $F : y'_k \rightarrow x'$ is a many-to-one mapping. As a result, there is allowable bias between the target y'_t and the prediction y'_k . Similarly, for the mapping route $y \rightarrow y'$, the mapping $F : y \rightarrow x'$ is a one-to-many mapping with the result that y can be mapped to infinity possible x' . Let us denote the unique target as x'_t and the actually mapped result as x'_k ; the mapping $G : x'_k \rightarrow y'$ is a many-to-one mapping. As a result, there is allowable bias between the target x'_t and the prediction x'_k . (b) The mapping routes of one2one CycleGAN. The motivation of one2one CycleGAN is to realize unique and accurate unpaired image translation. The mapping function G is a self-inverse function with the one-to-one mapping property. For the mapping route $x \rightarrow x'$, the mapping $G : x \rightarrow y'$ is a one-to-one mapping with the result that x is only mapped to the unique target y'_t . The mapping $F : y'_t \rightarrow x'$ is also a one-to-one mapping. As a result, there is no bias between the target and the prediction. Similarly, for the mapping route $y \rightarrow y'$, the mapping $F : y \rightarrow x'$ is a one-to-one mapping with the result that y can only be mapped to the unique target x'_t . The mapping $G : x'_t \rightarrow y'$ is also a one-to-one mapping. As a result, there is no bias between the target and the prediction.

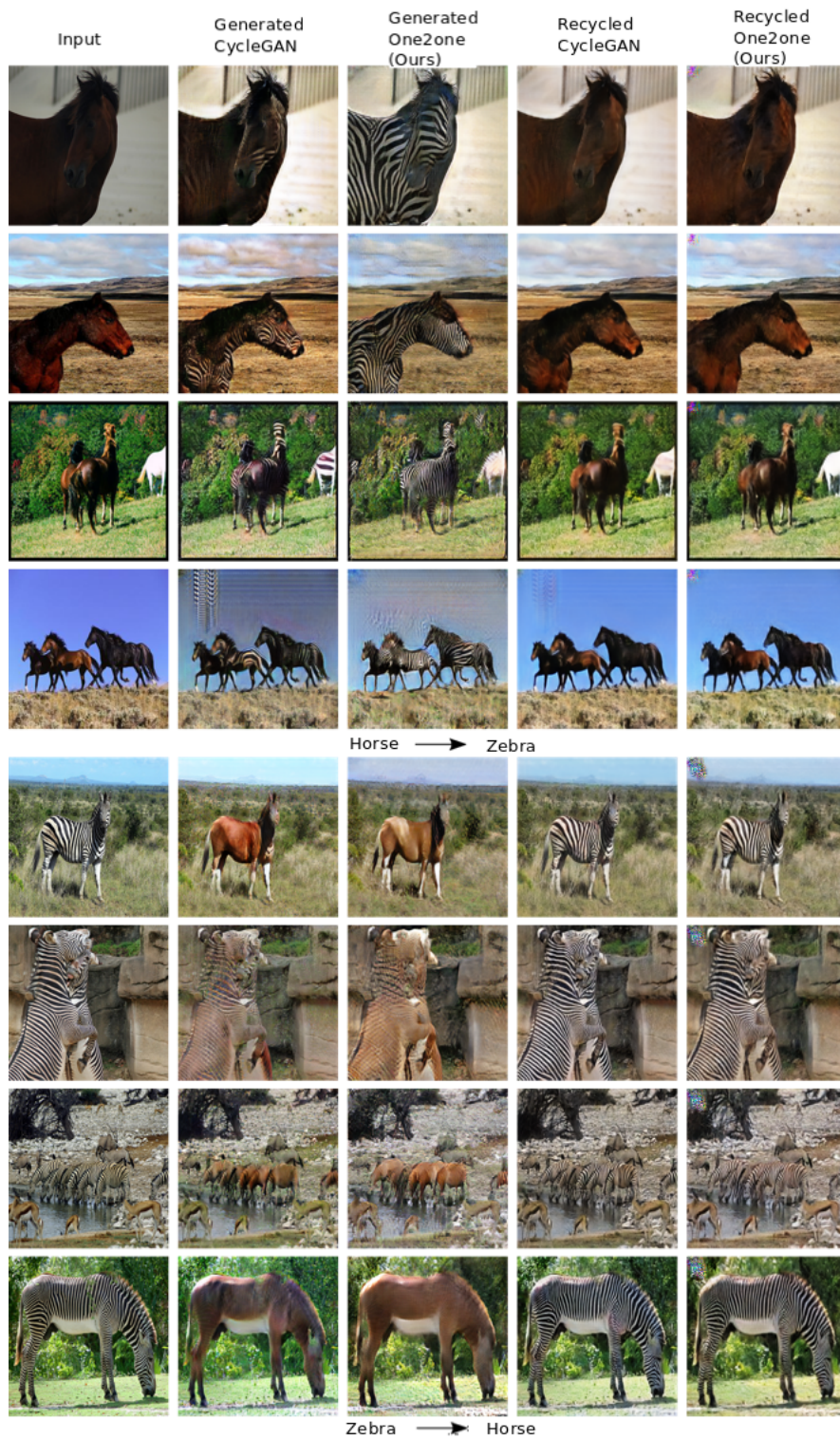


Figure 3.3: Visual comparison for horse↔zebra.

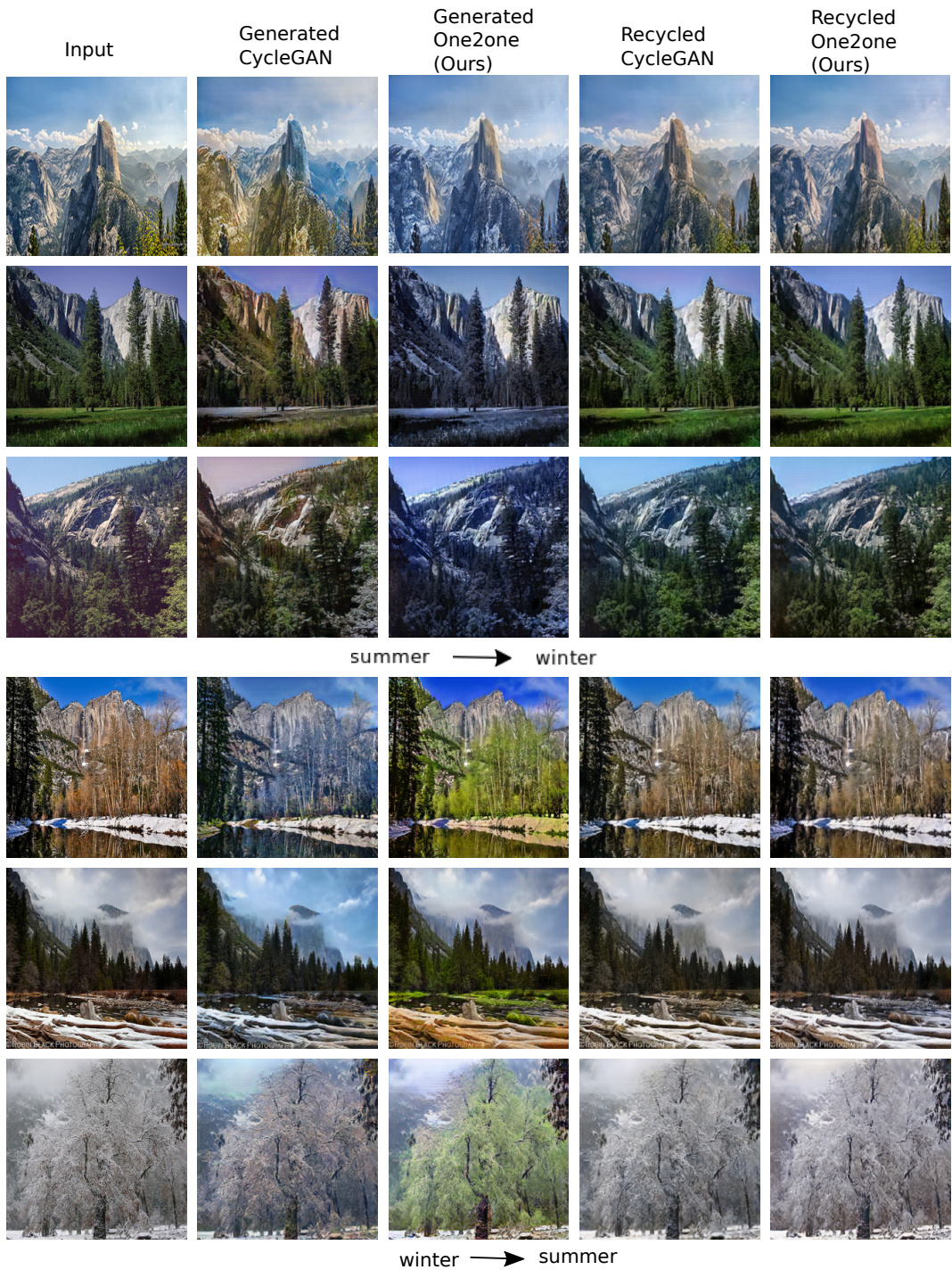


Figure 3.4: Visual comparison for summer↔winter on yosemite.



Figure 3.5: Visual comparison for apple↔orange.

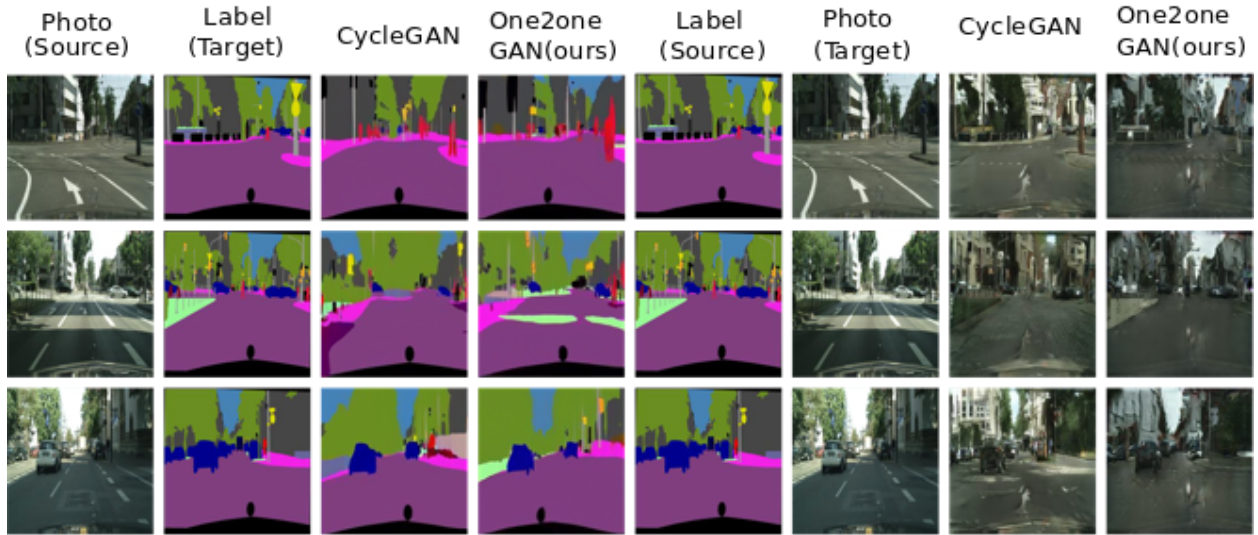


Figure 3.6: Visual comparison for photo \leftrightarrow label on the Cityscapes.

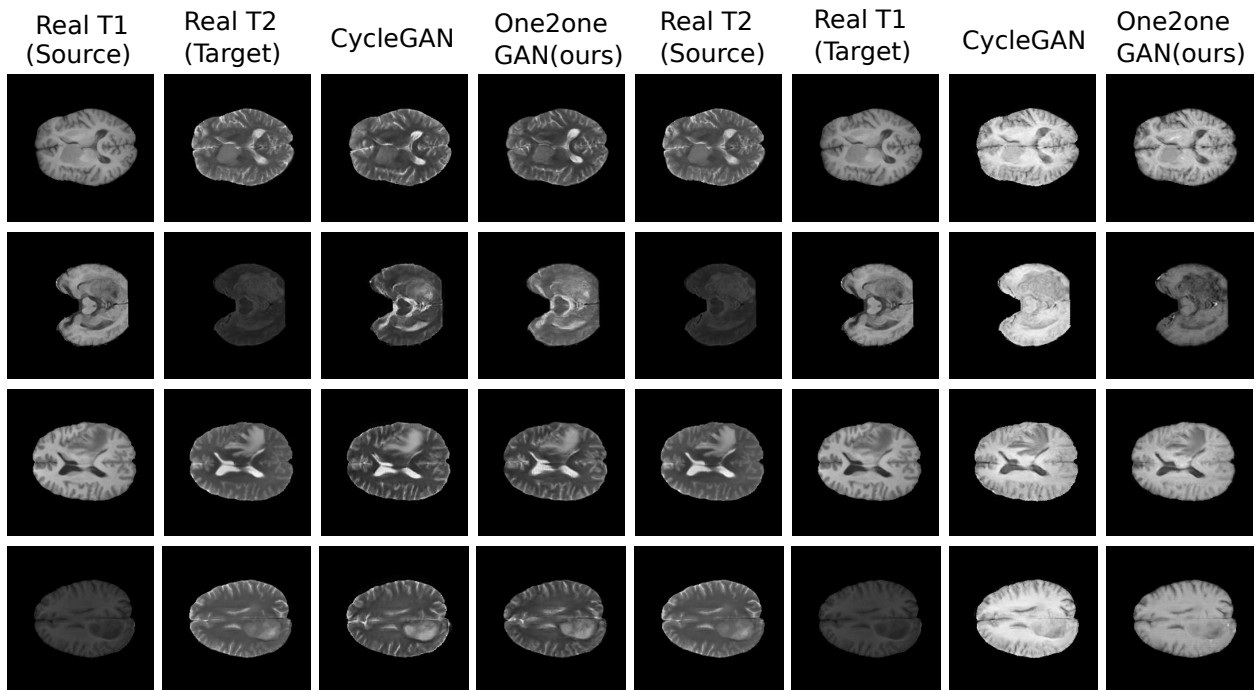


Figure 3.7: Qualitative comparison for T1 \leftrightarrow T2 on BRATS datasets.

CHAPTER 4

TEXT EMBEDDING BANK MODULE FOR DETAILED IMAGE PARAGRAPH CAPTIONING

4.1 Introduction

Image paragraph captioning is the task of generating logical and detailed descriptions by capturing subtle details of the visual input. Thanks to the advent of large datasets [72, 73, 74], many recent works [75, 76, 77] have shown promising results in generating a single high-level description for images and videos. A few works [78, 79, 6] have pushed the performance to new heights with the standard paragraph captioning dataset, the Stanford Visual Genome corpus, a dataset introduced by Krause et al. [78]. However, the coarse, scene-level captions that these models produce cannot meet the needs of real-world applications such as video retrieval, automatic medical report generation [80, 81, 82, 83], blind navigation, and automatic video subtitling, all of which require the model to capture fine-grained entities and produce a coherent description.

Relative to the performance of single-sentence caption generating models, the performance paragraph-length caption generating models are lower by a large margin. Paragraph captioning on images, and especially video, is a challenging task due to the requirement of both nuanced visual understanding and long-term language reasoning. Existing deep learning-based models typically consist of an image encoder to extract visual features in parallel with

a recurrent neural network (RNN) language model decoder to generate the sentences word by word sequentially. In the training stage, only a tiny scalar from the word level loss is available to optimize the image encoding training. This makes the visual feature extraction insufficiently detailed. To overcome this challenge, we propose the Text Embedding Bank (TEB) module. This module maps varied-length paragraphs to a fixed-length vector which we call TEB. Each unique vector in the TEB has similarity based on Euclidean distance and is indexed by the order of the word in the vocabulary. The TEB also has distributed memory. The TEB module, which holds the entire paragraph in a distributed memory model, can provide global supervision to better regularize the image encoder in the training stage. Additionally, RNNs are known to have a long-term dependency problem because of vanishing and exploding gradients which make it unable to meet long-term language reasoning. Since the TEB module has distributed memory and can provide order, it is better with long-term language reasoning.

We integrated our TEB module with the state-of-the-art methods on the only available paragraph captioning dataset, the Stanford Visual Genome corpus, and achieved a new state-of-the-art by a large margin.

4.2 Related Works

4.2.1 Image Captioning

This image-to-text problem is a classic in computer vision and natural language processing. The first work to use deep neural networks to solve this problem was the Neural Image Caption (NIC) in [84], which used a pre-trained convolutional neural network (CNN) as the

visual model and an RNN as the language model. The visual model extracted the visual features, which were then fed to the first time step of the RNN. The language model would take the visual features produced by the visual model at the first time step and predict the first word, before feeding the predicted word into the next time step and so on. At each time step, the difference between the predicted word and the ground truth word was optimized by a softmax with cross-entropy loss. Such a model could only predict one short simple sentence for each natural image. The performance of this one-sentence captioning task was improved in [85] by introducing an attention mechanism that focuses on related regions when generating a word per time step in the RNN model. In order to give a description for every object in an image, DenseCap[86] proposed a fully convolutional localization network which upgraded the region proposal network from Faster R-CNN[87] to localize the salient regions. The RNN model then took the corresponding visual features for each localized region to generate a sentence. However, simply joining all generated sentences together does not produce a coherent paragraph, which is a shortcoming of the DenseCap model.

Recently, the RNN/LSTM language model was replaced by a CNN in [88, 89] with comparable performance and the potential for parallel computing, which is a drawback of sequential models. In the inference process, however, this CNN model also needs to be computed sequentially. Since computation cost is a big issue for video captioning, [90] introduced a new method to find the most important frames, discarding redundant information and thus reducing computation cost.

4.2.2 Paragraph Captioning

Standard image captioning is the task of generating a single high-level sentence per image. Dense captioning is the task of generating a description for each salient object in an incoherent way. Paragraph captioning, however, overcomes the weaknesses of both of these tasks by generating fine-grained and coherent natural language descriptions, like a story. To meet the long-term language reasoning needs and the requirement of various topics in various sentences, a hierarchical recurrent neural network architecture [91, 92, 93, 78, 94] is widely used in paragraph captioning. For example, the model proposed in [93] generates multiple sentences for video captioning by capturing strong temporal dependencies. The model proposed in [78] uses a hierarchical recurrent network to build relationships between sentences. Regional features are passed to a sentence RNN to generate topic vectors with a halting distribution used to control the termination of a topic. The generated topic vectors are then consumed by a word RNN to generate sentences. In this way, this hierarchical RNN and DenseCap offer two ways of generating new topics, which is essential for multiple sentence generation. RTT-GAN [79] extends the hierarchical RNN by involving multi-level adversarial discriminators for paragraph generation. The paragraph generator is thus enforced to produce realistic paragraphs with a smooth logical transition between sentence topics. Furthermore, CapG-RevG [95] augments the hierarchical RNN with coherence vectors, global topic vectors, and a formulation of Variational Auto-Encoders [96] to further model the inherent ambiguity of associating paragraphs with images. CAE-LSTM [97] Convolutional Auto-Encoding (CAE) purely employs a convolutional and deconvolutional auto-encoding framework for topic modeling on the region-level features of an image. The IU Chest X-ray

dataset is used for automatic report generation on this unstructured report [94] by using co-attention and the hierarchical LSTM. The Diversity model [6] improves sentence diversity by introducing a repetitive penalty in the sequence-level training. However, all of these methods suffer from the fact that only a tiny partial scalar from the word-level loss can be used as guiding information to optimize the image encoding in training. Our TEB module can overcome this shortcoming and provides an alternative for the hierarchical recurrent neural network architecture. With our TEB module, a one-level recurrent neural network is enough to generate multiple sentences with a diverse range of topics.

4.2.3 Long-term dependency

GANs have been shown to improve real text generation in [98]. SeqGAN [99] was proposed to deal with the sequential and discrete property of text for text generation. LeakGAN [100] solves the sparse signal from the generator problem by leaking features from the generator to the discriminator for long sentence generation. MaskGan [101] introduced a way to fill in the blank using a GAN. Similarly, [102] uses long-term feature banks for detailed video understanding.

The proposed TEB module improves paragraph captioning by describing the rich content of a given image. Figure 4.1 shows an example of how the TEB module can be integrated with an existing image captioning pipeline.

4.2.4 Learning Vector Representation of words

The paragraph vector is based on word vectors, which are based on the idea of using a distributed vector representation of words. The basic idea is to predict a word given the other words in a context. The framework is shown in Figure 4.2.

In this framework, each word is mapped to a unique vector which is a column of a matrix. The column is indexed by the order of the word in the vocabulary. The features to predict the next word are the sum or concatenation of the vectors.

To express this in a mathematical equation, let $v_1, v_2, v_3, \dots, v_T$ represent the vectors of a sequence of training words. The objective function of the framework is to maximize the average log probability

$$\frac{1}{T} \sum_{t=k}^{T-k} \log p(t|_{t-k, \dots, t+k}) \quad (4.1)$$

Typically, a multi-class classifier such as softmax is used for the prediction task. So, we have

$$p(t|_{t-k, \dots, t+k}) = \frac{e^{y_{wt}}}{\sum_i e^{y_i}} \quad (4.2)$$

where y_i is the un-normalized log-probability for each output word i , which is computed as

$$y = b + Uh(t_{-k}, \dots, t_{+k};) \quad (4.3)$$

This framework is implemented in a neural network and trained using stochastic gradient descent through back-propagation [103]. This type of model is the well known neural language model [104].

Compared to existing image captioning models, which only use recurrent neural networks, after training converges, this framework can map words with similar meaning to a similar position in the vector space. For example, "wind" and "beautiful" are far away from each other in the vector space, while "beautiful" and "pretty" are closer. Additionally, the distance between each unique word vector also carries meaning. This means that it can be used for analogy questions answering in a simple vector algebra manipulation: "waiter" - "man" + "woman" = "waitress". This makes it easy to learn a linear matrix, such as a fully connected layer, to translate between visual features and these word vectors.

4.2.5 Paragraph Vector: A distributed memory model

Inspired by the word vector framework which can capture the semantics as a result of a prediction task, the paragraph vector also contributes to the prediction of the next word. In this paragraph vector framework (see Figure 4.3), similarly to the word vector framework, each word is still mapped to a unique vector which is a column of a matrix W , while each paragraph is mapped to a unique vector which is a column of a matrix P . Then, both the word vector and paragraph vector are fused (either summed or concatenated) as features to predict the next word. We use concatenation in our implementation.

The paragraph vector can be treated as a super word (or the topic of the paragraph) which acts as memory of the missing information from the current context. Hence, this framework is known as a distributed memory model. This property can compensate for the recurrent neural network's lack of generating logical connections between sentences in paragraph generation.

4.3 Approach

4.3.1 Integration of the paragraph vector as a TEB module for Image Paragraph Captioning

The integration of the paragraph vector as a TEB module for image paragraph captioning is illustrated in Figure 4.1. To show the generalizability of this TEB module, we integrated it with a typical deep learning-based image captioning model which consists of an image encoder (in the green box) and a language model decoder (in the yellow box). The TEB model can be integrated into any model with an image encoder language decoder structure by adding the module and concatenating the TEB’ semantic features to the existing visual features extracted by the image encoder. In this paper, two models are used which will be detailed in the next section.

4.4 Implementation

4.4.1 TEB module

For the paragraph vector framework[105], we adapted an implementation of [106]. The hyper-parameters are as follows: The vector size (TEB size) is 512, the sliding window size is 50, the sampling threshold is $1e - 5$, the negative size is 5. The paragraph vector model is trained for 1000 epochs before performing the inference to generate the TEB. Regardless of the dimensionality of the visual features from the image encoder, the visual features are converted to the same dimension of the TEB by several fully connected layers.

4.4.2 Integrating TEB on Diversity model [6]

As shown in Figure 4.1, we integrate our TEB module with the Diversity model [6] which is the current state-of-the-art model on the Stanford Visual Genome dataset. We used the model architecture and the entire training procedure from the Diversity model [6], except for the TEB module, for a fair comparison. This model uses the Bottom-Up and Top-Down model [107] as its backbone; self-critical sequence training (SCST) and a repetition penalty are also used.

4.4.3 Integrating TEB on Transformer model

We also integrate the TEB module into a transformer model. The transformer model is adapted from the Bottom-Up and Top-Down model [107] with the following modification: The LSTM-based language model is replaced by the transformer model [108]. We used both cross-entropy and SCST training, without the repetition penalty, and beam search instead of a greedy search.

4.5 Experiments

4.5.1 Datasets and Experiment Settings

Dataset. We conducted the experiments and evaluated our TEB module on the Stanford Visual Genome image paragraph dataset [78], a benchmark in the field of image paragraph captioning. The dataset contains 19,551 images and there is one human-annotated paragraph per image. On average, each paragraph has 67.5 words and each sentence consists of 11.91

Table 4.1: Our result compared with prior results on Stanford Visual Genome dataset

Methods	METEOR	CIDEr	BLEU-1	BLEU-2	BLUE-3	BLEU-4
Image-Flat [109]	12.82	11.06	34.04	19.95	12.20	7.71
Regions-Hierarchical [78]	15.95	13.52	41.90	24.11	14.23	8.69
RTT-GAN [79]	17.12	16.87	41.99	24.86	14.89	9.03
RTT-GAN(Plus) [79]	18.39	20.36	42.06	25.35	14.92	9.21
CapG-RevG [95]	18.62	20.93	42.38	25.52	15.15	9.43
CAE-LSTM [97]	18.82	25.15	-	-	-	9.67
Diversity [6]	17.86	30.63	43.54	27.44	17.33	10.58
Ours (Transformer)	15.45	23.38	41.49	23.38	11.96	6.00
Ours (Transformer + TEB)	15.88	24.84	41.86	24.64	13.97	6.40
Ours (Diversity + TEB)	18.93	32.53	45.24	28.44	17.93	10.98
Human	19.22	28.55	42.88	25.68	15.55	9.66

words. In our experiments, we follow the widely used train-val-test split in [78], taking 14,575 images for training, 2,487 for validation and 2,489 for testing.

Compared Methods. We compare the proposed method with the following state-of-the-art methods: (1) **Image-Flat**[109] is a standard image captioning model which directly decodes an image into a paragraph word by word, via a single LSTM. (2) **Regions-Hierarchical** [78] adopts a hierarchical LSTM to generate a paragraph, sentence by sentence. (3) **RTT-GAN** [79] integrates sentence attention and word attention into the hierarchical LSTM, coupled with an adversarial training strategy. (4) **CapG-RevG** [95] leverages coherence vectors and global topic vectors to generate coherent paragraphs and maintains the diversity of the paragraphs by a variational auto-encoder formulation. (5) **CAE-LSTM** [97] purely employs a convolutional and deconvolutional auto-encoding framework for topic modeling on the region-level features of an image. (6) **Diversity** [6] uses integrated penalty on trigram repetition to produce much more diverse paragraphs. (7) **Ours(TEB)** is the method in this dissertation. We add the TEB module on baseline methods for ablation

studies. There are three models: The "Diversity + TEB" model is the Diversity model [6] with SCST training [110], repetition penalty and TEB module. The "Transformer" model is the Bottom-UP and Top-Down model [107] with the LSTM replaced by the Transformer [108]. The "Transformer + TEB" is the "Transformer" model with our TEB module.

Evaluation Metrics. We adapted the most used metrics: METEOR [111], CIDEr [112], and BLEU1-4 [113]. We used the code released from the Microsoft COCO Evaluation server [114] to calculate the metric score.

4.5.2 Performance Comparison and Analysis

Quantitative Analysis. The performances of different models on the Stanford Visual Genome dataset are shown in Table 4.1. Overall, the results across three evaluation metrics consistently indicate that our proposed TEB module added to the diversity model achieves better performance than other state-of-the-art techniques including non-attention models (Image-Flat, Regions-Hierarchical, and CapG-RevG, CAE-LSTM, Diversity) and attention-based approaches (LSTM-ATT and RTT-GAN). Specifically, the CIDEr and BLEU-4 scores of our Diversity + TEB can achieve 7.57 % and 29.%, making a 6.2% and 3.8% relative improvement over the next best model (Diversity), respectively. As expected, by additionally modeling topics/gists in an image via an LSTM-based architecture, the Regions-Hierarchical model exhibits better performance than Image-Flat, which ignores inter-sentence dependency. Moreover, LSTM-ATT leads to a performance boost over Regions-Hierarchical, which directly encodes an image as a global representation by performing mean pooling over all region-level features. The results indicate that the advantage of region-level attention mech-

anisms in the two-level LSTM networks by learning to focus on the image regions are most indicative to infer the next word. More specifically, RTT-GAN and CapG-RevG do this by modeling reality and diversity of paragraphs with Generative Adversarial Networks and Variational Auto-Encoders, outperforming LSTM-ATT. However, the performance of both RTT-GAN and CapG-RevG is not as good as our CAE-LSTM, which exploits the inherent structure among all image regions for topic modeling in a convolutional and deconvolutional auto-encoding framework.

Qualitative Analysis. Figure 4.2 shows several paragraph examples generated by Diversity+TEB, Diversity and one human-annotated Ground Truth (GT) paragraph. From these exemplar results, it is easy to see that all of these paragraph generation models can produce somewhat relevant paragraphs, while our Diversity+TEB model generates the most coherent and detailed paragraphs with the aid of the TEB module which provides global supervision and remembers the whole text embedding. Compared to the Diversity model, which fails to capture part of the objects and lacks detail in its description of the captured objects, our Diversity+TEB model can capture many more objects in much more detail. For example, in the first row, the Diversity model only captures boats, water, and the pier, while our Diversity+TEB model captures the sky and the clouds beside the boats, beach, and water. In terms of detail, the Diversity model does not have the color description of the water, while our Diversity+TEB model has a color description for water, matching the ground truth. In general, Our Diversity+TEB model generates more human-like and coherent paragraphs, while the Diversity model tends to generate multiple sentences which are similar in meaning: The two sentences, "There are a large white boat in the water" and

”There is a large blue and white boat on the water”, provide very similar content. The comparisons in the remaining three rows further demonstrate that our Diversity+TEB model can generate more coherent and detailed paragraphs than the Diversity model.

Ablation study To demonstrate the robustness and generalizability of the TEB module, we do an ablation study on two baseline models: (1) Diversity with and without the TEB module and (2) Transformer with and without the TEB module. The diversity model uses an LSTM-based architecture as the decoder to generate paragraphs. The Transformer model uses the Bottom-Up and Top-Down model [107] as its backbone, but the decoder LSTM is replaced by the transformer model [108]. As shown in Table 4.1, the Transformer with the TEB module model achieves better performance than the Transformer model across all the three evaluation metrics. As mentioned in the above quantitative analysis and qualitative analysis sections, between the Diversity with the TEB module and the Diversity without the TEB module, the TEB does play an important role in improving the paragraph generation. In conclusion, these two ablation studies demonstrate that the TEB module is robust and can be applied to other existing image paragraph captioning models to improve the performance of the paragraph captioning.

Human Evaluation. To better understand and verify how the TEB module improves the image paragraph caption performance, a Turing test is performed to evaluate our Diversity+TEB against the baseline Diversity. In this Turing test, five well-educated evaluators are selected for human evaluation on 800 randomly chosen images from the testing set. The test procedure is as follows: The paragraphs that are generated by three methods (human annotation, Diversity, and Diversity+TEB) and the corresponding image are shown

one-by-one to the evaluators, who are then asked: Can you tell whether the given paragraph has been generated by a model or by a human being? According to the evaluators responses, the percentage of paragraphs that passed the Turing test is calculated. The results of the Turing test for Human, Diversity+TEB, and Diversity were 78.4 %, 43.2 %, and 19.1 %, clearly indicating that our Diversity+TEB outperforms the Diversity model.

4.6 Conclusion

In this chapter, we propose the Text Embedding Bank (TEB) module for image paragraph captioning, a task that requires capturing the fine-grained entities to generate a detailed and coherent paragraph. Our TEB module provides global and parallel deep supervision and distributed memory for nuanced image understanding and long-term language reasoning. Integrating the TEB module to existing state-of-the-art methods achieves new state-of-the-art results.

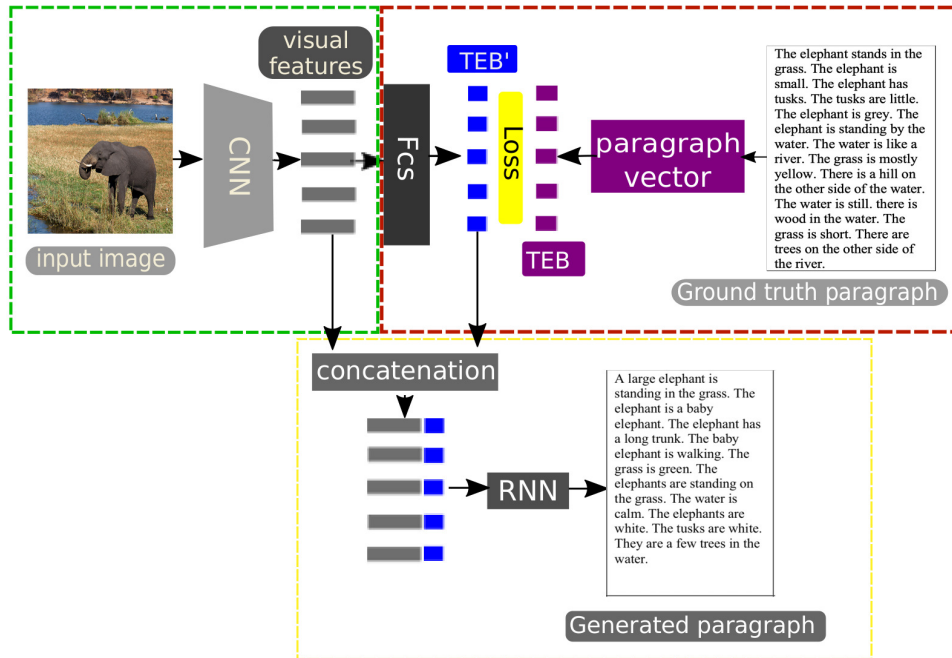


Figure 4.1: Integration of the paragraph vector framework as a TEB module to an existing deep learning-based image captioning model. There are three interconnected components divided into three dashed rectangular boxes. In the green box on the top left, the image encoder extracts visual features through a CNN model. In the yellow box on the bottom, an RNN-based language model decoder is used to generate paragraphs. Existing deep learning-based models only contain these two components. The red box on the top right box is the TEB module: In the training stage, for a image, paragraph pair, the varied-length paragraph is mapped to a fixed-length vector which is called TEB through the paragraph vector framework. The visual features from the image encoder are converted to the predicted TEB (called TEB') through several fully connected layers. The TEB' is supervised by the TEB through an L1 loss, which acts as global deep supervision to regularize the visual feature extraction for the image encoder. The visual features and TEB' are concatenated and fed into the RNN as input. The generated paragraph is supervised by the ground truth paragraph through a word-level loss. In the inference stage, the TEB is not available and the TEB' acts as the distributed memory to provide the semantic features of the whole paragraph to alleviate the long-term dependency problem for the language model.





Input Image	Ours (Diversity + TEB)	Diversity	Ground truth
	A bunch of boats are sitting on a beach. The water is calm and blue. There are a lot of boats in the water. The boats are white. There is a yellow umbrella on the boat. The sky is blue. The clouds are white and white.	A bunch of boats are on a pier. There are a large white boat in the water. There is a large blue and white boat on the water.	This is an image of a harbor. The harbor has many small boats in it. The water is blue. The water is reflecting the sky. The sky is partly cloudy. The clouds are white and fluffy. The sky is light blue. There are white buoys on the dock with small cloth sails in them. The sails are light brown and white.
	A man is standing in a of a man. The man is wearing a white shirt. The man has a black shirt on. The man is holding a hot dog. The sandwich is wearing a black. The men are wearing a blue shirt. There are people standing behind the man. There is a man in a blue shirt standing in the background. There are trees behind the man.	A man is standing in a white basket. He is wearing a black shirt and a black hat. The man is holding a hot dog in his hand. There is a man in a black shirt standing behind the man.	There is a man. The man is wearing a yellow shirt. The man is standing at a park. There are more people in the park. There are people sitting under the trees. There are people walking in the paths. The man is holding a sandwich. The sandwich is a hot dog. The sandwich has a sausage. The sandwich has onions.
	A large elephant is standing in the grass. The elephant is a baby elephant. The elephant has a long trunk. The baby elephant is walking. The grass is green. The elephants are standing on the grass. The water is calm. The elephants are white. The tusks are white. They are a few trees in the water.	A large elephant is standing in the water. The elephant is walking in the water. There is a large body of water behind the elephant. There are a small rock behind the elephant.	The elephant stands in the grass. The elephant is small. The elephant has tusks. The tusks are little. The elephant is grey. The elephant is standing by the water. The water is like a river. The grass is mostly yellow. There is a hill on the other side of the water. The water is still. there is wood in the water. The grass is short. There are trees on the other side of the river.
	A white toilet is in the bathroom. The toilet is white. The lid is white. There is a white toilet in the toilet. The toilet lid is up. The floor is made of white. The tiles are white. There is a white wall behind the toilet.	A white toilet is sitting on the ground. There is a white toilet in the toilet. There is a toilet in front of the toilet.	The toilet lid is up. The toilet bowl is cleaning. The toilet is a very light beige color. There's a white bar between the toilet lid and the toilet seat. The toilet is encased in a cubby space. The water in the toilet is low. The floor around the toilet is made of tiles. There are wires on the bottom left side of the toilet bowl.

Figure 4.2: Qualitative result comparison of paragraph outputs of our model (Diversity with TEB) and the baseline Diversity model [6]

CHAPTER 5

FUTURE WORK: TOWARDS EXTRACTING AND LEARNING VECTOR SPACE REPRESENTATIONS OF WORDS FOR IMAGE CAPTIONING

5.1 Introduction

Recent developments in Image Captioning have been inspired by advancements in object detection and machine translation in the past few years. The task of image captioning involves two main aspects: (1) resolving the object detection problem in computer vision and (2) creating a language model that can accurately generate a sentence describing the detected objects.

Seeing the success of encoder-decoder models with soft attention in the task of image caption [85], we use soft alignment [115] and modern approaches to object detection [116] as our baseline model. To extend this work, we investigate the effect of pre-trained embeddings on the task of image captioning by integrating GloVe embeddings [117] and BERT context vectors (Vaswani et al., 2017) to enhance the models performance and reduce training time.

The contributions of this chapter are the following:

- We provide an enhanced pyTorch implementation of the "soft" deterministic attention mechanism with an encoder-decoder architecture for image captioning as described in

[85].

- We integrate BERT context vectors and GloVe embeddings into our baseline model and enhance its performance.
- Finally, we visualize our results and quantitatively validate our models with the MS COCO validation dataset. We show that our BERT model integration outperforms the baseline model described in [85], while taking less time to train. We submit our results on the Microsoft COCO Image Captioning Challenge CodaLab test server. We show that our BERT model integration achieves the state-of-the-art result by a large margin.

5.2 Problem Description

Given a single raw image, our goal is to generate a caption \mathbf{y} encoded as a one-hot vector corresponding to our vocabulary.

$$\mathbf{y} = \{y_1, \dots, y_C\}, y_i \in R^V$$

where V is the size of the vocabulary and C is the length of the caption.

5.3 Data

There are several easily accessible datasets for training and validating models on the task of image captioning such as MS COCO, Flickr8k, and Flickr30k. In this paper, we use the MS COCO 2014 dataset for both training and validation. We utilized readily available MS

COCO cleaning and structuring scripts [118] to parse the captions, extract the vocabulary, and batch the images to optimize the training process for our models.

After re-sizing and normalizing all the images to 224x224 pixels, we extracted the captions and tokenized them with the NLTK tokenizer. Following that, we built a vocabulary with all the training dataset words, which came to be a list of 8,856 words.

5.4 Models and Algorithms

We use an encoder-decoder architecture to generate captions. The encoder is a Convolutional Neural Network (CNN) that takes in a single image and generates a vector describing the detected objects. This vector of objects is then passed to the decoder, which is a Long Short-Term Memory Network (LSTM) that attends to the image and outputs a descriptive caption one word at each time step.

In this section, we describe the encoder used in all our models and three variants of the attention-based decoder from [85]. The first decoder is an exact replica of the soft attention model described in [85] with optimized hyper-parameters. This initial model will act as our baseline. The second and third decoders are extensions on the baseline model that integrate GloVe embeddings and BERT’s pre-trained context vectors to the captions to enhance the model’s performance and reduce its training time.

5.4.1 Encoder

Similar to the encoder described in [85], we use a CNN to extract feature vectors from images. The Encoder produces L vectors, where each vector has D -dimensions that represent part of the image.

$$\mathbf{a} = \{a_1, \dots, a_L\} \in R^D$$

Although it is possible to create and train our own CNN for this task, we used the pre-trained ResNet-101 CNN as our encoder to reduce our training time and focus on enhancing the performance of the decoder. To use ResNet-101, we discard the pooling and linear layers—the last two layers—as we only need the image encoding, rather than the image classification. Then, we pass the output of the modified ResNet onto an adaptive pooling layer to create a fixed size output vector—fixed L —that can be easily passed to the decoder. We do not perform any fine-tuning to ResNet-101.

5.4.2 Decoder: Baseline Attention Model

We use a Long Short-Term Memory network to generate caption words one step at a time by conditioning on the previous step’s hidden state, the context vector, and the previously

generated words. Our implementation directly follows that of [85]

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ h_{t-1} \\ z_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

where i , f , o , and g are the input, forget, memory, and output states of the LSTM; h is the hidden state; c is the cell state (that keeps long term memory); and h_t denotes the hidden state at timestep t . Additionally, $T_{n,m}$ defines an affine transformation from dimension n to m . \odot is an element-wise multiplication.

\mathbf{E} represents an embedding matrix that is used and z_t denotes the context vectors of the relevant part of the image at time step t that is generated through soft-attention. To perform soft-attention in generating z_t , we use the "soft" attention mechanism detailed in Xu et al. (2016). In this baseline model, the caption embedding, \mathbf{E} , is learned alongside training the model to generate captions.

In the next two decoder descriptions, we will explain how we optimize \mathbf{E} to enhance the models performance.

5.4.3 Decoder: GloVe Attention Model

Recent methods for extracting and learning vector space representations for words have proven successful in capturing fine-grained semantic and syntactic regularities in words. In particular, GloVe word vectors [117] created a global log-bilinear regression model that generates word vector representations that enable Machine learning models to utilize these pre-trained embeddings. These embeddings are helpful because they can be pre-trained on vast amounts of text data instead of being trained alongside the task specific model, which usually has a much smaller dataset.

As an extension to the baseline decoder explained in the previous section, we integrate GloVe embeddings into our decoder by applying them to the images captions. However, we fine-tuned the embeddings alongside training our model to increase its accuracy and make it better fit the MS COCO dataset.

We downloaded the gloev.6b 300-dimensions pre-trained embeddings introduced in Pennington et al., (2014) and built a weights matrix that has a GloVe embedding for every word in our vocabulary. We then initialized the decoder embeddings with this weights matrix and fine-tuned it as we trained our model by propagating back the gradients.

5.4.4 Decoder: BERT Attention Model

In using the GloVe vector representations, each word is represented by a single unique vector no matter what context the word is used in. This raised concerns for several researchers

[119] as they realized that each word could have multiple meanings depending on where the word is used. Rather than having one representation for each word, BERT [119] uses a Transformer to generate a bi-directional contextualized word embedding conditioned on the context of the word in a sentence.

BERT has two distinct models: BERT base and BERT large. The base version has 12 encoder layers in its Transformer, 768 hidden units in its feedforward-network, and 12 attention heads. On the other hand, the large version has 24 encoder layers in its Transformer, 1024 hidden units in its feedforward-network, and 16 attention heads. Throughout our implementation, we use BERT base to generate the caption’s contextualized word vectors due to the increase in training time the large model introduces.

In the decoder, we take a batch of captions as our input $\mathbf{c} = \{c_1, \dots, c_B\}$, where B is the size of the batch and c_i is a full text representation of the caption. Then we iteratively take each caption c_i and perform the following steps on it:

1. Tokenize each caption with BERT’s wordPiece tokenizer to enable BERT to digest the caption and add the special '[CLS]' BERT token to the beginning of the caption.
2. Pass the wordPieces into BERT base.
3. Retrieve the output of the 12th layer (last layer) and discard the embedding of the special '[CLS]' token.
4. Detokenize the embeddings by summing the BERT context vectors of wordPieces that

belong to the same original word.

After doing the steps above to each caption in the batch we will have caption embeddings $\mathbf{b} = \{b_1, \dots, b_B\}$, where b_i is a tensor of size (caption size x 768) as each word has a vector of size 768 as its contextualized embedding. \mathbf{b} can then directly replace the GloVe embeddings and the trained embeddings used in the baseline model and the GloVe model respectively.

5.5 Experiments and Results

The Baseline, GloVe, and BERT models were trained and validated using the MS COCO 2014 dataset with the following optimized hyper-parameters obtained from [85]: (1) gradient clip = 5 (avoid gradient explosion), (2) number of epochs = 4 (limited to 4 epochs due to GPU accessibility), (3) batch size = 32, (4) decoder learning rate = 0.0004, (5) dropout rate = 0.5, (6) vocab size = 8856, (7) encoder dimension = 2,048 (based on RESNET-101’s output size), (8) attention dimension = 512, and (9) all weights initialized using a uniform distribution with range = [-0.1,0.1].

To implement the embedding extensions, we used embedding dimension of 512 for the baseline model, 300 for GloVe, and 768 for BERT. All the models were trained and validated on the same dataset splits with the same vocabulary to enable an accurate performance comparison.

Each epoch in the baseline and GloVe model took around 3.5 hours to train on a GTX 1070 Ti GPU, while the BERT model’s epoch took around 4.2 hours.

5.5.1 Baseline Attention Model

Figure 1 shows sample results obtained from the baseline model. Qualitatively analyzing the results, we see that the left image of Figure 1 has an accurate, grammatically correct hypothesis although the animal classifications are incorrect. Additionally, we see that the model used the word "herd" where it should have been "flock" due to its limited language model. The right image of Figure 1 shows a hypothesis that is more similar to the average hypotheses generated by the baseline model where many word repetitions occur. This means that the model correctly learned some representations, but did not yet finish training.

We have additionally noticed that this model is unable to generate sentences that have the same meaning as the reference sentences while using different words; it seems that the model is attempting to copy the reference sentence word by word. This can be explained by the fact that no pre-trained embeddings were used in this model. Therefore, it was difficult for the model to learn accurate word representations that would allow it to switch similar words.

5.5.2 GloVe Attention Model

Although the GloVe model has quantitative results similar to those of the baseline model in validation loss and BLEU scores, the GloVe model proved to be able to generate captions that use a different style of writing than the reference captions by using different words that are similar in meaning. This change can be explained by the fact that GloVe embeddings offer the model the ability to pick and choose the best possible word from a cluster of similar words. The left image in Figure 5.2 shows an example where the words 'Asian people'

were translated to 'children' in the generated caption. However, this model has repetition problems similar to those of the baseline model due to the limited training we did.

Table 5.1: Model validation loss and BLEU scores on the validation dataset

Model	Val (loss)	BLEU-1	BLEU-2	BLEU-3	BLEU-4
BASELINE MODEL	3.091	51.79	22.41	10.03	4.81
GLOVE MODEL	3.101	51.66	22.30	10.00	4.80
BERT MODEL	1.548	84.76	70.77	59.89	51.19

5.5.3 BERT Attention Model

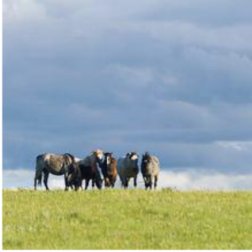
Although we expected the BERT model to outperform both the baseline and GloVe models because of its reliance on contextualized word embeddings, we were surprised by the extent of increase in BLEU score it obtained. The validation loss (Cross Entropy) decreased much faster in the BERT model, which shows that the BERT embeddings are very accurate in representing contextualized words. Additionally, this shows that context is very important in generating image captions, which makes a lot of sense because captions are supposed to relate objects in an image together and make sense of them.

Figure 5.3 shows two examples of the BERT model predicting captions; both captions

Table 5.2: Leaderboard of various methods on the online MS-COCO test server

Model	BLEU-1		BLEU-2		BLEU-3		BLEU-4		METEOR		ROUGE-L		CIDEr-D	
	c4	c40	c4	c40	c4	c40	c4	c40	c4	c40	c4	c40	c4	c40
SCST [110]	78.1	93.7	61.9	86.0	47.0	75.9	35.2	64.5	27.0	35.5	56.3	70.7	114.7	116.0
LSTM-A[120]	78.7	93.7	62.7	86.7	47.6	76.5	35.6	65.2	27.0	35.4	56.4	70.5	116.0	118.0
Up-Down[107]	80.2	95.2	64.1	88.8	49.1	79.4	36.9	68.5	27.6	36.7	57.1	72.4	117.9	120.5
RFNet[121]	80.4	95.0	64.9	89.3	50.1	80.1	38.0	69.2	28.2	37.2	58.2	73.1	122.9	125.1
GCN-LSTM[122]	-	-	65.5	89.3	50.8	80.3	38.7	69.7	28.5	37.6	58.5	73.4	125.3	126.5
SGAE[123]	81.0	95.3	65.6	89.5	50.7	80.4	38.5	69.7	28.2	37.2	58.6	73.6	123.8	126.5
AoANet[124]	81.0	95.0	65.8	89.6	51.4	81.3	39.4	71.2	29.1	38.5	58.9	74.5	126.9	129.6
EMB(ours)	84.6	96.1	70.6	92.6	59.8	90.4	51.2	83.4	29.4	39.1	59.3	75.1	136.5	141.7

Baseline Model



Hypotheses: a herd of sheep grazing in a field .
References: a pack of horses stand in a field

Baseline Model



Hypotheses: a man riding a surfboard on top of a surfboard surfboard .
References: a man riding a wave on top of a yellow surfboard .

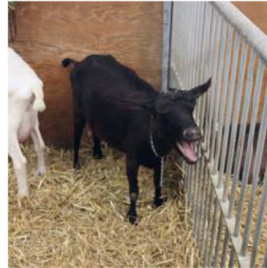
Figure 5.1: (left) Successful hypothesis from the baseline model. (right) Incorrect hypothesis from the baseline model.

GloVe Model



Hypotheses: a children sitting sitting at a table with food .
References: asian people are sitting at a table eating soup

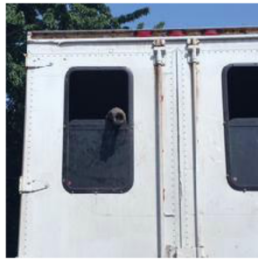
Glove Model



Hypotheses: a black standing a head open in a pen .
References: a sheep with its mouth open in a pen .

Figure 5.2: (left) Decent hypothesis from the GloVe model. (right) Incorrect hypothesis from the GloVe model.

BERT Model



Hypotheses: a elephant sticks his trunk out the back of of a truck .
References: an elephant sticks his trunk out the back window of a truck .

BERT Model



Hypotheses: a nice living room decorated to the christmas .
References: a lovely living room decorated for the holidays .

Figure 5.3: Accurate captions by the BERT model.

make sense and are grammatically correct. Interestingly, the predicted captions (hypotheses) predict similar sentences to the reference caption but use different word variants to explain similar things. In Figure 5.3, we see that 'back window' was translated to 'back' and 'holidays' was translated to 'Christmas'. This is interesting as it shows that the model offers correct captions that are not exactly the same as the reference captions, which is the goal of this task. BERT captions had few repetitions and were generally very well written.

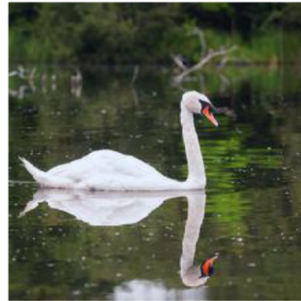
5.5.4 Summary of findings

Table 5.1 shows a full summary of our results. We validated our dataset by running a full single epoch on the MS COCO 2014 validation data and compared each hypotheses to 5 reference captions. Table 5.1 reports the validation loss, BLEU-1, BLEU-2, BLEU-3, and BLEU-4 which gives an accurate indication of how well each model performs.

Baseline and GloVe yielded very similar results, having only a slight improvement with Glove due to the introduction of pre-trained embeddings that were trained on vast amounts of data. BERT, on the other hand, had much better results in all aspects as shown in Table 5.1. Our BERT model results outperformed the results obtained by Xu et al. (2016) while being trained on fewer epochs [118].

The performances of different models on the Microsoft COCO Image Captioning Challenge CodaLab test server are shown in Table 5.2. Overall, the results across three evaluation metrics consistently indicate that our proposed BERT model integration achieves better performance. Specifically, the CIDEr and BLEU-4 scores of our BERT model integration achieve 7.57 % and 29.95%, for a 9.6% and 11.8% relative improvement over the next best

model (AOANet[124]), respectively.

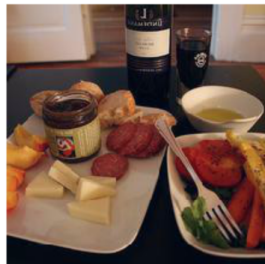


References: a swan floating in the water near a tree branch .

Baseline : a bird is in a water with a lake . .

GloVe : a bird is in the water near a pond . .

BERT : a duck floating in the water near a tree branch .



References: a platter filled with fruit , veggies and dip sits on top of a table .

Baseline : a table of with food and a and a . on a of a table .

GloVe : a plate of with food and vegetables , a . on a of a table .

BERT : a table filled with meat , veggies and fruit sits on top of a table .

Figure 5.4: Direct comparison of the three main models proposed.

Figure 5.4 shows a direct a comparison between the three models we implemented. The results clearly show that Baseline and GloVe yield similar results, while BERT outperforms them by large margins.

Figure 5.5 shows a failed attempt at implementing a program that visualizes the attention and maps it onto the image so that we can see what the model is attending to while predicting a specific word. The issue is how to map the attention values onto the image.

5.6 Conclusion

We proposed two extensions to the attention based approach to image captioning introduced in [85] that enhanced the performance of the model and reduced training time. Our BERT approach surpasses the MS COCO validation scores obtained by [85] while being trained on fewer epochs with the same hyper-parameters. Our experiments outline the importance of word embeddings in natural language processing and offer a new method in integrating BERT with already developed models to enhance their performance. We submit our results on the Microsoft COCO Image Captioning Challenge CodaLab test server. We show that our BERT model integration achieves the state-of-the-art result by a large margin.

Possible extensions to our work would be to train a new model with BERT large as apposed to BERT base, utilize beam search in validation, and train the models until the training loss converges.

Hypotheses: a woman street with a walking down walking on the street .
References: a city street with people walking and vehicles on the road .



Figure 5.5: Failed attempt to visualizing attention.

APPENDIX A

DC-DENSEUNET: 2D-3D DENSELY COUPLED, DENSELY CONNECTED UNET FOR AUTOMATIC LIVER LESION SEGMENTATION FROM CT VOLUMES

A.1 Introduction

Among cancers, that of the liver cancer is one of the most common and deadly [125, 126]. The accurate assessment of liver tumor volume, shape, location and texture can assist doctors in making diagnoses and in planning and evaluating treatments. 3D volumetric images exist widely in the medical imaging field; CT and MRI, for example, are the main imaging modalities for clinical diagnosis. Therefore, automatic liver and liver tumor segmentation methods are in high demand in clinical practice.

Automatic segmentation of liver and liver tumor in CT images is more challenging than that in natural images. First, the CT volume images are 3D volumes with anisotropic resolution. The resolution varies intra-slice between different CT images and the resolution is several times lower in the inter-slice dimension than that in the intra-slice dimension in the same CT image. Second, even in the contrast-enhanced CT volumes, there is a very low-intensity contrast between liver and its neighboring tissues and organs. Tumors, especially, vary greatly in location, shape and size (see Figure A.1). So due the heterogeneous and diffuse shape of the lesion, automatic segmentation of the lesion is very challenging. On the other hand, as manual segmentation of tumors by a radiologist is tedious and time-consuming,

many published methods utilize datasets containing less than 20 segmented tumors.

For a 3D volume image, it might be intuitive to just replace the 2D convolution with the 3D convolution in the state-of-the-art 2D fully convolutional neural networks (FCNNs) model. But there are several drawbacks to this approach: (1) With the same architecture, the memory of the 3D counterpart is too large to fit in the GPU memory limit, which constrains it from going deeper and wider to have the same performance as its 2D counterpart. (2) Since the resolution of the inter-slice is several times lower than that of intra-slice, it might be hard for a 3D kernel to learn stably in an anisotropic volume. The current solution is to resample the volume with a new resolution to make it isotropic. But the problem is that if the new resolution is too low—near the resolution of the inter-slice—then the small tumor will disappear and the whole new volume will be blurry and lacking in detail. On the other hand, if the new resolution is too high—close to the resolution of the intra-slice—then the dimension of the new volume will be huge, resulting in excessive computational load and memory issues. (3) The 3D networks lack both a pre-trained model and sufficient 3D data for generalization. On the other hand, directly applying a 2D FCNN on a 3D volume image has a severe theoretical limitation: 2D convolution cannot take into account the spatial feature in the intra-slice dimension.

In order to tackle the limitations of both 2D FCNs and 3D FCNs mentioned above, an efficient way of combining them seems in order. In this chapter, we propose a novel 2D-3D densely coupled, densely connected UNet (DC-DenseUNet) to jointly learn hybrid features from volume images with anisotropic resolution. The detailed pipeline and architecture of DC-DenseUNet are presented in Section 2.3. In general, the novelty and contributions of

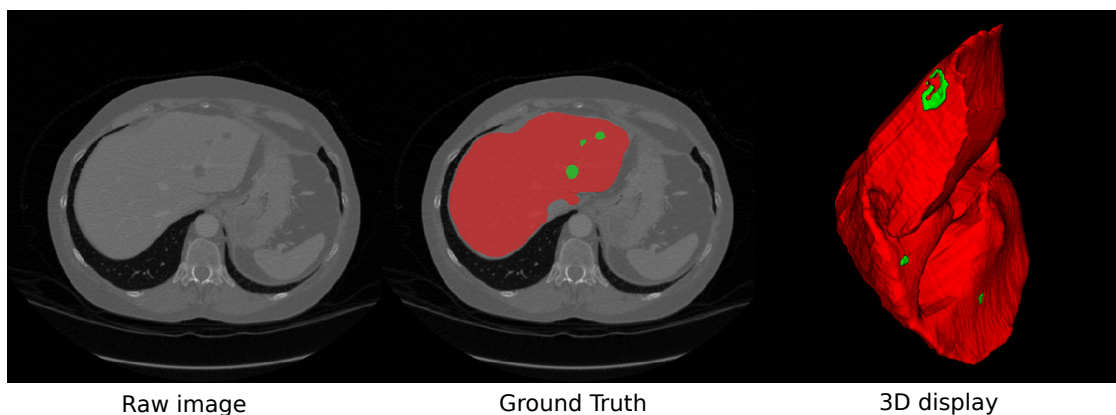


Figure A.1: Example of contrast-enhanced CT scans showing the large variations of shape, size, and location of liver lesions. The red regions denote the liver and the green ones denote the tumors.

this chapter are as follows:

- 1 We propose a novel densely coupled layer-wise feature fusion operation between 2D features and 3D features, where the 2D features and 3D features are from the corresponding layer of the 2D DenseUNet and 3D DenseUNet respectively. In this way, the performance can take the advantage of both 2D DenseUNet and 3D DenseUNet. The advantage of 2D DenseUNet is that there is no boundary for it to go deeper and wider. The advantage of 3D DenseUNet is that its 3D convolution can take into account the spatial feature in the inter-slice dimension.
- 2 DC-DenseUNet is designed to jointly learn to 2D and 3D features in an efficient manner. This densely coupled layer-wise feature fusion operation adds no new parameters or new FLOPs. This dense fusion has three types which will be described in detail in Section 2.3.

A.2 Related Work

A.2.1 Non-deep Learning Methods

Before deep learning was used for semantic image segmentation, the traditional non-learning-based approaches usually relied on hand-crafted features, including atlas-based[127], active shape model (ASM)-based [128], levelset-based [129], graph-cut-based[130] methods, etc.

A.2.2 Deep Learning Methods

Semantic segmentation assigns pixel-wise labels for a given image. The application of deep neural networks for image segmentation dates back to [131]. It can be treated as a dense classification problem. So the application of the deep neural network starts with replacing the last fully connected layer with the fully convolutional layer in the classification model. By convention, this model is called a fully convolutional network (FCNs)[132, 23]. FCNs have been dominant and proven successful on several segmentation benchmarks [133, 134, 2, 135, 136]. Following this trend, SegNet[137] adds deconvolution to form an encoder-decoder network. UNet[138] adds skip connection between the same level layers of an encoder-decoder network and has demonstrated big improvement for medical image segmentation. And V-net[139] used a similar strategy on volume image segmentation. PSPNet[140]. In order to resolve the anisotropic resolution in the 3D volume image, the methods have tried to re-sample the volume images with isotropic resolution [141, 142]. DeepEM3D-Net composes 3D convolutions in the early stage layers and 2D convolutions in the latter stage layers. DI2IN network [143, 144] treats the the CT image and its segmentation image as two domains and applies GAN[24] to map the image from one to the other. H-DenseUNet [145] concatenates

the pixel-wise prediction from 2D networks to the input of 3D networks and fuses the 2D features and 3D features from the last layer. PSPNet [140] uses different pool sizes and the fuses the multi-scale features together, but the limitation is that the pool size variation is limited and hard-defined. It lacks flexibility to fit the continuous resolution variation.

Recently, during the 2017 ISBI LiTS challenge, Han [146], proposed a 2.5D ResUNet which uses 5 adjacent slices as input to attain more z dimension spatial information. Liu [147] designed the architecture to transfer 2D features to 3D features in ResUNet and LSTM, and used focal loss, but the 2D and 3D features are not learned jointly and effectively. BDC-LSTM [148] treats slices as a time series in a bidirectional convolutional LSTM to explore the 3D contexts, an approach which still suffers from anisotropic features. Compared with all this related work, our method densely couples the 2D features from 2D DenseUNet and 3D features from 3D DenseUNet in an efficient way and improves the performance. Compared to the H-DenseUNet [145], this operation fuses the 2D features and 3D features not only at the last layer but at every layer; therefore, the well-learned 2D feature can be involved and help with the training of the 3D feature. This results in an effect similar to that of DenseNet, which fuses the features from the lower layer to the higher layer. Then the features from the lower layers can be involved and help the training of the 3D features.

A.3 Method

A.3.1 Pipeline

There are two stages in the pipeline:

Stage I: Coarse Segmentation of the Liver to Get the Region of Interest (ROI)

In this stage, we use the same method as in [142]. First, the whole CT image is resampled to $1mm \times 1mm \times 1mm$. Second, a 2.5D training method is used here, where the input is 3 adjacent slices and the mask of the corresponding center slice is the ground truth. We follow the same ResNet architecture as that in [142]. Finally, the trained ResNet model can generate a mask with label 1 for the liver region and 0 for background for each CT image in the training and testing datasets. Then each CT volume and its corresponding mask are cropped to contain only the liver area based on this generated mask. We call this cropped CT volume and its mask the ROI pair.

Stage II: Fine Segmentation of the Liver and Tumor

In this stage, there are two inputs: the 3D-input for the 3D DenseUNet and the 2D-input for the 2D DenseUNet. The ROI pair from the stage I is resampled to the original resolution. Then the 3D-input is randomly cropped from the ROI pair. The input pipeline is as follows: A $224 \times 224 \times 12$ volume patch is randomly cropped. This volume patch is passed to the 2D DenseUNet and 3D DenseUNet with batch sizes of 12 and 1, respectively. The dimension and batch size of the 3D-input are $224 \times 224 \times 12$ and 1 respectively. The dimension and batch size of the 2D-input are 224×224 and 12 respectively. The relationship between the 2D-input and 3D-input is as follows: Each of the 12 slices and its two adjacent slices from the volume patch are put together as the 2D-input with a batch size of 12. And the order of samples in the 2D-input batch follows the same slice order as in the 3D-input batch. The batch 12 slices are all from the same region. In the inference, the DC-DenseUNet outputs

Table A.1: DC-DenseUNet architecture

	Feature size	2D DenseUNet-165($k = 24$)	Feature size	3D DenseUNet-63 ($k = 16$)
input	224×224	-	$224 \times 224 \times 12$	-
convolution 1	112×112	$7 \times 7, 48, \text{stride } 2$	$112 \times 112 \times 6$	$7 \times 7 \times 7, 48, \text{stride } 2$
pooling	56×56	3×3 max pool, stride 2	$56 \times 56 \times 3$	$2 \times 2 \times 2$ max pool, stride 2
dense block 1	56×56	$\begin{bmatrix} 1 \times 1, & 96 \text{ conv} \\ 3 \times 3 & 24 \text{ conv} \end{bmatrix} \times 6$	$56 \times 56 \times 3$	$\begin{bmatrix} 1 \times 1 \times 1, & 64 \text{ conv} \\ 3 \times 3 \times 3 & 16 \text{ conv} \end{bmatrix} \times 3$
transition layer 1	56×56	1×1 conv	$56 \times 56 \times 3$	$1 \times 1 \times 1$ conv
	28×28	2×2 average pool	$28 \times 28 \times 3$	$2 \times 2 \times 1$ average pool
dense block 2	28×28	$\begin{bmatrix} 1 \times 1, & 96 \text{ conv} \\ 3 \times 3 & 24 \text{ conv} \end{bmatrix} \times 12$	$28 \times 28 \times 3$	$\begin{bmatrix} 1 \times 1 \times 1, & 64 \text{ conv} \\ 3 \times 3 \times 3 & 16 \text{ conv} \end{bmatrix} \times 4$
transition layer 2	28×28	1×1 conv	$28 \times 28 \times 3$	$1 \times 1 \times 1$ conv
	14×14	2×2 average pool	$14 \times 14 \times 3$	$2 \times 2 \times 1$ average pool
dense block 3	14×14	$\begin{bmatrix} 1 \times 1, & 96 \text{ conv} \\ 3 \times 3 & 24 \text{ conv} \end{bmatrix} \times 36$	$14 \times 14 \times 3$	$\begin{bmatrix} 1 \times 1 \times 1, & 64 \text{ conv} \\ 3 \times 3 \times 3 & 16 \text{ conv} \end{bmatrix} \times 12$
transition layer 3	14×14	1×1 conv	$14 \times 14 \times 3$	$1 \times 1 \times 1$ conv
	7×7	2×2 average pool	$7 \times 7 \times 3$	$2 \times 2 \times 1$ average pool
dense block 4	7×7	$\begin{bmatrix} 1 \times 1, & 96 \text{ conv} \\ 3 \times 3 & 24 \text{ conv} \end{bmatrix} \times 24$	$7 \times 7 \times 3$	$\begin{bmatrix} 1 \times 1 \times 1, & 64 \text{ conv} \\ 3 \times 3 \times 3 & 16 \text{ conv} \end{bmatrix} \times 8$
upsampling layer 1	14×14	2×2 upsampling [dense block 3,2D],384,conv	$14 \times 14 \times 3$	$2 \times 2 \times 1$ upsampling [dense block 3,2D,3D],252,conv
upsampling layer 2	28×28	2×2 upsampling [dense block 2,2D],192,conv	$28 \times 28 \times 3$	$2 \times 2 \times 1$ upsampling [dense block 2,2D,3D],112,conv
upsampling layer 3	56×56	2×2 upsampling [dense block 1,2D],96,conv	$56 \times 56 \times 3$	$2 \times 2 \times 1$ upsampling [dense block 1,2D,3D],48,conv
upsampling layer 4	112×112	2×2 upsampling [convolution 1,2D],48,conv	$112 \times 112 \times 6$	$2 \times 2 \times 2$ upsampling [convolution 1,2D,3D],48,conv
upsampling layer 5	224×224	2×2 upsampling,32,conv	$224 \times 224 \times 12$	$2 \times 2 \times 2$ upsampling [upsampling layer 5,2D],32,conv
convolution 2	224×224	$1 \times 1, 3, \text{conv}$	$224 \times 224 \times 12$	$1 \times 1 \times 1$ [convolution 2,2D], 3,conv
output	224×224	softmax,1	$224 \times 224 \times 12$	softmax,1

a cropped fine segmentation of the liver and tumor. We call this the fine mask. In the fine mask, the tumors that are outside of the liver region are removed first, then it is padding back to the whole size, and finally it is re-sampled to the original resolution.

A.3.2 DenseUNet Architecture

The 2D DenseUNet architecture is modified from the structure of DenseNet-161 [149], which is composed of several dense blocks and BC layers between them to change the feature map channel number between two consecutive dense blocks. In each dense block, there are direct connections from any layer to all subsequent layers. Each layer produces k feature maps and k is called the growth rate.

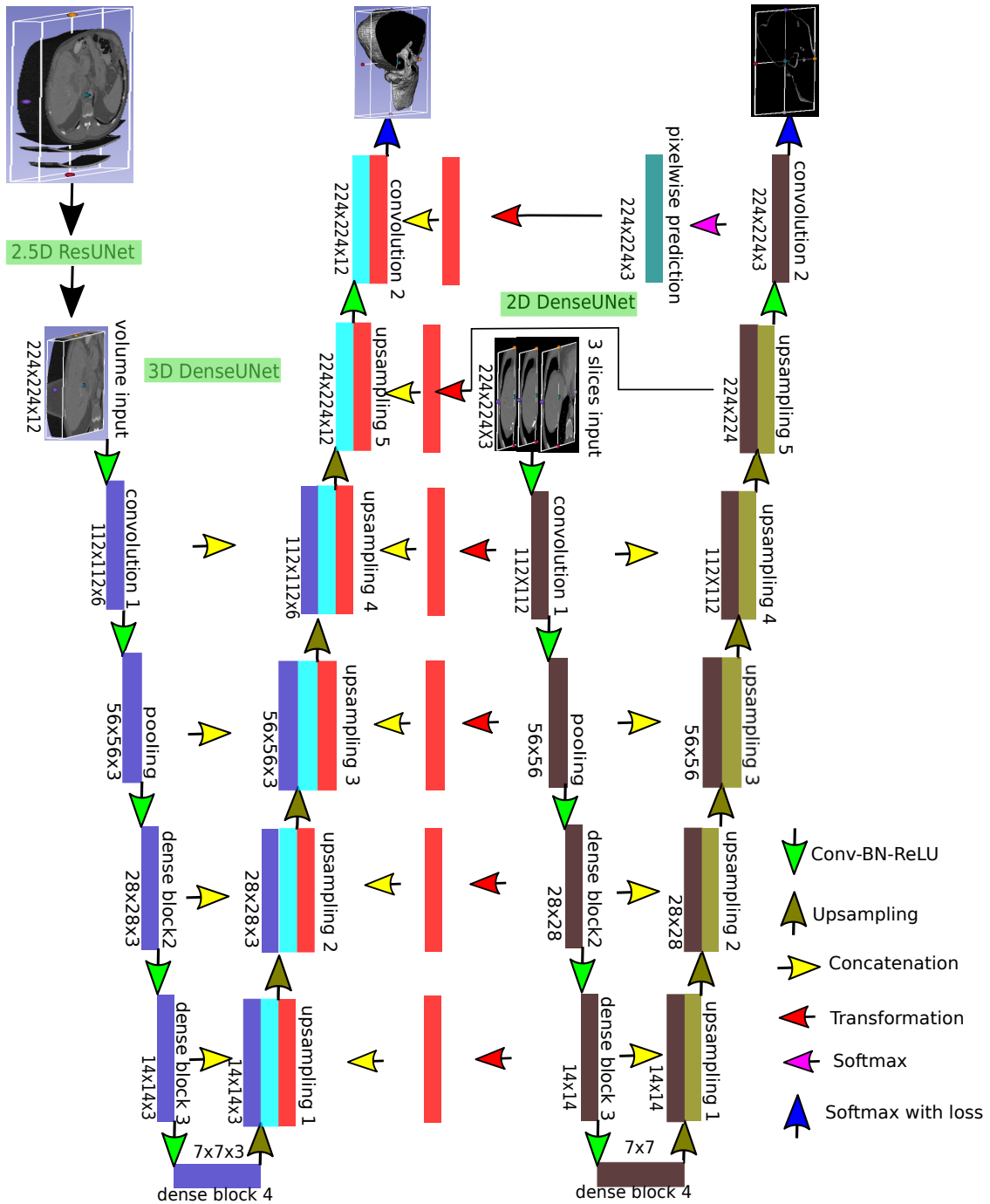


Figure A.2: DC-DenseUNet pipeline

A.3.3 Densely Coupled Fusion of The 2D and 3D Feature

As shown in Figure A.2 and Table A.1, there are two U-shaped parts coupled with the vertical red blocks.

3D DenseUNet

The left U-shaped part is the 3D DenseUNet: The blocks in deep blue color are the 3D feature map from the encoder,;in the encoder part, each block is followed by a Convolution-Batch normalization-ReLU operation. At the same time, the block in deep blue color is concatenated to the same level feature map in the decoder. The blocks in light blue color are the feature maps in the decoder; each block in this part is followed by the upsampling operation.

2D DenseUNet

The right U-shaped part is the 2D DenseUNet: The blocks in deep yellow are the 2D feature map from the encoder, and each block is followed by a Convolution-Batch normalization-ReLU operation. At the same time, the block in light yellow color is concatenated to the same level feature map in the decoder.

Densely Coupled Fusion of The 2D and 3D Feature

The fusion process consists of two steps: As shown in Figure A.2 and Table A.1, here we use the fusion at the dense block 2 as an example, the 2D feature (the block in deep yellow color) dimension is $12 \times 28 \times 28 \times 24$ ($B \times H \times W \times C$); this 2D feature is reshaped to the transition feature (the block in red) with dimension $1 \times 28 \times 28 \times 3 \times 96$ ($B \times H \times W \times D \times C$).Then

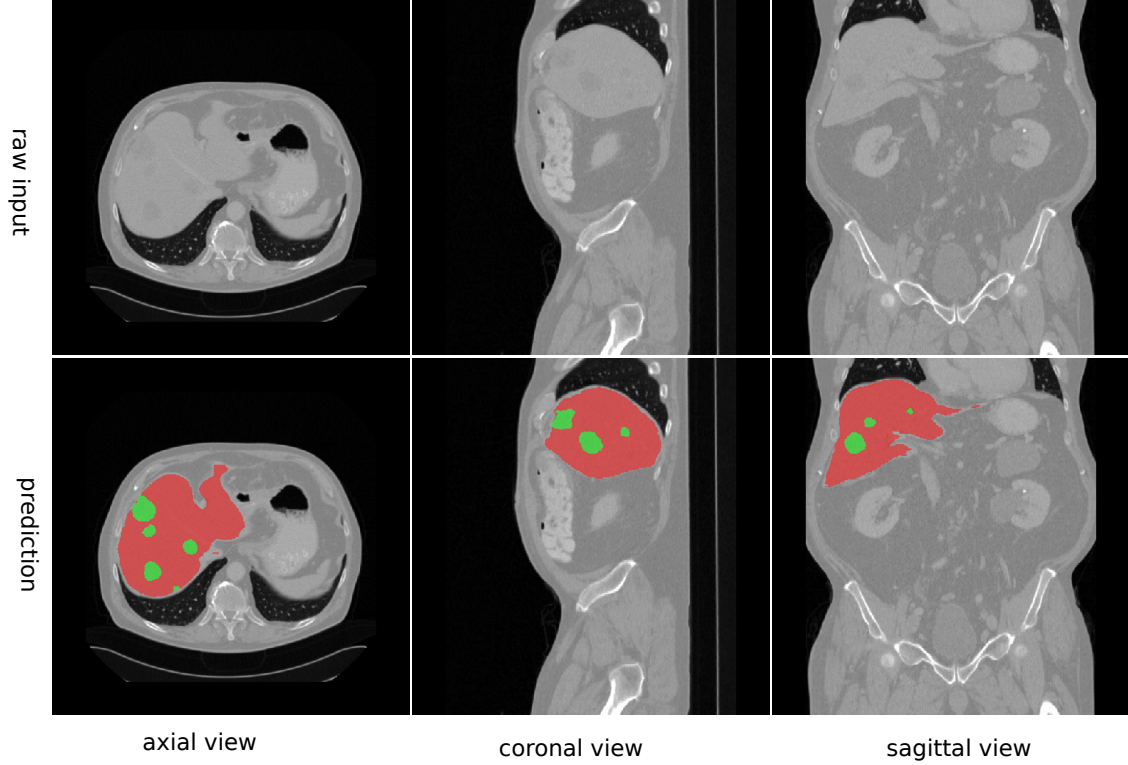


Figure A.3: Examples of liver and tumor segmentation results of DC-DenseUNet from the test dataset. The red regions denote the liver and the green ones denote the tumors.

this transition feature is concatenated with the 3D feature (the block in light blue color) with dimension $1 \times 28 \times 28 \times 3 \times 16$ ($B \times H \times W \times D \times C$). (Note: B denotes batch size; H denotes height; W denotes width; D denotes depth; C denotes channel number.) This fusion is followed by a $2 \times 2 \times 1$ convolution and the output channel is 224. There are three types of fusion based on where the fusion takes place. Type I is the fusion between the four dense blocks X from 2D DenseUNet and the four upsampling layers from 3D DenseUNet, where X is 1,2,3,4. Type II is the fusion between the layer upsampling 5 of the 3D DenseUNet and 2D DenseUNet. Type III is the pixel-wise probability map from 2D DenseUNet to the layer of the convolution 2 from 3D DenseUNet.

A.3.4 Loss Function, Training and Inference Schemes

In this section, we present more details regarding the loss function, training and the inference schemes.

1) Loss Function: To train the networks, we employed weighted cross-entropy function as the loss function, which is described as:

$$L(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^N w_i^c y_i^c \log \hat{y}_i^c$$

where y, i, c denotes the probability of voxel i belongs to class c (background, liver or lesion), w_{ic} denotes the weight and y_{ic} indicates the ground truth label for voxel i .

The total loss is:

$$L_{total} = \lambda L(y_{2d}, \hat{y}_{2d}) + L(y_{3d}, \hat{y}_{3d})$$

where (y_{2d}, \hat{y}_{2d}) and $L(y_{3d}, \hat{y}_{3d})$ are the cross-entropy loss for the 2D DenseUNet and 3D DenseUNet respectively. λ is the balanced weight and is set empirically as 0.5 in our experiments.

2) Training Scheme: We first train the 2.5D ResUNet in the same way as Han [142] to get the coarse liver segmentation results. Then a liver region cropped dataset is fed to DC-DenseUNet, and the whole network is jointly fine-tuned with the above total loss.

3) Inference Scheme: In the test stage, we first get the coarse liver segmentation result. Then DC-DenseUNet can generate accurate liver and tumor predicted probabilities within the ROI. After that, the final lesion segmentation result is obtained by removing lesions outside the final liver region.

A.4 Experiments and Results

A.4.1 Dataset, Pre-processing and Evaluation Metrics

A.4.2 DataSet

This Dataset is from MICCAI 2017 LiTS - Liver Tumor Segmentation Challenge (lits-challenge.com). The training dataset contains 130 CT scans, and the testing dataset contains 70 CT scans. All these CT scans are contrast enhanced. The in-plane resolution varies from 0.55 mm to 1.00 mm and the inter-slice resolution varies from 0.45 mm to 6.00 mm.

A.4.3 Pre-processing

In stage I, for the coarse segmentation of the liver, no special pre-processing was performed except that we truncated the image intensity value of all scans to the range of $[-200, 200]$ HU to remove the irrelevant details. And all the training images were resampled to a fixed resolution of $1 \times 1 \times 2.5mm^3$. In the second stage of fine segmentation of liver and liver tumor, the cropped liver region volume uses the original resolution. This is to avoid possible blurring from image resampling and to avoid missing very small lesions.

A.4.4 Evaluation Metrics

We employed DICE per case and global DICE as the evaluation metrics. The global Dice score combines all data sets into one, and the DICE per case averages the Dice per volume score over all the test cases. The qualitative result is shown in Figure A.3. As shown in Figure A.3, comparing the raw image and our prediction, even very small tumors are well

Table A.2: Segmentation results on the test dataset (from LiTS 2017 leaderaboard and publications)(Dice: %).

Model	Lesion		Liver	
	Dice per case	Dice global	Dice per case	Dice global
3D DenseUNet without pre-trained model [145]	59.4	78.8	96.3	92.9
UNet [146]	65	-	-	-
ResNet [142]	67	-	-	-
2D DenseUNet without pre-trained model [145]	67.7	80.1	94.7	94.7
2D DenseNet with pre-trained model [145]	68.3	81.8	95.3	95.9
hchen	68.6	82.9	96.1	96.5
2D DenseUNet with pre-trained model [145]	70.2	82.1	95.8	96.3
leHealth	70.2	79.4	96.1	96.7
AH-Net [147]	63.4	83.4	96.3	97.0
H-DenseUNet [145]	72.2	82.4	96.1	96.5
DC-DenseUNet(ours)	72.6	82.6	96.1	96.5

segmented. The quantitative result is in Table A.2. Our result achieved the best performance for lesion segmentation in DICE per case.

A.4.5 Ablation Study

We conducted comprehensive experiments to gauge the effectiveness of our approach. We provide an ablation study of our proposed approach on the segmentation results on the test dataset (see Table A.3). As stated in Section 2.3, the DC-DenseUNet consists of two sub-DenseUNets: 2D DenseUNet and 3D DenseUNet. As shown in Table A.3, we do the ablation study on the 2D DenseUNet and the 3D DenseUNet individually. To validate the effectiveness of confusion of 2D features and 3D features to solve the anisotropic resolution issues in the segmentation of volume data, we carry out the following comparisons:

- (1). Compared with the 2D DenseUNet, our method provides gains for both the liver and lesion segmentation. This indicates that the 3D features extracted from the 3D DenseUNet can help extract more intra-plane features.
- (2). Comparison between the 3D DenseUNet (same architecture as the 2D DenseUNet)

Table A.3: Segmentation results by ablation study of our methods on the test dataset.(Dice: %).

Model	Lesion		Liver	
	Dice per case	Dice global	Dice per case	Dice global
3D DenseUNet without pre-trained model	60.2	79.7	93.4	92.8
3D DenseUNet(same architecture as the 2D DenseUNet) without pre-trained model	63.4	79.8	93.9	93.2
2D DenseUNet without pre-trained model	67.5	79.9	94.6	94.7
2D DenseNet with pre-trained model	68.6	81.7	95.4	95.8
DC-DenseUNet(ours)	72.6	82.6	96.1	96.5

without pre-trained model and 2D DenseUNet without pre-trained model: The latter performs better. This indicates that only 3D kernel cannot learn stably in this anisotropic volume. Because this dataset has the same in-slice resolution while the resolution of the inter-slice is several times lower than the in-slice resolution. While for the 2D denseUNet, its 2D kernel can learn stably because the in-slice resolution is the same.

(3). Comparison between our method (DC-DenseUNet) and the 3D DenseUNet without pre-trained model: Our method provides gains both for the liver and lesion segmentation. This indicates that transforming and fusing the 2D in-slice features and 3D intra-slice features solve this anisotropic resolution issue effectively.

A.4.6 Implementation Details

This model is implemented using Tensorflow package [150]. The initial learning rate was 0.001 and decayed according to the equation $lr = lr \times \left(1 - \frac{Iteration}{TotalIteration}\right)^{0.9}$. We used stochastic gradient descent with momentum. For data augmentation, we adopted random mirror and scaling between 0.8 and 1.2 for all training data for data augmentation to alleviate the over-fitting problem. In the training phase, our model takes 25 hours to converge, while in the test phase, depending on the slice number of each volume, our model takes 60 to 260 seconds.

A.5 Conclusions

Proper segmentation of liver and liver tumors is a prerequisite for any accurate CAD system utilized in liver cancer treatment planning and monitoring as accurate volume calculation and location estimation are the keys to accurate prognosis. In this chapter, we propose the 2D-3D densely coupled DenseUNet network which is capable of fusing 2D features from 2D DenseUNet to 3D features and concatenating to 3D feature maps from 3D DenseUNet. After concatenation, the following convolution parameter is trained to optimize these two-dimensional features. By virtue of its concatenated fusion design and improved lesion segmentation, this method solves the anisotropic resolution in 3D volume images in learnable way.

REFERENCES

- [1] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest et al., “The multimodal brain tumor image segmentation benchmark (brats),” *IEEE transactions on medical imaging*, vol. 34, no. 10, pp. 1993–2024, 2014.
- [2] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.
- [3] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.
- [4] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.
- [5] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [6] L. Melas-Kyriazi, A. Rush, and G. Han, “Training for diversity in image paragraph captioning,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 757–761.
- [7] L. Xiang, Y. Chen, W. Chang, Y. Zhan, W. Lin, Q. Wang, and D. Shen, “Ultrafast t2-weighted mr reconstruction using complementary t1-weighted information,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018, pp. 215–223.
- [8] Y. Huang, L. Shao, and A. F. Frangi, “Simultaneous super-resolution and cross-modality synthesis of 3d medical images using weakly-supervised joint convolutional sparse coding,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6070–6079.

- [9] M. Chen, A. Jog, A. Carass, and J. L. Prince, “Using image synthesis for multi-channel registration of different image modalities,” in *Medical Imaging 2015: Image Processing*, vol. 9413. International Society for Optics and Photonics, 2015, p. 94131Q.
- [10] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, “Toward multimodal image-to-image translation,” in *Advances in Neural Information Processing Systems*, 2017, pp. 465–476.
- [11] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, “Stargan: Unified generative adversarial networks for multi-domain image-to-image translation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8789–8797.
- [12] A. Dosovitskiy and T. Brox, “Inverting visual representations with convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4829–4837.
- [13] A. Mahendran and A. Vedaldi, “Visualizing deep convolutional neural networks using natural pre-images,” *International Journal of Computer Vision*, vol. 120, no. 3, pp. 233–255, 2016.
- [14] A. V. Dalca, J. Guttag, and M. R. Sabuncu, “Anatomical priors in convolutional networks for unsupervised biomedical segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9290–9299.
- [15] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, “Parseval networks: Improving robustness to adversarial examples,” *arXiv preprint arXiv:1704.08847*, 2017.
- [16] J. Bruna, A. Szlam, and Y. LeCun, “Signal recovery from pooling representations,” *arXiv preprint arXiv:1311.4025*, 2013.
- [17] J.-H. Jacobsen, A. Smeulders, and E. Oyallon, “i-revnet: Deep invertible networks,” *arXiv preprint arXiv:1802.07088*, 2018.
- [18] L. Ardizzone, J. Kruse, S. Wirkert, D. Rahner, E. W. Pellegrini, R. S. Klessen, L. Maier-Hein, C. Rother, and U. Köthe, “Analyzing inverse problems with invertible neural networks,” *arXiv preprint arXiv:1808.04730*, 2018.
- [19] D. P. Kingma and P. Dhariwal, “Glow: Generative flow with invertible 1x1 convolutions,” in *Advances in Neural Information Processing Systems*, 2018, pp. 10 215–10 224.
- [20] D. Liu, Z. Wang, B. Wen, J. Yang, W. Han, and T. S. Huang, “Robust single image super-resolution via deep networks with sparse prior,” *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3194–3207, 2016.
- [21] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, “Image analogies,” in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 2001, pp. 327–340.

- [22] A. A. Efros and T. K. Leung, “Texture synthesis by non-parametric sampling,” in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 1033–1038.
- [23] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.
- [24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
- [25] J. Donahue, P. Krähenbühl, and T. Darrell, “Adversarial feature learning,” *arXiv preprint arXiv:1605.09782*, 2016.
- [26] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, “Scribbler: Controlling deep image synthesis with sketch and color,” *arXiv preprint arXiv:1612.00835*, 2016.
- [27] L. Karacan, Z. Akata, A. Erdem, and E. Erdem, “Learning to generate images of outdoor scenes from attributes and semantic layouts,” *arXiv preprint arXiv:1612.00215*, 2016.
- [28] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, “High-resolution image synthesis and semantic manipulation with conditional gans,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [29] A. Almahairi, S. Rajeswar, A. Sordoni, P. Bachman, and A. Courville, “Augmented cyclegan: Learning many-to-many mappings from unpaired data,” *arXiv preprint arXiv:1802.10151*, 2018.
- [30] M.-Y. Liu, T. Breuel, and J. Kautz, “Unsupervised image-to-image translation networks,” in *Advances in neural information processing systems*, 2017, pp. 700–708.
- [31] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang, “Diverse image-to-image translation via disentangled representations,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 35–51.
- [32] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, “Multimodal unsupervised image-to-image translation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 172–189.
- [33] H.-Y. Lee, H.-Y. Tseng, Q. Mao, J.-B. Huang, Y.-D. Lu, M. Singh, and M.-H. Yang, “Drit++: Diverse image-to-image translation via disentangled representations,” *arXiv preprint arXiv:1905.01270*, 2019.
- [34] Z. Shen, M. Huang, J. Shi, X. Xue, and T. Huang, “Towards instance-level image-to-image translation,” *arXiv preprint arXiv:1905.01744*, 2019.

- [35] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2414–2423.
- [36] Q. Chen and V. Koltun, “Photographic image synthesis with cascaded refinement networks,” in *IEEE International Conference on Computer Vision (ICCV)*, vol. 1, no. 2, 2017, p. 3.
- [37] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. S. Lempitsky, “Texture networks: Feed-forward synthesis of textures and stylized images.” in *ICML*, 2016, pp. 1349–1357.
- [38] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in *European Conference on Computer Vision*. Springer, 2016, pp. 694–711.
- [39] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.
- [40] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training gans,” in *Advances in Neural Information Processing Systems*, 2016, pp. 2234–2242.
- [41] X. Wang and A. Gupta, “Generative image modeling using style and structure adversarial networks,” in *European Conference on Computer Vision*. Springer, 2016, pp. 318–335.
- [42] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in *European Conference on Computer Vision*. Springer, 2016, pp. 649–666.
- [43] A. Owens, P. Isola, J. McDermott, A. Torralba, E. H. Adelson, and W. T. Freeman, “Visually indicated sounds,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2405–2413.
- [44] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [45] A. Hore and D. Ziou, “Image quality metrics: Psnr vs. ssim,” in *Pattern recognition (icpr), 2010 20th international conference on*. IEEE, 2010, pp. 2366–2369.
- [46] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.
- [47] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

- [48] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, “Learning to discover cross-domain relations with generative adversarial networks,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1857–1865.
- [49] L. Song, Z. Lu, R. He, Z. Sun, and T. Tan, “Geometry guided adversarial facial expression synthesis,” in *2018 ACM Multimedia Conference on Multimedia Conference*. ACM, 2018, pp. 627–635.
- [50] A. Pumarola, A. Agudo, A. M. Martinez, A. Sanfeliu, and F. Moreno-Noguer, “Ganimation: Anatomically-aware facial animation from a single image,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 818–833.
- [51] Z. Zhang, L. Yang, and Y. Zheng, “Translating and segmenting multimodal medical volumes with cycle-and shape-consistency generative adversarial network,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9242–9251.
- [52] Y. Zhang, S. Miao, T. Mansi, and R. Liao, “Task driven generative modeling for unsupervised domain adaptation: Application to x-ray image segmentation,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018, pp. 599–607.
- [53] R. Zhang, T. Pfister, and J. Li, “Harmonic unpaired image-to-image translation,” *arXiv preprint arXiv:1902.09727*, 2019.
- [54] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv preprint arXiv:1511.06434*, 2015.
- [55] Z. Shen, Y. Chen, S. K. Zhou, B. Georgescu, X. Liu, and T. S. Huang, “Towards learning a self-inverse network for bidirectional image-to-image translation,” *arXiv preprint arXiv:1909.04104*, 2019.
- [56] Z. Gan, L. Chen, W. Wang, Y. Pu, Y. Zhang, H. Liu, C. Li, and L. Carin, “Triangle generative adversarial networks,” in *Advances in Neural Information Processing Systems*, 2017, pp. 5247–5256.
- [57] G. Lu, Z. Zhou, Y. Song, K. Ren, and Y. Yu, “Guiding the one-to-one mapping in cycleGAN via optimal transport,” *arXiv preprint arXiv:1811.06284*, 2018.
- [58] R. Zhang, T. Pfister, and J. Li, “Harmonic unpaired image-to-image translation,” *CoRR*, vol. abs/1902.09727, 2019. [Online]. Available: <http://arxiv.org/abs/1902.09727>
- [59] T. Dekel, C. Gan, D. Krishnan, C. Liu, and W. T. Freeman, “Smart, sparse contours to represent and edit images,” *arXiv preprint arXiv:1712.08232*, 2017.

- [60] Y. Lu, S. Wu, Y.-W. Tai, and C.-K. Tang, “Image generation from sketch constraint using contextual gan,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 205–220.
- [61] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, G. Liu, A. Tao, J. Kautz, and B. Catanzaro, “Video-to-video synthesis,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.
- [62] A. Bansal, S. Ma, D. Ramanan, and Y. Sheikh, “Recycle-gan: Unsupervised video retargeting,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 119–135.
- [63] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, “High-resolution image synthesis and semantic manipulation with conditional gans,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8798–8807.
- [64] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang, “Mode seeking generative adversarial networks for diverse image synthesis,” *arXiv preprint arXiv:1903.05628*, 2019.
- [65] S. Ma, J. Fu, C. Wen Chen, and T. Mei, “Da-gan: Instance-level image translation by deep attention generative adversarial networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5657–5666.
- [66] X. Chen, C. Xu, X. Yang, and D. Tao, “Attention-gan for object transfiguration in wild images,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 164–180.
- [67] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim, “Unsupervised attention-guided image-to-image translation,” in *Advances in Neural Information Processing Systems*, 2018, pp. 3693–3703.
- [68] X. Liang, H. Zhang, L. Lin, and E. Xing, “Generative semantic manipulation with mask-contrasting gan,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 558–573.
- [69] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [70] S. Benaim and L. Wolf, “One-sided unsupervised domain mapping,” in *Advances in neural information processing systems*, 2017, pp. 752–762.
- [71] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest et al., “The multimodal brain tumor image segmentation benchmark (brats),” *IEEE transactions on medical imaging*, vol. 34, no. 10, pp. 1993–2024, 2015.

- [72] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [73] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, “From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions,” *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 67–78, 2014.
- [74] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma et al., “Visual genome: Connecting language and vision using crowdsourced dense image annotations,” *International Journal of Computer Vision*, vol. 123, no. 1, pp. 32–73, 2017.
- [75] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille, “Deep captioning with multimodal recurrent neural networks (m-rnn),” *arXiv preprint arXiv:1412.6632*, 2014.
- [76] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, “Image captioning with semantic attention,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4651–4659.
- [77] Z. Shen, J. Li, Z. Su, M. Li, Y. Chen, Y.-G. Jiang, and X. Xue, “Weakly supervised dense video captioning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1916–1924.
- [78] J. Krause, J. Johnson, R. Krishna, and L. Fei-Fei, “A hierarchical approach for generating descriptive image paragraphs,” in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 3337–3345.
- [79] X. Liang, Z. Hu, H. Zhang, C. Gan, and E. P. Xing, “Recurrent topic-transition gan for visual paragraph generation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3362–3371.
- [80] H. Greenspan, B. Van Ginneken, and R. M. Summers, “Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique,” *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1153–1159, 2016.
- [81] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, “Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases,” in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 3462–3471.
- [82] X. Wang, Y. Peng, L. Lu, Z. Lu, and R. M. Summers, “Tienet: Text-image embedding network for common thorax disease classification and reporting in chest x-rays,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9049–9058.
- [83] C. Y. Li, X. Liang, Z. Hu, and E. P. Xing, “Hybrid retrieval-generation reinforced agent for medical image report generation,” *arXiv preprint arXiv:1805.08298*, 2018.

- [84] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.
- [85] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in *International conference on machine learning*, 2015, pp. 2048–2057.
- [86] J. Johnson, A. Karpathy, and L. Fei-Fei, “Densecap: Fully convolutional localization networks for dense captioning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4565–4574.
- [87] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in neural information processing systems*, 2015, pp. 91–99.
- [88] J. Aneja, A. Deshpande, and A. G. Schwing, “Convolutional image captioning,” *arXiv preprint arXiv:1711.09151*, 2017.
- [89] Q. Wang and A. B. Chan, “Cnn+ cnn: Convolutional decoders for image captioning,” *arXiv preprint arXiv:1805.09019*, 2018.
- [90] Y. Chen, S. Wang, W. Zhang, and Q. Huang, “Less is more: Picking informative frames for video captioning,” *arXiv preprint arXiv:1803.01457*, 2018.
- [91] J. Li, M.-T. Luong, and D. Jurafsky, “A hierarchical neural autoencoder for paragraphs and documents,” *arXiv preprint arXiv:1506.01057*, 2015.
- [92] R. Lin, S. Liu, M. Yang, M. Li, M. Zhou, and S. Li, “Hierarchical recurrent neural network for document modeling,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 899–907.
- [93] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu, “Video paragraph captioning using hierarchical recurrent neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4584–4593.
- [94] B. Jing, P. Xie, and E. Xing, “On the automatic generation of medical imaging reports,” *arXiv preprint arXiv:1711.08195*, 2017.
- [95] M. Chatterjee and A. G. Schwing, “Diverse and coherent paragraph generation from images,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 729–744.
- [96] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [97] J. Wang, Y. Pan, T. Yao, J. Tang, and T. Mei, “Convolutional auto-encoding of sentence topics for image paragraph generation,” *arXiv preprint arXiv:1908.00249*, 2019.

- [98] Y. Zhang, Z. Gan, K. Fan, Z. Chen, R. Henao, D. Shen, and L. Carin, “Adversarial feature matching for text generation,” *arXiv preprint arXiv:1706.03850*, 2017.
- [99] L. Yu, W. Zhang, J. Wang, and Y. Yu, “Seqgan: Sequence generative adversarial nets with policy gradient.” in *AAAI*, 2017, pp. 2852–2858.
- [100] J. Guo, S. Lu, H. Cai, W. Zhang, Y. Yu, and J. Wang, “Long text generation via adversarial training with leaked information,” *arXiv preprint arXiv:1709.08624*, 2017.
- [101] W. Fedus, I. Goodfellow, and A. M. Dai, “Maskgan: Better text generation via filling in the _,” *arXiv preprint arXiv:1801.07736*, 2018.
- [102] C.-Y. Wu, C. Feichtenhofer, H. Fan, K. He, P. Krahenbuhl, and R. Girshick, “Long-term feature banks for detailed video understanding,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 284–293.
- [103] D. E. Rumelhart, G. E. Hinton, R. J. Williams et al., “Learning representations by back-propagating errors,” *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.
- [104] Y. Bengio, H. Schwenk, J.-S. Senécal, and F. Morin, “Gauvain, jean-luc. neural probabilistic language models,” *Innovations in Machine Learning*, pp. 137–186.
- [105] Q. Le and T. Mikolov, “Distributed representations of sentences and documents,” in *International Conference on Machine Learning*, 2014, pp. 1188–1196.
- [106] J. H. Lau and T. Baldwin, “An empirical evaluation of doc2vec with practical insights into document embedding generation,” *arXiv preprint arXiv:1607.05368*, 2016.
- [107] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, “Bottom-up and top-down attention for image captioning and visual question answering,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6077–6086.
- [108] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [109] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3128–3137.
- [110] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, “Self-critical sequence training for image captioning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7008–7024.
- [111] S. Banerjee and A. Lavie, “Meteor: An automatic metric for mt evaluation with improved correlation with human judgments,” in *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, 2005, pp. 65–72.

- [112] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, “Cider: Consensus-based image description evaluation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4566–4575.
- [113] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.
- [114] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick, “Microsoft coco captions: Data collection and evaluation server,” *arXiv preprint arXiv:1504.00325*, 2015.
- [115] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [116] J. Ba, V. Mnih, and K. Kavukcuoglu, “Multiple object recognition with visual attention,” *arXiv preprint arXiv:1412.7755*, 2014.
- [117] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
- [118] “A PyTorch Tutorial to Image Captioning , howpublished = <https://github.com/sgrvinod/a-pytorch-tutorial-to-image-captioning>, note = Accessed: 2019-12-07.”
- [119] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [120] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei, “Boosting image captioning with attributes,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4894–4902.
- [121] W. Jiang, L. Ma, Y.-G. Jiang, W. Liu, and T. Zhang, “Recurrent fusion network for image captioning,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 499–515.
- [122] T. Yao, Y. Pan, Y. Li, and T. Mei, “Exploring visual relationship for image captioning,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 684–699.
- [123] X. Yang, K. Tang, H. Zhang, and J. Cai, “Auto-encoding scene graphs for image captioning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 685–10 694.

- [124] L. Huang, W. Wang, J. Chen, and X.-Y. Wei, “Attention on attention for image captioning,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4634–4643.
- [125] J. Ferlay, H.-R. Shin, F. Bray, D. Forman, C. Mathers, and D. M. Parkin, “Estimates of worldwide burden of cancer in 2008: Globocan 2008,” *International journal of cancer*, vol. 127, no. 12, pp. 2893–2917, 2010.
- [126] R. Lu, P. Marziliano, and C. H. Thng, “Liver tumor volume estimation by semi-automatic segmentation method,” in *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*. IEEE, 2006, pp. 3296–3299.
- [127] M. G. Linguraru, J. K. Sandberg, Z. Li, J. A. Pura, and R. M. Summers, “Atlas-based automated segmentation of spleen and liver using adaptive enhancement estimation,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2009, pp. 1001–1008.
- [128] D. Kainmüller, T. Lange, and H. Lamecker, “Shape constrained automatic segmentation of the liver based on a heuristic intensity model,” in *Proc. MICCAI Workshop 3D Segmentation in the Clinic: A Grand Challenge*, 2007, pp. 109–116.
- [129] J. Lee, N. Kim, H. Lee, J. B. Seo, H. J. Won, Y. M. Shin, Y. G. Shin, and S.-H. Kim, “Efficient liver segmentation using a level-set method with optimal detection of the initial liver boundary from level-set speed images,” *Computer methods and programs in biomedicine*, vol. 88, no. 1, pp. 26–38, 2007.
- [130] L. Massoptier and S. Casciari, “Fully automatic liver segmentation through graph-cut technique,” in *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*. IEEE, 2007, pp. 5243–5246.
- [131] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.
- [132] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” *arXiv preprint arXiv:1312.6229*, 2013.
- [133] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International journal of computer vision*, vol. 111, no. 1, pp. 98–136, 2015.
- [134] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille, “The role of context for object detection and semantic segmentation in the wild,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 891–898.

- [135] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ade20k dataset,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, no. 2. IEEE, 2017, p. 4.
- [136] H. Caesar, J. Uijlings, and V. Ferrari, “Coco-stuff: Thing and stuff classes in context,” *CoRR*, *abs/1612.03716*, vol. 5, p. 8, 2016.
- [137] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [138] V. Badrinarayanan, A. Kendall, and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.
- [139] F. Milletari, N. Navab, and S.-A. Ahmadi, “V-net: Fully convolutional neural networks for volumetric medical image segmentation,” in *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE, 2016, pp. 565–571.
- [140] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2881–2890.
- [141] P. Moeskops, M. A. Viergever, A. M. Mendrik, L. S. de Vries, M. J. Benders, and I. Išgum, “Automatic segmentation of mr brain images with a convolutional neural network,” *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1252–1261, 2016.
- [142] X. Han, “Automatic liver lesion segmentation using a deep convolutional neural network method,” *arXiv preprint arXiv:1704.07239*, 2017.
- [143] D. Yang, D. Xu, S. K. Zhou, B. Georgescu, M. Chen, S. Grbic, D. Metaxas, and D. Comaniciu, “Automatic liver segmentation using an adversarial image-to-image network,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2017, pp. 507–515.
- [144] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, “Semantic segmentation using adversarial networks,” *arXiv preprint arXiv:1611.08408*, 2016.
- [145] X. Li, H. Chen, X. Qi, Q. Dou, C.-W. Fu, and P. A. Heng, “H-denseunet: Hybrid densely connected unet for liver and liver tumor segmentation from ct volumes,” *arXiv preprint arXiv:1709.07330*, 2017.
- [146] G. Chlebus, H. Meine, J. H. Moltz, and A. Schenk, “Neural network-based automatic liver tumor segmentation with random forest-based candidate filtering,” *arXiv preprint arXiv:1706.00842*, 2017.
- [147] S. Liu, D. Xu, S. K. Zhou, T. Mertelmeier, J. Wicklein, A. Jerebko, S. Grbic, O. Pauly, W. Cai, and D. Comaniciu, “3d anisotropic hybrid network: Transferring convolutional features from 2d images to 3d anisotropic volumes,” *arXiv preprint arXiv:1711.08580*, 2017.

- [148] J. Chen, L. Yang, Y. Zhang, M. Alber, and D. Z. Chen, “Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation,” in *Advances in Neural Information Processing Systems*, 2016, pp. 3036–3044.
- [149] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, vol. 1, no. 2, 2017, p. 3.
- [150] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin et al., “Tensorflow: Large-scale machine learning on heterogeneous distributed systems,” *arXiv preprint arXiv:1603.04467*, 2016.