# Exploring Deep Representation Learning on Vision and Language Intelligence

by

Shuyang Dai

Department of Electrical and Computer Engineering
Duke University

Date: _____

Approved:

_____

Lawerence Carin, Advisor

_____

David Carlson

_____

Yiran Chen

_____

Hai Li

_____

Rabih Younes

Dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Department of Electrical and Computer Engineering
in the Graduate School of
Duke University

2021

# ABSTRACT

## Exploring Deep Representation Learning on Vision and Language Intelligence

by

Shuyang Dai

Department of Electrical and Computer Engineering
Duke University

Date: _____

Approved:

_____

Lawerence Carin, Advisor

_____

David Carlson

_____

Yiran Chen

_____

Hai Li

_____

Rabih Younes

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical and Computer Engineering in the Graduate School of Duke University

2021

Copyright © 2021 by Shuyang Dai
All rights reserved

# Abstract

Deep neural networks have achieved tremendous success in recent years, with applications in various tasks involving both computer vision and natural language processing. Representation learning is often adopted to extract useful latent features for these tasks. In this dissertation, I will discuss the contributions that I have made in using representation learning methodologies for deep generative models, as well as unsupervised domain adaptation.

The first part of the dissertation will mainly focus on deep generative models for vision and language intelligence. I will present Symmetric Variational Autoencoder, which unifies the Variational Bayesian and adversarial training frameworks. Then, I will show the application of such generative models in the natural language domain, and present a VAE framework with a hyperbolic latent space.

For the second part, I will mainly focus on representation learning for unsupervised domain adaptation (UDA). In this problem setup, we want to extract representative features that contain mostly task-oriented information but little domain-related information. I will first present to learn such features in a contrastive manner: pulling data of the same class together while pushing those that are not away from each other. Next, I will focus on UDA where large domain gaps exist. To tackle such a UDA problem, I propose to use unlabeled domain bridges, and transform the original problem into several intermediate ones.

# Acknowledgements

Being a Ph.D. at Duke University was full of excitement, and joining the group of Professor Lawrence Carin was one of the best decision I have ever made. I would like to thank my advisor, Professor Carin, for all his guidance and support. He has taught me not only the right way to research, but also the right attitude to face all kinds of challenges through my career. I would always remember what he told us during our weekly meetings: progress is all that matters, not the final outcome. Thank you Larry, for being a great lifetime mentor of mine.

I would also like to show my appreciation for my dissertation committee members, Professors David Carlson, Yiran Chen, Hai Li, and Rabih Younes, for their time and effort to serve on my committee.

I would like to thank Dr. Kihyuk Sohn, Dr. Yi-Hsuan Tsai, and Professor Manmohan Chandraker for hosting me as a summer research assistant at NEC Laboratories. I am also very grateful to Dr. Zhe Gan, Dr. Yu Cheng, and Dr. Jingjing Liu for providing me with great guidance and help when I was a research intern at Microsoft.

Last but not the least, I want to thank my fellow group members, who have helped me either in or outside my research, through my Ph.D. career. Amongst many others, they are Yitong Li, Xinyuan Zhang, Dong Wang, Kevin Liang, Wenlin Wang, Bai Li, Jianqiao Li, Chunyuan Li, Yulai Cong, Miaoyun Zhao, Dinghan Shen, Zhe Gan, Yizhe Zhang, Yunchen Pu, Liqun Chen, Siyang Yuan, Paidamoyo Chapfuwa, Gregory Spell, Ruiyi Zhang, Qi Wei, Guoyin Wang, Pengyu Cheng, Ke Bai and Chenyang Tao.

# Contents

**Abstract** **iv**

**Acknowledgements** **v**

**List of Figures** **x**

**List of Tables** **xii**

**1 Introduction** **1**

1.1 Background . . . . . 1

1.1.1 Variational Autoencoder . . . . . 1

1.1.2 Generative Adversarial Networks . . . . . 3

1.1.3 Adversarial Variational Bayes . . . . . 5

1.1.4 Domain Adversarial Neural Networks . . . . . 7

1.2 Thesis Contribution . . . . . 9

1.2.1 Unifying VAE and Adversarial Training . . . . . 9

1.2.2 Learning Sentence Representations in a Hyperbolic Space . . . 9

1.2.3 Adaptation via two Contrastively Trained Classifiers . . . . . 10

1.2.4 Tackling Large Domain Gap using Unlabeled Bridges . . . . . 11

**2 Symmetric Variational Autoencoder** **12**

2.1 Introduction . . . . . 12

2.2 Review of Variational Autoencoder . . . . . 14

2.2.1 Background . . . . . 14

2.2.2 Limitations of the VAE . . . . . 16

2.3 Refined VAE: Imposition of Symmetry . . . . . 18

2.3.1 Symmetric KL divergence . . . . . 18
2.3.2 Adversarial solution . . . . . 19
2.3.3 Interpretation in terms of LRT statistic . . . . . 20
2.4 Connections to Prior Work . . . . . 21
2.4.1 Adversarially Learned Inference . . . . . 21
2.4.2 Original GAN . . . . . 22
2.4.3 Wasserstein GAN . . . . . 23
2.4.4 Amelioration of vanishing gradients . . . . . 24
2.5 Model Augmentation . . . . . 25
2.6 Experiments . . . . . 27
2.6.1 Toy Data . . . . . 27
2.6.2 MNIST . . . . . 28
2.6.3 CelebA . . . . . 29
2.6.4 CIFAR-10 . . . . . 31
2.7 Conclusions . . . . . 33
**3 Representation Learning in Hyperbolic Space 34**
3.1 Introduction . . . . . 34
3.2 Preliminaries . . . . . 37
3.2.1 Variational Autoencoder . . . . . 37
3.2.2 Hyperbolic Space . . . . . 38
3.3 Adversarial Poincaré VAE . . . . . 40
3.3.1 Flexible Wrapped Distribution Encoder . . . . . 41
3.3.2 Poincaré Decoder . . . . . 42
3.3.3 Implementing APo-VAE . . . . . 42

3.3.4 Data-driven Prior . . . . . 44
3.4 Related Work . . . . . 45
3.5 Experiments . . . . . 46
3.5.1 Experimental Setup . . . . . 46
3.5.2 Experimental Results . . . . . 48
3.6 Conclusions . . . . . 52
3.7 Supplementary Material . . . . . 53
3.7.1 Basics of Riemannian Geometry (Extended) . . . . . 53
3.7.2 Additional Related Work . . . . . 55
3.7.3 Metrics (Dialogue Response Generation) . . . . . 55
3.7.4 Additional Implementation Details . . . . . 56
3.7.5 Additional Results . . . . . 58
**4 Contrastively Learnt Latent Representation for Domain Adaptation 61**
4.1 Introduction . . . . . 61
4.2 Related Work . . . . . 64
4.3 Approach . . . . . 65
4.3.1 Global Alignment with MMD . . . . . 67
4.3.2 Contrastively Smoothed Class Alignment . . . . . 68
4.3.3 Training Procedure . . . . . 70
4.4 Experiments . . . . . 72
4.4.1 Digit Datasets . . . . . 74
4.4.2 CIFAR-10 and STL-10 Datasets . . . . . 76
4.4.3 VisDA Dataset . . . . . 77
4.4.4 Amazon Reviews Dataset . . . . . 78

4.4.5 Ablation Study . . . . . 79

4.5 Conclusions . . . . . 81

**5 Bridging Domain Gap by Learning Intermediate Latent Representation 82**

5.1 Introduction . . . . . 82

5.2 Related Work . . . . . 84

5.3 Method . . . . . 85

5.3.1 Domain Adversarial Neural Network . . . . . 86

5.3.2 Challenge in Domain Adversarial Learning . . . . . 87

5.3.3 Adaptation with Bridging Domain . . . . . 88

5.4 Experiments . . . . . 91

5.4.1 Toy Experiment with Two Moons . . . . . 91

5.4.2 Digit Classification . . . . . 92

5.4.3 Recognizing Cars in SV Domain at Night . . . . . 93

5.4.4 Foggy Scene Segmentation . . . . . 98

5.5 Conclusions . . . . . 100

5.6 Supplementary Material . . . . . 101

5.6.1 Proof of Eq. 4.8 . . . . . 101

5.6.2 Additional Experiments . . . . . 103

5.6.3 Unsupervised Discovery of Bridging Domains . . . . . 103

**6 Conclusion 108**

**Bibliography 111**

**Biography 1**

# List of Figures


1.1 GAN Framework . . . . . 4

2.1 Characteristics of the encoder and decoder of the conventional VAE . 17

2.2 Characteristics of the Symmetric VAE . . . . . 18

2.3 sVAE results on toy dataset . . . . . 27

2.4 sVAE results on MNIST . . . . . 28

2.5 CelebA generation results . . . . . 30

2.6 CelebA reconstruction results . . . . . 30

2.7 sVAE-r and ALICE CIFAR quantitative evaluation with different values of $\lambda$ . . . . . 31

2.8 sVAE CIFAR unsupervised generation . . . . . 32

2.9 sVAE-r (with $\lambda = 1$) CIFAR unsupervised generation . . . . . 32

3.1 Illustration of the latent hierarchy in natural language . . . . . 35

3.2 Model framework of the proposed APo-VAE . . . . . 41

3.3 Visualization of the hyperbolic latent space . . . . . 50

3.4 Visualization of different mathematical operations in a hyperbolic space 54

3.5 Mapping to the hyperbolic space . . . . . 54

3.6 Visualization of the Euclidean space . . . . . 56

3.7 Visualization of the hyperbolic latent space from different datasets . . 57

4.1 Comparison between previous classifier-discrepancy-based methods and our proposed CoSCA . . . . . 62

4.2 Framework of the proposed CoSCA . . . . . 66

4.3 Sample images from the Digit, CIFAR-10 and STL datasets . . . . . 73

4.4 t-SNE embedding of the features $G(x)$ . . . . . 75

4.5 Ablation study II on CoSCA and different variations of MCD . . . . 80

5.1 Illustration of UDA with bridges . . . . . 83

5.2 Translated two moons . . . . . 87

5.3 The learning framework for our extension of DANN using multiple discriminators . . . . . 89

5.4 Sample images of CompCars surveillance (SV) domain . . . . . 94

5.5 t-SNE plots of CompCars web and SV domains . . . . . 94

5.6 Validation accuracy on SV4–5 . . . . . 96

5.7 Validation accuracy over training epochs . . . . . 103

5.8 Performance based on *supervised* bridging domain discovery using ground truth lighting conditions . . . . . 104

5.9 Performance based on *unsupervised* bridging domain discovery using discriminator score . . . . . 104

5.10 visualize images from the surveillance domain (epoch 10) . . . . . 105

5.11 visualize images from the surveillance domain (epoch 50) . . . . . 105

5.12 visualize images from the surveillance domain (epoch 150) . . . . . 105

5.13 AUC between predicted closeness to the source domain at each training epoch . . . . . 106

# List of Tables


2.1 Quantitative Results on MNIST . . . . . 29

2.2 Unsupervised Inception Score on CIFAR-10 . . . . . 32

3.1 Results on PTB, Yahoo, and Yelp datasets . . . . . 49

3.2 Unaligned style transfer on the Yelp restaurant reviews dataset . . . . 51

3.3 Results on SwitchBoard . . . . . 51

3.4 Human evaluation results . . . . . 52

3.5 Generate a sample based on the latent code . . . . . 57

3.6 Results on DailyDialog . . . . . 58

3.7 Results on forward and reverse PPL . . . . . 58

3.8 Human evaluation results on DailyDialog . . . . . 59

3.9 Examples of context-response pairs for different models . . . . . 60

4.1 Results on visual domain adaptation tasks . . . . . 74

4.2 Test accuracy of ResNet101 model fine-tuned on the VisDA dataset . 77

4.3 Results on the Amazon Reviews dataset . . . . . 78

4.4 Ablation study I on CoSCA with different variations of MCD . . . . . 79

5.1 Average classification accuracy on test set of each domain . . . . . 92

5.2 Accuracy on MNIST-M and SVHN test sets averaged over 10 runs . . 93

5.3 Accuracy and standard error over 5 runs on SV test sets . . . . . 95

5.4 Accuracy and standard error over 5 runs on SV5 test set . . . . . 96

5.5 Accuracy and standard error over 5 runs on SV4–5 test set . . . . . 98

5.6 mIoU and standard error over 5 runs on Foggy Cityscapes 0.02 test set 99

# Chapter 1

# Introduction

Deep neural networks have achieved great success in various applications, across both computer vision, and natural language processing (NLP) domains. Representation learning is often adopted to extract useful latent features for certain applications. In this dissertation, I mainly focus on two applications: 1) Deep generative models; and 2) Unsupervised domain adaptation (UDA), and demonstrate the effectiveness of different representation learning methodologies on improving the overall model performance. This chapter provides reviews of existing deep generative models and UDA models, upon which the proposed models are based. I will then discuss the contributions that I have made in these fields during my Ph.D. study.

## 1.1 Background

### 1.1.1 Variational Autoencoder

The variational autoencoder (VAE) [KW14] is based on optimizing a variational lower bound, connected to inferring an approximate posterior distribution on latent variables; such learning is typically not performed in an adversarial manner. VAEs have been demonstrated to be effective models for inferring latent variables, in that the reconstructed data do typically look like the original data. This model yields general forms of encoders and decoders, but it is based on the original variational Bayesian (VB) formulation.

Assume observed data samples $\boldsymbol{x} \sim q(\boldsymbol{x})$, where $q(\boldsymbol{x})$ is the true and unknown

distribution we wish to approximate. Consider $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$, a model with parameters $\boldsymbol{\theta}$ and latent code $\boldsymbol{z}$. With prior $p(\boldsymbol{z})$ on the codes, the modeled generative process is $\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})$, with $\boldsymbol{z} \sim p(\boldsymbol{z})$. We may marginalize out the latent codes, and hence the model is $\boldsymbol{x} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int d\boldsymbol{z} p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$. To learn $\boldsymbol{\theta}$, we typically seek to maximize the expected log likelihood: $\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}}\, \mathbb{E}_{q(\boldsymbol{x})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$, where one typically invokes the approximation $\mathbb{E}_{q(\boldsymbol{x})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx \frac{1}{N}\sum_{n=1}^{N} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_n)$ assuming $N$ iid observed samples $\{\boldsymbol{x}_n\}_{n=1,N}$.

It is typically intractable to evaluate $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ directly, as $\int d\boldsymbol{z} p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$ generally doesn't have a closed form. Consequently, a typical approach is to consider a model $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ for the posterior of the latent code $\boldsymbol{z}$ given observed $\boldsymbol{x}$, characterized by parameters $\boldsymbol{\phi}$. Distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is often termed an *encoder*, and $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ is a *decoder* [KW14]; both are here stochastic, *vis-à-vis* their deterministic counterparts associated with a traditional autoencoder [VLL+10]. Consider the variational expression

$$\mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{q(\boldsymbol{x})}\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \log \left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\right] \tag{1.1}$$

In practice the expectation wrt $\boldsymbol{x} \sim q(\boldsymbol{x})$ is evaluated via sampling, assuming $N$ observed samples $\{\boldsymbol{x}_n\}_{n=1,N}$. One typically must also utilize sampling from $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ to evaluate the corresponding expectation in (1.1). Learning is effected as $(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}}) = \text{argmax}_{\boldsymbol{\theta},\boldsymbol{\phi}}\ \mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi})$, and a model so learned is termed a variational autoencoder (VAE) [KW14].

While $\mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{q(\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) - \text{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}))] \leq \mathbb{E}_{q(\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})]$, the variational expression may be alternatively represented as

$$\mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\text{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})) + C_x \tag{1.2}$$

where $q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) = q(\boldsymbol{x})q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and $C_x = \mathbb{E}_{q(\boldsymbol{x})} \log q(\boldsymbol{x})$.

One may readily show that

$$
\begin{aligned}
& \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})) \\
= \; & \mathbb{E}_{q(\boldsymbol{x})}\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})) + \mathrm{KL}(q(\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{x})) \qquad (1.3) \\
= \; & \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})}\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})) + \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z})\|p(\boldsymbol{z})) \qquad (1.4)
\end{aligned}
$$

where $q_{\boldsymbol{\phi}}(\boldsymbol{z}) = \int q(\boldsymbol{x})q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})d\boldsymbol{x}$. To maximize $\mathcal{L}_x(\boldsymbol{\theta},\boldsymbol{\phi})$, we seek minimization of $\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{z},\boldsymbol{z}))$. Hence, from (1.3) the goal is to align $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ with $q(\boldsymbol{x})$, while from (1.4) the goal is to align $q_{\boldsymbol{\phi}}(\boldsymbol{z})$ with $p(\boldsymbol{z})$. The other terms seek to match the respective conditional distributions. All of these conditions are implied by minimizing $\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{z},\boldsymbol{z}))$.

### 1.1.2 Generative Adversarial Networks

Generative Adversarial Networks (GANs) [GPAM+14] constitute another recent framework for learning a generative model. The original idea is to build an effective generative model $p_\theta(x|z)$ with $z \sim p(z)$. Different from VAEs, there is no desire to simultaneously design an inference network $q_\phi(z|x)$, and thus GANs do not reconstruct.

GANs consist of a generator $G$ and a discriminator $D$ that compete in a two-player game. The generator is trained to generate synthetic data that look similar to the real data in order to fool the discriminator, while the discriminator wants to distinguish between the real data and the fake ones provided by the generator. The training procedure is explained in (1.5):

$$
\min_G \max_D V(D,G) = E_{x\sim q(x)} \log D(x) + E_{z\sim p(z)} \log(1 - D(G(z))) \qquad (1.5)
$$

where $q(x)$ and $p(z)$ are the distribution for the data and latent code, specifically,

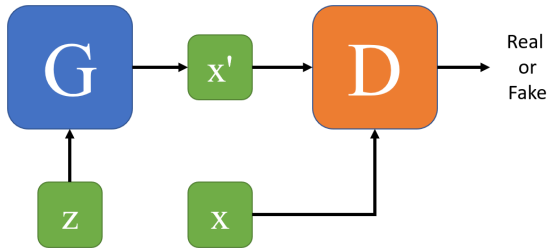



**Figure 1.1**: Discriminator $D$ and Generator $G$ are learned during the training process for a GAN.

and $G(z)$ is the proposed $p_\theta(x|z)$ as discussed previously. Fig. 1.1 shows the training procedure of GANs, where $x'$ is the synthetic data generated by $G(z)$. Note that for a fixed generator, the optimal discriminator is $D(x) = \frac{q(x)}{p_\theta(x|z)+q(x)}$. Therefore, the global equilibrium is achieved when $p_\theta(x|z) = q(x)$, in which the generator successfully fools the discriminator.

Different from VAEs, which suffers from generating blurry images, GANs is well-known for the quality of its generation. However, one issue of GANs is that often time, the synthetic images have low diversity, i.e. the mode collapse. This is because the model rarely converges to the desired global equilibrium, while settling in a sub-optimal local solution in which samples produced by the generator looks the same. This characteristic of GANs makes the model hard to be trained. Most followup works try to solve the issue of GAN by introducing more stable model architecture that provides better optimization paths.

One of the first improvements in the training of GANs when generating images is to apply the CNN architectures. This is first done by Deep convolutional GAN (DCGAN) [RMC16]. CNN allows both the generator and the discriminator to learn good up-sampling and down-sampling operations, which helps to improve the quality of the generated images. Techniques such as batch normalization is required when training deep networks in order to have more stable training. Other techniques such

as using leaky ReLU activation functions also provides better results in general.

Alternative cost functions are also applied in GANs. Wasserstein GAN [ACB17] uses earth moving distance in the cost function in order to prevent vanishing gradients in training GANs. Different from the original GANs, WGAN is more likely to provide gradients that are useful for training the generator. In the later work [GAA+17], gradient penalty is added to penalize the norm of discriminator gradients with respect to data samples during training. This solves the issue in original WGAN, in which clipping the parameters of the discriminator reduces the capacity of the model.

Other models such as Bidirectional GAN [DKD17] and Adversarially learned inference (ALI) [DBP+17] provide an inference model to allow data samples to be mapped to latent representations. This is done to improve the quality of the data representation. However, the reconstruction quality suggests that the model often do not encode and decode samples. This limitation would further discussed in the later section.

### 1.1.3 Adversarial Variational Bayes

Variational Autoencoder [KW14] represents one of the most successful deep generative models. By using the reparameterization trick, one can train VAE with stochastic gradient decent. While the original implementation of VAE requires a Gaussian assumption for the encoder, many recent works try to remove this requirement. Normalizing flow [RMea15] employs a sequence of invertible transformation to make the distribution of the latent codes arbitrarily flexible. More recently, Stein VAE [PGH+17] applies Stein variational gradient descent [LW16] to infer the distribution of latent codes, discarding the assumption of a parametric form of posterior distribution for the latent code. However, these methods do not aim to solve the fundamental limitation of ML-based methods, as they are all based on the variational formulation

in (2.1).

Adversarial Variational Bayes (AVB) [MNG16] is introduced as a technique for training VAE with arbitrarily expressive inference models, i.e. it does not require a Gaussian assumption for the encoder. This is achieved by using a discriminative network that allows to rephrase the maximum-likelihood problem as a min-max game (adversarial). This indeed establishes a principle connection between VAEs and GANs. The original paper addresses both advantages and disadvantages of VAEs and GANs: while GANs generally yield visually sharper results when applied to leaning a representation of natural images, VAEs can nicely yield both generative and inference models. However, even though VAEs may have a better log-likelihoods, the reconstructed images often vaguely resemble the input. This is due to the fact that the inference models used during training do not capture the true posterior distribution (discussed in details in the following section).

In order to solve this issue and have arbitrarily flexible inference models, AVB tries to combine VAEs and GANs in a maximum-likelihood (ML) point of view, leading to a ML assignment for the generative model with the correct posterior distribution. The adversarial manner of AVB shows up in the following proposition:

**Proposition 1.** *For $p_\theta(x|z)$ and $q_\phi(z|x)$ fixed, the optimal discriminator $f_{\psi^*}$ is given by:*

$$f_{\psi^*}(x, z) = \log q_\phi(z|x) - \log p(z) \tag{1.6}$$

*according to the objective:*

$$\max_\psi E_{q(x)} E_{q_\phi(z|x)} \log \sigma(f_\psi(x, z)) + E_{q(x)} E_{p(z)} \log(1 - \sigma(f_\psi(x, z))) \tag{1.7}$$

This can be similarly proved as Proposition 1 in [GPAM+14]. Together with the original VAE objective in (2.1), we can rewrite the optimization objective as:

$$\max_{\theta,\phi} E_{q(x)} E_{q_\phi(z|x)}(-f_{\psi^*}(x,z) + \log p_\theta(x|z)) \tag{1.8}$$

To optimize (1.8), it is required to calculate the gradients of (1.8) with respect to $\theta$ and $\phi$. Using the reparameterization trick [KW14], (1.8) can rewritten in the form:

$$\max_{\theta,\phi} E_{q(x)} E_{\epsilon}(-f_{\psi^*}(x, z_\phi(x,\epsilon)) + \log p_\theta(x|z_\phi(x,\epsilon))) \tag{1.9}$$

for a suitable function $z_\phi(x,\epsilon)$, i.e. a neural network that takes data $x$ and Gaussian noise $\epsilon$ from $\mathcal{N}(0,1)$. Together with Proposition 1, (1.9) allows us to take unbiased estimates of the gradients of (1.8) with respect to $\theta$ and $\phi$.

Generally speaking, AVB strives to optimize the same objective as VAE, but approximates the Kullback-Leibler divergence using an adversarial approach instead of relying on a closed-form formula as what is done in VAE. This helps to remove the Gaussian assumption for the encoder in the original VAE, and provides a ML assignment with a correct posterior distribution.

### 1.1.4 Domain Adversarial Neural Networks

With recent advances in supervised deep learning, many vision problems, including image recognition [KSH12, SZ15, SLJ+15, HZRS16], fine-grained classification [WBW+11, KSDFF13, SKP15], object detection [GDDM14, RHGS15, RDGF16, LGG+17, LDG+17] and semantic segmentation [SLD17, CPK+18], have met tremendous improvements. While the success is driven by several factors, such as improved deep learning architectures [HZRS16, HSS18] or optimization techniques [DHS11, KB14, IS15], it is strongly dependent on the existence of large-scale labeled training data, such as ImageNet [DDS+09]. Unfortunately, such large-scale labeled dataset may not be available for each application domain. This demands new ways of knowl-

edge transfer from existing labeled data to individual target applications, potentially with an access to large-scale *unlabeled* data from the application domain.

Unsupervised domain adaptation (UDA) [BDBCP07, BDBC+10] has been proposed to improve the generalization capability of classification models on the target domain using labeled data from the source domain and unlabeled data from the target domain. Deep domain adaptation [THZ+14, GUA+16], which realizes UDA in deep learning frameworks, has been successful in many vision applications [CLS+18, IFYA18, HWYD16, HTP+17, THS+18]. The general idea is to reduce the discrepancy metric measured by the domain discriminator [GUA+16] between the source and target domains at certain feature-level. It is successful if the discriminator learns the transformation mechanisms between the two domains well.

Domain adversarial neural network [GUA+16] learns domain-invariant features to transfer a classifier learned from labeled source domain to unlabeled target domain. This is realized by first learning the domain-related information and then leveraging it with features extracted from the input data. DANN uses a domain discriminator $d$ to control the amount of domain-related information in the extracted feature. Firstly, $d$ itself needs to be trained well so that it can tell the difference between source and target domains. The discriminator is updated by maximizing the following:

$$\mathcal{L}_d = \mathbb{E}_{x\sim\mathcal{D}_S} \log d(f(x)) + \mathbb{E}_{x\sim\mathcal{D}_T} \log(1 - d(f(x))) \tag{1.10}$$

In comparison, the feature extractor $f$ wants to confuse the discriminator $d$ to remove any domain-specific information. Moreover, to make sure the extracted feature is task-related, $f$ is trained to generate features that can be correctly classified by the classifier $C$ trained by minimizing the following:

$$\mathcal{L}_C = \mathbb{E}_{(x,y)\sim\mathcal{D}_S\times\mathcal{Y}}[-y \log C(f(x))] \tag{1.11}$$

and a learning objective for feature extractor is as follows:

$$\min_{\theta_f, \theta_C} \mathcal{L}_C + \lambda \mathcal{L}_d. \tag{1.12}$$

## 1.2 Thesis Contribution

### 1.2.1 Unifying VAE and Adversarial Training

The original Variational Bayesian framework yields a lower bound on the log likelihood of the observed data, and therefore model learning is connected to maximum-likelihood (ML) approaches. From the perspective of designing generative models, it has been recognized recently that ML-based learning has limitations [AB17]: such learning tends to yield models that match observed data, but also have a high probability of generating unrealistic synthetic data.

To alleviate such a problem, we propose a new form of the variational autoencoder (VAE), based on the *symmetric* Kullback-Leibler divergence. It is demonstrated that learning of the resulting symmetric VAE (sVAE) has close connections to previously developed adversarial-learning methods. This relationship helps unify the previously distinct techniques of VAE and adversarially learning, and provides insights that allow us to ameliorate shortcomings with some previously developed adversarial methods. In addition to an analysis that motivates and explains the sVAE, an extensive set of experiments validate the utility of the approach.

### 1.2.2 Learning Sentence Representations in a Hyperbolic Space

Natural language often exhibits inherent hierarchical structure ingrained with complex syntax and semantics. However, most state-of-the-art deep generative models learn embeddings only in Euclidean vector space, without accounting for this structural property of language. We investigate text generation in a hyperbolic latent space

to learn continuous hierarchical representations. An Adversarial Poincaré Variational Autoencoder (APo-VAE) is presented, where both the prior and variational posterior of latent variables are defined over a Poincaré ball via wrapped normal distributions. By adopting the primal-dual formulation of Kullback-Leibler divergence, an adversarial learning procedure is introduced to empower robust model training. Extensive experiments in language modeling, unaligned style transfer, and dialog-response generation demonstrate the effectiveness of the proposed APo-VAE model over VAEs in Euclidean latent space, thanks to its superb capabilities in capturing latent language hierarchies in hyperbolic space.

### 1.2.3 Adaptation via two Contrastively Trained Classifiers

Recent unsupervised approaches to domain adaptation primarily focus on minimizing the gap between the source and the target domains through refining the feature generator, in order to learn a better distribution matching between the two domains. This minimization can be achieved via a domain classifier to detect target-domain features that are divergent from source-domain features. However, by optimizing via such domain classification discrepancy, ambiguous target samples that are not smoothly distributed on the low-dimensional data manifold are often missed. To solve this issue, we propose a novel Contrastively Smoothed Class Alignment (CoSCA) model, that explicitly incorporates both intra- and inter-class domain discrepancy to better align ambiguous target samples with the source domain. CoSCA estimates the underlying label hypothesis of target samples, and simultaneously adapts their feature representations by optimizing a proposed contrastive loss. In addition, Maximum Mean Discrepancy (MMD) is utilized to directly match features between source and target samples for better global alignment. Experiments on several benchmark datasets demonstrate that CoSCA can outperform state-of-the-art approaches for

unsupervised domain adaptation by producing more discriminative features.

### 1.2.4 Tackling Large Domain Gap using Unlabeled Bridges

We tackle an unsupervised domain adaptation problem where the domain discrepancy between labeled source and unlabeled target domains is large due to many factors of inter and intra-domain variation. While deep domain adaptation methods are realized by reducing the domain discrepancy, it is difficult to minimize when domains are significantly unalike. In this work, we propose to decompose domain discrepancy into multiple but smaller, and thus easier to minimize, discrepancies by introducing unlabeled bridging domains that connect the source and target domains. We realize our proposal through an extension of the domain adversarial neural network with multiple discriminators, each of which accounts for reducing discrepancies between unlabeled (bridge, target) domains and a mix of all precedent domains including labeled source. We validate the effectiveness of our proposal on several adaptation tasks including object recognition and semantic segmentation.

# Chapter 2

# Symmetric Variational Autoencoder

## 2.1 Introduction

Generative models that are *descriptive* of data have been widely employed in statistics and machine learning. Factor models (FMs) represent one commonly used generative model [TB99], and mixtures of FMs have been employed to account for more-general data distributions [GH97a]. These models typically have latent variables (e.g., factor scores) that are inferred given observed data; the latent variables are often used for a down-stream goal, such as classification [CCL+08]. After training, such models are useful for inference tasks given subsequent observed data. However, when one draws from such models, by drawing latent variables from the prior and pushing them through the model to synthesize data, the synthetic data typically do not appear to be realistic. This suggests that while these models may be useful for analyzing observed data in terms of inferred latent variables, they are also capable of describing a large set of data that do not appear to be real.

The generative adversarial network (GAN) [GPAM+14] represents a significant recent advance toward development of generative models that are capable of synthesizing realistic data. Such models also employ latent variables, drawn from a simple distribution analogous to the aforementioned prior, and these random variables are fed through a (deep) neural network. The neural network acts as a functional transformation of the original random variables, yielding a model capable of representing sophisticated distributions. Adversarial learning discourages the network from yielding synthetic data that are unrealistic, from the perspective of a learned neural-network-based classifier. However, GANs are notoriously difficult to train,

and multiple generalizations and techniques have been developed to improve learning performance [SGZ+16], for example Wasserstein GAN (WGAN) [AB17, ACB17] and energy-based GAN (EB-GAN) [ZML17].

While the original GAN and variants were capable of synthesizing highly realistic data (e.g., images), the models lacked the ability to infer the latent variables given observed data. This limitation has been mitigated recently by methods like adversarial learned inference (ALI) [DBP+17], and related approaches. However, ALI appears to be inadequate from the standpoint of inference, in that, given observed data and associated inferred latent variables, the subsequently synthesized data often do not look particularly close to the original data.

The variational autoencoder (VAE) [KW14] is a class of generative models that precedes GAN. VAE learning is based on optimizing a variational lower bound, connected to inferring an approximate posterior distribution on latent variables; such learning is typically not performed in an adversarial manner. VAEs have been demonstrated to be effective models for inferring latent variables, in that the reconstructed data do typically look like the original data, albeit in a blurry manner [DBP+17]. The form of the VAE has been generalized recently, in terms of the adversarial variational Bayesian (AVB) framework [MNG16]. This model yields general forms of encoders and decoders, but it is based on the original variational Bayesian (VB) formulation. The original VB framework yields a lower bound on the log likelihood of the observed data, and therefore model learning is connected to maximum-likelihood (ML) approaches. From the perspective of designing generative models, it has been recognized recently that ML-based learning has limitations [AB17]: such learning tends to yield models that match observed data, but also have a high probability of generating unrealistic synthetic data.

The original VAE employs the Kullback-Leibler divergence to constitute the vari-

ational lower bound. As is well known, the KL distance metric is asymmetric. We demonstrate that this asymmetry encourages design of decoders (generators) that often yield unrealistic synthetic data when the latent variables are drawn from the prior. From a different but related perspective, the encoder infers latent variables (across all training data) that only encompass a subset of the prior. As demonstrated below, these limitations of the encoder and decoder within conventional VAE learning are intertwined.

We consequently propose a new *symmetric* VAE (sVAE), based on a symmetric form of the KL divergence and associated variational bound. The proposed sVAE is learned using an approach related to that employed in the AVB [MNG16], but in a new manner connected to the symmetric variational bound. Analysis of the sVAE demonstrates that it has close connections to ALI [DBP+17], WGAN [ACB17] and to the original GAN [GPAM+14] framework; in fact, ALI is recovered exactly, as a special case of the proposed sVAE. This provides a new and explicit linkage between the VAE (after it is made symmetric) and a wide class of adversarially trained generative models. Additionally, with this insight, we are able to ameliorate much of the aforementioned limitations of ALI, from the perspective of data reconstruction. In addition to analyzing properties of the sVAE, we demonstrate excellent performance on an extensive set of experiments.

## 2.2 Review of Variational Autoencoder

### 2.2.1 Background

Assume observed data samples $\boldsymbol{x} \sim q(\boldsymbol{x})$, where $q(\boldsymbol{x})$ is the true and unknown distribution we wish to approximate. Consider $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$, a model with parameters $\boldsymbol{\theta}$ and latent code $\boldsymbol{z}$. With prior $p(\boldsymbol{z})$ on the codes, the modeled generative process is $\boldsymbol{x} \sim p_\theta(\boldsymbol{x}|\boldsymbol{z})$, with $\boldsymbol{z} \sim p(\boldsymbol{z})$. We may marginalize out the latent codes, and hence the

model is $\boldsymbol{x} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int d\boldsymbol{z} p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$. To learn $\boldsymbol{\theta}$, we typically seek to maximize the expected log likelihood: $\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{q(\boldsymbol{x})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$, where one typically invokes the approximation $\mathbb{E}_{q(\boldsymbol{x})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx \frac{1}{N} \sum_{n=1}^{N} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_n)$ assuming $N$ iid observed samples $\{\boldsymbol{x}_n\}_{n=1,N}$.

It is typically intractable to evaluate $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ directly, as $\int d\boldsymbol{z} p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$ generally doesn't have a closed form. Consequently, a typical approach is to consider a model $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ for the posterior of the latent code $\boldsymbol{z}$ given observed $\boldsymbol{x}$, characterized by parameters $\boldsymbol{\phi}$. Distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is often termed an *encoder*, and $p_{\theta}(\boldsymbol{x}|\boldsymbol{z})$ is a *decoder* [KW14]; both are here stochastic, *vis-à-vis* their deterministic counterparts associated with a traditional autoencoder [VLL+10]. Consider the variational expression

$$\mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{q(\boldsymbol{x})} \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \log \left[ \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \right] \tag{2.1}$$

In practice the expectation wrt $\boldsymbol{x} \sim q(\boldsymbol{x})$ is evaluated via sampling, assuming $N$ observed samples $\{\boldsymbol{x}_n\}_{n=1,N}$. One typically must also utilize sampling from $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ to evaluate the corresponding expectation in (2.1). Learning is effected as $(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}}) = \text{argmax}_{\boldsymbol{\theta},\boldsymbol{\phi}} \ \mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi})$, and a model so learned is termed a variational autoencoder (VAE) [KW14].

While $\mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{q(\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) - \text{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}))] \leq \mathbb{E}_{q(\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})]$, the variational expression may alternatively be represented as

$$\mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\text{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})) + C_x \tag{2.2}$$

where $q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) = q(\boldsymbol{x}) q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z}) p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and $C_x = \mathbb{E}_{q(\boldsymbol{x})} \log q(\boldsymbol{x})$. One

may readily show that

$$
\begin{aligned}
& \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{z},\boldsymbol{z})) \\
= \; & \mathbb{E}_{q(\boldsymbol{x})}\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})) + \mathrm{KL}(q(\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{x})) \qquad (2.3) \\
= \; & \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})}\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})) + \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z})\|p(\boldsymbol{z})) \qquad (2.4)
\end{aligned}
$$

where $q_{\boldsymbol{\phi}}(\boldsymbol{z}) = \int q(\boldsymbol{x})q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})d\boldsymbol{x}$. To maximize $\mathcal{L}_x(\boldsymbol{\theta},\boldsymbol{\phi})$, we seek minimization of $\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{z},\boldsymbol{z}))$. Hence, from (2.3) the goal is to align $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ with $q(\boldsymbol{x})$, while from (2.4) the goal is to align $q_{\boldsymbol{\phi}}(\boldsymbol{z})$ with $p(\boldsymbol{z})$. The other terms seek to match the respective conditional distributions. All of these conditions are implied by minimizing $\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{z},\boldsymbol{z}))$. However, the KL divergence is *asymmetric*, which yields limitations wrt the learned model.

### 2.2.2 Limitations of the VAE

The support $\mathcal{S}^{\epsilon}_{p(\boldsymbol{z})}$ of a distribution $p(\boldsymbol{z})$ is defined as the member of the set $\{\tilde{\mathcal{S}}^{\epsilon}_{p(\boldsymbol{z})} : \int_{\tilde{\mathcal{S}}^{\epsilon}_{p(\boldsymbol{z})}} p(\boldsymbol{z})d\boldsymbol{z} = 1-\epsilon\}$ with minimum size $\|\tilde{\mathcal{S}}^{\epsilon}_{p(\boldsymbol{z})}\| \triangleq \int_{\tilde{\mathcal{S}}^{\epsilon}_{p(\boldsymbol{z})}} d\boldsymbol{z}$. We are typically interested in $\epsilon \to 0^+$. For notational convenience we replace $\mathcal{S}^{\epsilon}_{p(\boldsymbol{z})}$ with $\mathcal{S}_{p(\boldsymbol{z})}$, with the understanding $\epsilon$ is small. We also define $\mathcal{S}_{p(\boldsymbol{z})_-}$ as the largest set for which $\int_{\mathcal{S}_{p(\boldsymbol{z})_-}} p(\boldsymbol{z})d\boldsymbol{z} = \epsilon$, and hence $\int_{\mathcal{S}_{p(\boldsymbol{z})}} p(\boldsymbol{z})d\boldsymbol{z} + \int_{\mathcal{S}_{p(\boldsymbol{z})_-}} p(\boldsymbol{z})d\boldsymbol{z} = 1$. For simplicity of exposition, we assume $\mathcal{S}_{p(\boldsymbol{z})}$ and $\mathcal{S}_{p(\boldsymbol{z})_-}$ are unique; the meaning of the subsequent analysis is unaffected by this assumption.

Consider $-\mathrm{KL}(q(\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{x})) = \mathbb{E}_{q(\boldsymbol{x})}\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) - C_x$, which from (2.2) and (2.3) we seek to make large when learning $\boldsymbol{\theta}$. The following discussion borrows insights from [ACB17], although that analysis was not placed within the context of the VAE. Since $\int_{\mathcal{S}_{q(\boldsymbol{x})_-}} q(\boldsymbol{x})\log p_{\boldsymbol{\theta}}(\boldsymbol{x})d\boldsymbol{x} \approx 0$, $\mathbb{E}_{q(\boldsymbol{x})}\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx \int_{\mathcal{S}_{q(\boldsymbol{x})}} q(\boldsymbol{x})\log p_{\boldsymbol{\theta}}(\boldsymbol{x})d\boldsymbol{x}$, and $\mathcal{S}_{q(\boldsymbol{x})} = (\mathcal{S}_{q(\boldsymbol{x})} \cap \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})}) \cup (\mathcal{S}_{q(\boldsymbol{x})} \cap \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})_-})$. If $\mathcal{S}_{q(\boldsymbol{x})} \cap \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})_-} \neq \emptyset$, there is a strong (negative) penalty introduced by $\int_{\mathcal{S}_{q(\boldsymbol{x})} \cap \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})_-}} q(\boldsymbol{x})\log p_{\boldsymbol{\theta}}(\boldsymbol{x})d\boldsymbol{x}$, and therefore maximization of

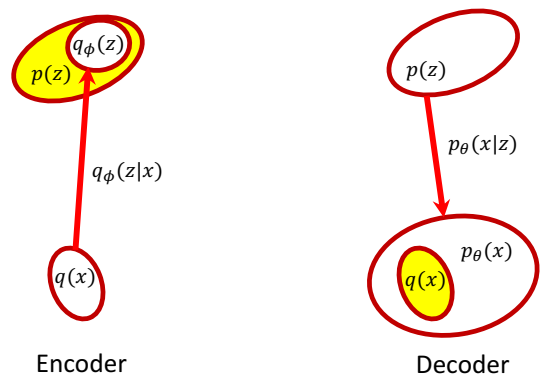



**Figure 2.1**: Characteristics of the encoder and decoder of the conventional VAE $\mathcal{L}_x$, for which the support of the distributions satisfy $\mathcal{S}_{q(\boldsymbol{x})} \subset \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})}$ and $\mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})} \subset \mathcal{S}_{p(\boldsymbol{z})}$, implying that the generative model $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ has a high probability of generating unrealistic draws.

$\mathbb{E}_{q(\boldsymbol{x})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ encourages $\mathcal{S}_{q(\boldsymbol{x})} \cap \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})_-} = \emptyset$. By contrast, there is not a substantial penalty to $\mathcal{S}_{q(\boldsymbol{x})_-} \cap \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})} \neq \emptyset$.

Summarizing these conditions, the goal of maximizing $-\mathrm{KL}(q(\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}))$ encourages $\mathcal{S}_{q(\boldsymbol{x})} \subset \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})}$. This implies that $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ can synthesize all $\boldsymbol{x}$ that may be drawn from $q(\boldsymbol{x})$, but additionally there is (often) high probability that $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ will synthesize $\boldsymbol{x}$ that will *not* be drawn from $q(\boldsymbol{x})$.

Similarly, $-\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}) \| p(\boldsymbol{z})) = h(q_{\boldsymbol{\phi}}(\boldsymbol{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})} \log p(\boldsymbol{z})$ encourages $\mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})} \subset \mathcal{S}_{p(\boldsymbol{z})}$, and the goal of increasing differential entropy $h(q_{\boldsymbol{\phi}}(\boldsymbol{z})) = -\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})} \log q_{\boldsymbol{\phi}}(\boldsymbol{z})$ encourages that $\mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})} \cap \mathcal{S}_{p(\boldsymbol{z})}$ be as large as possible.

Hence, the goal of large $-\mathrm{KL}(q(\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}))$ and $-\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}) \| p(\boldsymbol{z}))$ are saying the same thing, from different perspectives: $(i)$ seeking large $-\mathrm{KL}(q(\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}))$ implies that there is a high probability that $\boldsymbol{x}$ drawn from $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ will be different from those drawn from $q(\boldsymbol{x})$, and $(ii)$ large $-\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}) \| p(\boldsymbol{z}))$ implies that $\boldsymbol{z}$ drawn from $p(\boldsymbol{z})$ are likely to be different from those drawn from $q_{\boldsymbol{\phi}}(\boldsymbol{z})$, with $\boldsymbol{z} \in \{\mathcal{S}_{p(\boldsymbol{z})} \cap \mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})_-}\}$ responsible for the $\boldsymbol{x}$ that are inconsistent with $q(\boldsymbol{x})$. These properties are summarized in Fig. 2.1.

Considering the remaining terms in (2.3) and (2.4), and using similar logic on

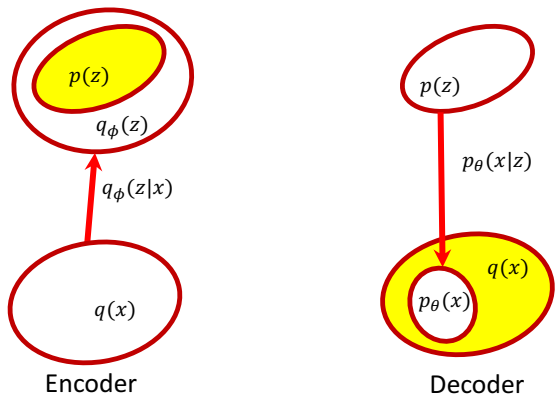



**Figure 2.2**: Characteristics of the new VAE expression, $\mathcal{L}_z$.

$-\mathbb{E}_{q(\boldsymbol{x})}\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})) = h(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})) + \mathbb{E}_{q(\boldsymbol{x})}\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\log p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$, the model encourages $\mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \subset \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})}$. From $-\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})}\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})) = h(q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})}\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})}\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$, the model also encourages $\mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})} \subset \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})}$. The differential entropies $h(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}))$ and $h(q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z}))$ encourage that $\mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \cap \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})}$ and $\mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})} \cap \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})}$ be as large as possible. Since $\mathcal{S}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \subset \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})}$, it is anticipated that $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ will under-estimate the variance of $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$, as is common with the variational approximation to the posterior [BKM17].

## 2.3 Refined VAE: Imposition of Symmetry

### 2.3.1 Symmetric KL divergence

Consider the new variational expression

$$\mathcal{L}_z(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{p(\boldsymbol{z})}\mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})}\log\left[\frac{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})q(\boldsymbol{x})}{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})}\right] \tag{2.5}$$

$$= -\mathrm{KL}(p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})\|q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})) + C_z \tag{2.6}$$

where $C_z = -h(p(\boldsymbol{z}))$. Using logic analogous to that applied to $\mathcal{L}_x$, maximization of $\mathcal{L}_z$ encourages distribution supports reflected in Fig. 2.2.

Defining $\mathcal{L}_{xz}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi}) + \mathcal{L}_z(\boldsymbol{\theta}, \boldsymbol{\phi})$, we have

$$\mathcal{L}_{xz}(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\mathrm{KL}_s(q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})) + K \tag{2.7}$$

where $K = C_x + C_z$, and the *symmetric* KL divergence is $\mathrm{KL}_s(q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})) \triangleq \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})) + \mathrm{KL}(p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) \| q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}))$. Maximization of $\mathcal{L}_{xz}(\boldsymbol{\theta}, \boldsymbol{\phi})$ seeks minimizing $\mathrm{KL}_s(q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}))$, which *simultaneously* imposes the conditions summarized in Figs. 2.1 and 2.2.

One may show that

$$\begin{aligned}
& \mathrm{KL}_s(q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})) \\
= \; & \mathbb{E}_{p(\boldsymbol{z})} \mathrm{KL}(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) \| q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z})} \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{z}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})) \\
& + \mathrm{KL}_s(p(\boldsymbol{z}) \| q_{\boldsymbol{\phi}}(\boldsymbol{z})) \qquad (2.8) \\
= \; & \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})} \mathrm{KL}(p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) \| q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})) + \mathbb{E}_{q(\boldsymbol{x})} \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})) \\
& + \mathrm{KL}_s(p_{\boldsymbol{\theta}}(\boldsymbol{x}) \| q(\boldsymbol{x})) \qquad (2.9)
\end{aligned}$$

Considering the representation in (2.9), the goal of small $\mathrm{KL}_s(p_{\boldsymbol{\theta}}(\boldsymbol{x}) \| q(\boldsymbol{x}))$ encourages $\mathcal{S}_{q(\boldsymbol{x})} \subset \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})}$ *and* $\mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})} \subset \mathcal{S}_{q(\boldsymbol{x})}$, and hence that $\mathcal{S}_{q(\boldsymbol{x})} = \mathcal{S}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})}$. Further, since $-\mathrm{KL}_s(p_{\boldsymbol{\theta}}(\boldsymbol{x}) \| q(\boldsymbol{x})) = \mathbb{E}_{q(\boldsymbol{x})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) + \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x})} \log q(\boldsymbol{x}) + h(p_{\boldsymbol{\theta}}(\boldsymbol{x})) - C_x$, maximization of $-\mathrm{KL}_s(p_{\boldsymbol{\theta}}(\boldsymbol{x}) \| q(\boldsymbol{x}))$ seeks to minimize the cross-entropy between $q(\boldsymbol{x})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{x})$, encouraging a complete matching of the distributions $q(\boldsymbol{x})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{x})$, not just shared support. From (2.8), a match is simultaneously encouraged between $p(\boldsymbol{z})$ and $q_{\boldsymbol{\phi}}(\boldsymbol{z})$. Further, the respective conditional distributions are also encouraged to match.

### 2.3.2 Adversarial solution

Assuming fixed $(\boldsymbol{\theta}, \boldsymbol{\phi})$, and using logic analogous to Proposition 1 in [MNG16], we consider

$$g(\boldsymbol{\psi}) = \mathbb{E}_{p_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})} \log(1 - \sigma(f_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z})) + \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})} \log \sigma(f_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z})) \tag{2.10}$$

where $\sigma(\zeta) = 1/(1+\exp(-\zeta))$. The scalar function $f_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z})$ is represented by a deep neural network with parameters $\boldsymbol{\psi}$, and network inputs $(\boldsymbol{x}, \boldsymbol{z})$. For fixed $(\boldsymbol{\theta}, \boldsymbol{\phi})$, the parameters $\boldsymbol{\psi}^*$ that maximize $g(\boldsymbol{\psi})$ yield

$$f_{\boldsymbol{\psi}^*}(\boldsymbol{x}, \boldsymbol{z}) = \log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \tag{2.11}$$

and hence

$$\mathcal{L}_x(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x}, \boldsymbol{z}) + C_x \tag{2.12}$$

$$\mathcal{L}_z(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x}, \boldsymbol{z}) + C_z \tag{2.13}$$

Hence, to optimize $\mathcal{L}_{xz}(\boldsymbol{\theta}, \boldsymbol{\phi})$ we consider the cost function

$$\ell(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{\psi}^*) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x}, \boldsymbol{z}) - \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x}, \boldsymbol{z}) \tag{2.14}$$

Assuming (2.11) holds, we have

$$\ell(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{\psi}^*) = -\mathrm{KL}_s(q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})) \leq 0 \tag{2.15}$$

and the goal is to achieve $\ell(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{\psi}^*) = 0$ through joint optimization of $(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{\psi}^*)$. Model learning consists of alternating between (2.10) and (2.14), maximizing (2.10) wrt $\boldsymbol{\psi}$ with $(\boldsymbol{\theta}, \boldsymbol{\phi})$ fixed, and maximizing (2.14) wrt $(\boldsymbol{\theta}, \boldsymbol{\phi})$ with $\boldsymbol{\psi}$ fixed.

The expectations in (2.10) and (2.14) are approximated by averaging over samples, and therefore to implement this solution we need only be able to sample from $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, and we do not require explicit forms for these distributions. For example, a draw from $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ may be constituted as $\boldsymbol{z} = h_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{\epsilon})$, where $h_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{\epsilon})$ is implemented as a neural network with parameters $\boldsymbol{\phi}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.

### 2.3.3 Interpretation in terms of LRT statistic

In (2.10) a classifier is designed to distinguish between samples $(\boldsymbol{x}, \boldsymbol{z})$ drawn from $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z}) p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and from $q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) = q(\boldsymbol{x}) q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$. Implicit in that ex-

pression is that there is equal probability that either of these distributions are selected for drawing $(\boldsymbol{x}, \boldsymbol{z})$, i.e., that $(\boldsymbol{x}, \boldsymbol{z}) \sim [p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) + q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})]/2$. Under this assumption, given observed $(\boldsymbol{x}, \boldsymbol{z})$, the probability of it being drawn from $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$ is $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})/(p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) + q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}))$, and the probability of it being drawn from $q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})$ is $q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})/(p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) + q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}))$ [GPAM+14]. Since the denominator $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) + q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})$ is shared by these distributions, and assuming function $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})/q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})$ is known, an observed $(\boldsymbol{x}, \boldsymbol{z})$ is inferred as being drawn from the underlying distributions as

$$\text{if } p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})/q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) > 1, \quad (\boldsymbol{x}, \boldsymbol{z}) \rightarrow p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) \tag{2.16}$$

$$\text{if } p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})/q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) < 1, \quad (\boldsymbol{x}, \boldsymbol{z}) \rightarrow q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z}) \tag{2.17}$$

This is the well-known likelihood ratio test (LRT) [Tre01], and is reflected by (2.11). We have therefore derived a learning procedure based on the log-LRT, as reflected in (2.14). The solution is "adversarial," in the sense that when optimizing $(\boldsymbol{\theta}, \boldsymbol{\phi})$ the objective in (2.14) seeks to "fool" the LRT test statistic, while for fixed $(\boldsymbol{\theta}, \boldsymbol{\phi})$ maximization of (2.10) wrt $\boldsymbol{\psi}$ corresponds to updating the LRT. This adversarial solution comes as a natural consequence of symmetrizing the traditional VAE learning procedure.

## 2.4 Connections to Prior Work

### 2.4.1 Adversarially Learned Inference

The adversarially learned inference (ALI) [DBP+17] framework seeks to learn both an encoder and decoder, like the approach proposed above, and is based on optimizing

$$\begin{aligned}(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\phi}}) \;=\; & \operatorname{argmin}_{\boldsymbol{\theta}, \boldsymbol{\phi}} \max_{\boldsymbol{\psi}} \{\mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})} \log \sigma(f_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z})) \\ & + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{z})} \log(1 - \sigma(f_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z})))\}\end{aligned} \tag{2.18}$$

This has similarities to our approach, where the term $\max_{\boldsymbol{\psi}} \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})} \log \sigma(f_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})} \log(1-\sigma(f_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})))$ is identical to our maximization of (2.10) wrt $\boldsymbol{\psi}$. However, in the proposed approach, rather than directly then optimizing wrt $(\boldsymbol{\theta},\boldsymbol{\phi})$, as in (2.18), in (2.14) the result from this term is used to define $f_{\boldsymbol{\psi}^*}(\boldsymbol{x},\boldsymbol{z})$, which is then employed in (2.14) to subsequently optimize over $(\boldsymbol{\theta},\boldsymbol{\phi})$.

Note that $\log \sigma(\cdot)$ is a monotonically increasing function, and therefore we may replace (2.14) as

$$\begin{aligned} \ell'(\boldsymbol{\theta},\boldsymbol{\phi};\boldsymbol{\psi}^*) &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})} \log \sigma(f_{\boldsymbol{\psi}^*}(\boldsymbol{x},\boldsymbol{z})) \\ &\quad + \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})} \log \sigma(-f_{\boldsymbol{\psi}^*}(\boldsymbol{x},\boldsymbol{z})) \end{aligned} \tag{2.19}$$

and note $\sigma(-f_{\boldsymbol{\psi}^*}(\boldsymbol{x},\boldsymbol{z};\boldsymbol{\theta},\boldsymbol{\phi})) = 1-\sigma(f_{\boldsymbol{\psi}^*}(\boldsymbol{x},\boldsymbol{z};\boldsymbol{\theta},\boldsymbol{\phi}))$. Maximizing (2.19) wrt $(\boldsymbol{\theta},\boldsymbol{\phi})$ with fixed $\boldsymbol{\psi}^*$ corresponds to the minimization wrt $(\boldsymbol{\theta},\boldsymbol{\phi})$ reflected in (2.18). Hence, the proposed approach is *exactly* ALI, if in (2.14) we replace $\pm f_{\boldsymbol{\psi}^*}$ with $\log \sigma(\pm f_{\boldsymbol{\psi}^*})$.

### 2.4.2 Original GAN

The proposed approach assumed both a decoder $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and an encoder $p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, and we considered the symmetric $\mathrm{KL}_s(q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})\|p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z}))$. We now simplify the model for the case in which we only have a decoder, and the synthesized data are drawn $\boldsymbol{x} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ with $\boldsymbol{z} \sim p(\boldsymbol{z})$, and we wish to learn $\boldsymbol{\theta}$ such that data synthesized in this manner match observed data $\boldsymbol{x} \sim q(\boldsymbol{x})$. Consider the symmetric

$$\begin{aligned} \mathrm{KL}_s(q(\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{x})) &= \mathbb{E}_{p(\boldsymbol{z})}\mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x}) \\ &\quad - \mathbb{E}_{q(\boldsymbol{x})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x}) \end{aligned} \tag{2.20}$$

where for fixed $\boldsymbol{\theta}$

$$f_{\boldsymbol{\psi}^*}(\boldsymbol{x}) = \log(p_{\boldsymbol{\theta}}(\boldsymbol{x})/q(\boldsymbol{x})) \tag{2.21}$$

We consider a simplified form of (2.10), specifically

$$\begin{aligned} g(\boldsymbol{\psi}) &= \mathbb{E}_{p(\boldsymbol{z})}\mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})} \log \sigma(f_{\boldsymbol{\psi}}(\boldsymbol{x})) \\ &\quad + \mathbb{E}_{q(\boldsymbol{x})} \log(1 - \sigma(f_{\boldsymbol{\psi}}(\boldsymbol{x})) \end{aligned} \tag{2.22}$$

which we seek to maximize wrt $\boldsymbol{\psi}$ with fixed $\boldsymbol{\theta}$, with optimal solution as in (2.21). We optimize $\boldsymbol{\theta}$ seeking to maximize $-\mathrm{KL}_s(q(\boldsymbol{x})\|p_{\boldsymbol{\theta}}(\boldsymbol{x}))$, as $\mathrm{argmax}_{\boldsymbol{\theta}}\ \ell(\boldsymbol{\theta};\boldsymbol{\psi}^*)$ where

$$\ell(\boldsymbol{\theta};\boldsymbol{\psi}^*) = \mathbb{E}_{q(\boldsymbol{x})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x}) - \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x}) \tag{2.23}$$

with $\mathbb{E}_{q(\boldsymbol{x})} f_{\boldsymbol{\psi}^*}(\boldsymbol{x})$ independent of the update parameter $\boldsymbol{\theta}$. We observe that in seeking to maximize $\ell(\boldsymbol{\theta};\boldsymbol{\psi}^*)$, parameters $\boldsymbol{\theta}$ are updated as to "fool" the log-LRT $\log[q(\boldsymbol{x})/p_{\boldsymbol{\theta}}(\boldsymbol{x})]$. Learning consists of iteratively updating $\boldsymbol{\psi}$ by maximizing $g(\boldsymbol{\psi})$ and updating $\boldsymbol{\theta}$ by maximizing $\ell(\boldsymbol{\theta};\boldsymbol{\psi}^*)$.

Recall that $\log \sigma(\cdot)$ is a monotonically increasing function, and therefore we may replace (2.23) as

$$\ell'(\boldsymbol{\theta};\boldsymbol{\psi}^*) = \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})} \log \sigma(-f_{\boldsymbol{\psi}^*}(\boldsymbol{x})) \tag{2.24}$$

Using the same logic as discussed above in the context of ALI, maximizing $\ell'(\boldsymbol{\theta};\boldsymbol{\psi}^*)$ wrt $\boldsymbol{\theta}$ may be replaced by minimization, by transforming $\sigma(\mu) \to \sigma(-\mu)$. With this simple modification, minimizing the modified (2.24) wrt $\boldsymbol{\theta}$ and maximizing (2.22) wrt $\boldsymbol{\psi}$, we *exactly* recover the original GAN [GPAM+14], for the special (but common) case of a sigmoidal discriminator.

### 2.4.3 Wasserstein GAN

The Wasserstein GAN (WGAN) [ACB17] setup is represented as

$$\boldsymbol{\theta} = \mathrm{argmin}_{\boldsymbol{\theta}} \max_{\boldsymbol{\psi}} \{\mathbb{E}_{q(\boldsymbol{x})} f_{\boldsymbol{\psi}}(\boldsymbol{x}) - \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})} f_{\boldsymbol{\psi}}(\boldsymbol{x})\} \tag{2.25}$$

where $f_{\boldsymbol{\psi}}(\boldsymbol{x})$ must be a 1-Lipschitz function. Typically $f_{\boldsymbol{\psi}}(\boldsymbol{x})$ is represented by a neural network with parameters $\boldsymbol{\psi}$, with parameter clipping or $\ell_2$ regularization on

the weights (to constrain the amplitude of $f_{\boldsymbol{\psi}}(\boldsymbol{x})$). Note that WGAN is closely related to (2.23), but in WGAN $f_{\boldsymbol{\psi}}(\boldsymbol{x})$ doesn't make an explicit connection to the underlying likelihood ratio, as in (2.21).

It is believed that the current paper is the first to consider symmetric variational learning, introducing $\mathcal{L}_z$, from which we have made explicit connections to previously developed adversarial-learning methods. Previous efforts have been made to match $q_{\boldsymbol{\phi}}(\boldsymbol{z})$ to $p(\boldsymbol{z})$, which is a consequence of the proposed symmetric VAE (sVAE). For example, [MSJ+15] introduced a modification to the original VAE formulation, but it loses connection to the variational lower bound [MNG16].

### 2.4.4 Amelioration of vanishing gradients

As discussed in [ACB17], a key distinction between the WGAN framework in (2.25) and the original GAN [GPAM+14] is that the latter uses a binary discriminator to distinguish real and synthesized data; the $f_{\boldsymbol{\psi}}(\boldsymbol{x})$ in WGAN is a 1-Lipschitz function, rather than an explicit discriminator. A challenge with GAN is that as the discriminator gets better at distinguishing real and synthetic data, the gradients wrt the discriminator parameters vanish, and learning is undermined. The WGAN was designed to ameliorate this problem [ACB17].

From the discussion in Section 2.4.1, we note that the key distinction between the proposed sVAE and ALI is that the latter uses a binary discriminator to distinguish $(\boldsymbol{x}, \boldsymbol{z})$ manifested via the generator from $(\boldsymbol{x}, \boldsymbol{z})$ manifested via the encoder. By contrast, the sVAE uses a log-LRT, rather than a binary classifier, with it inferred in an adversarial manner. ALI is therefore undermined by vanishing gradients as the binary discriminator gets better, with this avoided by sVAE. The sVAE brings the same intuition associated with WGAN (addressing vanishing gradients) to a generalized VAE framework, with a generator *and* a decoder; WGAN only considers a

generator. Further, as discussed in Section 2.4.3, unlike WGAN, which requires gradient clipping or other forms of regularization to approximate 1-Lipschitz functions, in the proposed sVAE the $f_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{z})$ arises naturally from the symmetrized VAE and we do not require imposition of Lipschitz conditions. As discussed in Section 2.6, this simplification has yielded robustness in implementation.

## 2.5 Model Augmentation

A significant limitation of the original ALI setup is an inability to accurately reconstruct observed data via the process $\boldsymbol{x} \to \boldsymbol{z} \to \hat{\boldsymbol{x}}$ [DBP+17]. With the proposed sVAE, which is intimately connected to ALI, we may readily address this shortcoming. The variational expressions may be written as $\mathcal{L}_x = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) - \mathbb{E}_{q(\boldsymbol{x})}\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\|p(\boldsymbol{z}))$ and $\mathcal{L}_z = \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})} \log p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) - \mathbb{E}_{p(\boldsymbol{z})}\mathrm{KL}(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\|q(\boldsymbol{x}))$. In both of these expressions, the first term to the right of the equality enforces model fit, and the second term penalizes the posterior distribution *for individual data samples* for being dissimilar from the prior (i.e., penalizes $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ from being dissimilar from $p(\boldsymbol{z})$, and likewise wrt $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and $q(\boldsymbol{x})$). The proposed sVAE encourages the *cumulative* distributions $q_{\boldsymbol{\phi}}(\boldsymbol{z})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ to match $p(\boldsymbol{z})$ and $q(\boldsymbol{x})$, respectively. By simultaneously encouraging more peaked $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$, we anticipate better "cycle consistency" [ZPIE17] and hence more accurate reconstructions.

To encourage $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ that are more peaked in the space of $\boldsymbol{z}$ for individual $\boldsymbol{x}$, and also to consider more peaked $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$, we may augment the variational expressions

as

$$\begin{aligned}\mathcal{L}'_x &= (\lambda+1)\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})}\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) \\ &\quad -\mathbb{E}_{q(\boldsymbol{x})}\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\|p(\boldsymbol{z})) \qquad (2.26)\\ \mathcal{L}'_z &= (\lambda+1)\mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}\log p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \\ &\quad -\mathbb{E}_{p(\boldsymbol{z})}\mathrm{KL}(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\|q(\boldsymbol{x})) \qquad (2.27)\end{aligned}$$

where $\lambda \geq 0$. For $\lambda = 0$ the original variational expressions are retained, and for $\lambda > 0$, $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ are allowed to diverge more from $p(\boldsymbol{z})$ and $q(\boldsymbol{x})$, respectively, while placing more emphasis on the data-fit terms. Defining $\mathcal{L}'_{xz} = \mathcal{L}'_x + \mathcal{L}'_z$, we have

$$\begin{aligned}\mathcal{L}'_{xz} = \mathcal{L}_{xz} \quad + \quad & \lambda[\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})}\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) \\ & + \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}\log p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})] \qquad (2.28)\end{aligned}$$

Model learning is the same as discussed in Sec. 2.3.2, with the modification

$$\begin{aligned}\ell'(\boldsymbol{\theta},\boldsymbol{\phi};\boldsymbol{\psi}^*) = \quad & \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x},\boldsymbol{z})}[f_{\boldsymbol{\psi}^*}(\boldsymbol{x},\boldsymbol{z}) + \lambda\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] \\ - \quad & \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}[f_{\boldsymbol{\psi}^*}(\boldsymbol{x},\boldsymbol{z}) - \lambda\log p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})] \qquad (2.29)\end{aligned}$$

A disadvantage of this approach is that it requires explicit forms for $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and $p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, while the setup in Sec. 2.3.2 only requires the ability to sample from these distributions.

We can now make a connection to additional related work, particularly [PWH+17], which considered a similar setup to (2.26) and (2.27), for the special case of $\lambda = 1$. While [PWH+17] had a similar idea of using a symmetrized VAE, they didn't make the theoretical justification presented in Section 2.3. Further, and more importantly, the way in which learning was performed in [PWH+17] is distinct from that applied here, in that [PWH+17] required an additional adversarial learning step, increasing implementation complexity. Consequently, [PWH+17] did not use adversarial learning to approximate the log-LRT, and therefore it cannot make the explicit connection to ALI and WGAN that were made in Sections 2.4.1 and 2.4.3, respectively.

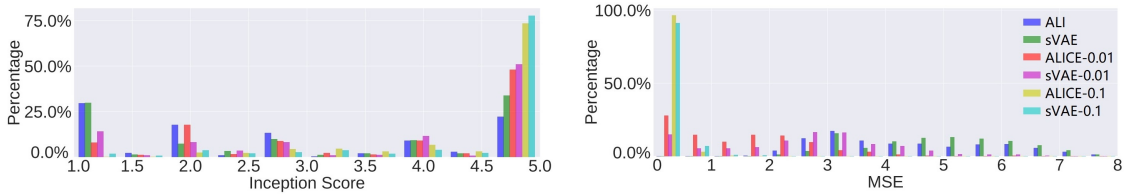



**Figure 2.3**: sVAE results on toy dataset. Top: Inception Score for ALI and sVAE with $\lambda = 0, 0.01, 0.1$. Bottom: Mean Squared Error (MSE).

## 2.6 Experiments

In addition to evaluating our model on a toy dataset, we consider MNIST, CelebA and CIFAR-10 for both reconstruction and generation tasks. As done for the model ALI with Cross Entropy regularization (ALICE) [LLC+17], we also add the augmentation term ($\lambda > 0$ as discussed in Sec. 2.5) to sVAE as a regularizer, and denote the new model as sVAE-r. More specifically, we show the results based on the two models: *i*) sVAE: the model is developed in Sec. 2.3 to optimize $g(\boldsymbol{\psi})$ in (2.10) and $\ell(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{\psi}^*)$ in (2.14). *ii*) sVAE-r: the model is sVAE with regularization term to optimize $g(\boldsymbol{\psi})$ in (2.10) and $\ell'(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{\psi}^*)$ in (2.29). The quantitative evaluation is based on the mean square error (MSE) of reconstructions, log-likelihood calculated via the annealed importance sampling (AIS) [WBSG16], and inception score (IS) [SGZ+16].

All parameters are initialized with Xavier [GB10] and optimized using Adam [KB15] with learning rate of 0.0001. No dataset-specific tuning or regularization, other than dropout [SHK+14], is performed. The architectures for the encoder, decoder and discriminator are detailed in the Appendix. All experimental results were performed on a single NVIDIA TITAN X GPU.

### 2.6.1 Toy Data

In order to show the robustness and stability of our model, we test sVAE and sVAE-r on a toy dataset designed in the same manner as the one in ALICE [LLC+17]. In

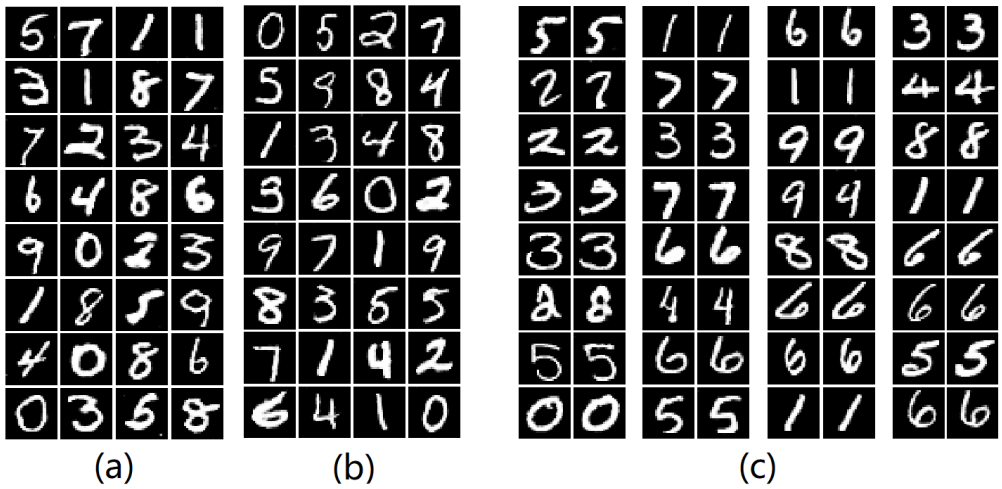



**Figure 2.4**: sVAE results on MNIST. (a) Reconstructed images. In each block: column one for ground-truth, and column two for reconstructed images by sVAE. (b) Generated sample images by sVAE. Note that $\lambda$ is set to 0.1

this dataset, the true distribution of data $\boldsymbol{x}$ is a two-dimensional Gaussian mixture model with five components. The latent code $\boldsymbol{z}$ is a standard Gaussian distribution $\mathcal{N}(0, 1)$. To perform the test, we consider using different values of $\lambda$ for both sVAE-r and ALICE. For each $\lambda$, 576 experiments with different choices of architecture and hyper-parameters are conducted. In all experiments, we use mean square error (MSE) and inception score (IS) to evaluate the performance of the two models. Figure 2.3 shows the histogram results for each model. As we can see, both ALICE and sVAE-r are able to reconstruct images when $\lambda = 0.1$, while sVAE-r provides better overall inception score.

### 2.6.2 MNIST

The results of image generation and reconstruction for sVAE, as applied to the MNIST dataset, are shown in Figure 2.4. By adding the regularization term, sVAE overcomes the limitation of image reconstruction in ALI. The log-likelihood of sVAE shown in Table 2.1 is calculated using the annealed importance sampling method on

the binarized MNIST dataset, as proposed in [WBSG16]. Our model achieves 79.3 nats, outperforming normalizing flow (85.1 nats) while also being competitive to the state-of-art (79.2 nats). In addition, sVAE is able to provide compelling generated images, outperforming GAN [GPAM+14] and WGAN-GP [GAA+17] based on the inception scores.

**Table 2.1**: Quantitative Results on MNIST

| **Model** | $\log p(x) \geq$ | IS |
|---|---|---|
| NF (k=80) [RMea15] | -85.1 | - |
| AVB [MNG16] | -79.5 | - |
| PixelRNN [OKK16] | -79.2 | - |
| GAN [GPAM+14] | - | 8.34 ± .03 |
| WGAN-GP [GAA+17] | - | 8.45 ± .04 |
| **sVAE-r (ours)** | -79.3 | 9.29 ± .06 |

### 2.6.3 CelebA

We evaluate sVAE on the CelebA dataset and compare the results with ALI. In experiments we noted that for high-dimensional data like the CelebA, ALICE [LLC+17] shows a trade-off between reconstruction and generation, while sVAE-r does not have this issue. If the regularization term is not included in ALI, the reconstructed images do not match the original images. On the other hand, when the regularization term is added, ALI is capable of reconstructing images but the generated images are flawed. In comparison, sVAE-r does well in both generation and reconstruction with different values of $\lambda$. The results for both sVAE and ALI are shown in Figure 2.5 and 2.6.

Generally speaking, adding the augmentation term as shown in (2.28) should encourage more peaked $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$. Nevertheless, ALICE fails in the inference process and performs more like an autoencoder. This is due to the fact that the dis-

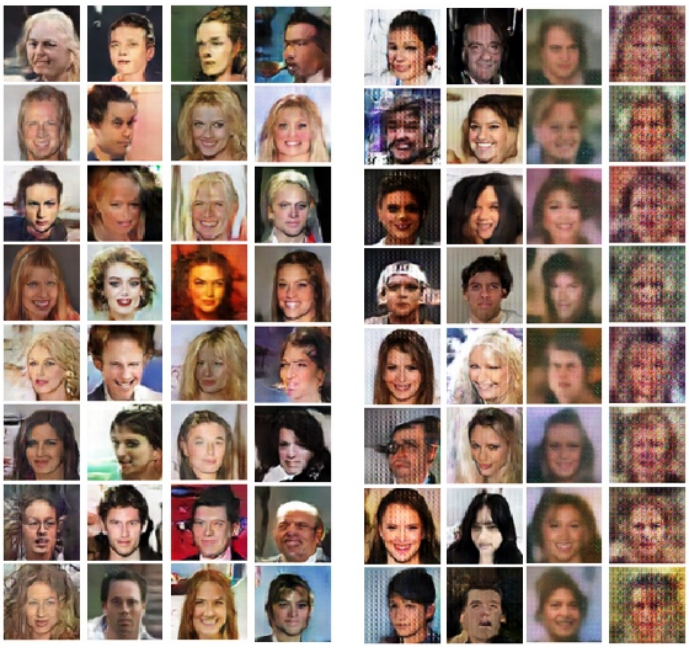


**Figure 2.5**: CelebA generation results. Left block: sVAE-r generation. Right block: ALICE generation. $\lambda = 0, 0.1, 1$ and 10 from left to right in each block.

**Figure 2.6**: CelebA reconstruction results. Left column: The ground truth. Middle block: sVAE-r reconstruction. Right block: ALICE reconstruction. $\lambda = 0, 0.1, 1$ and 10 from left to right in each block.

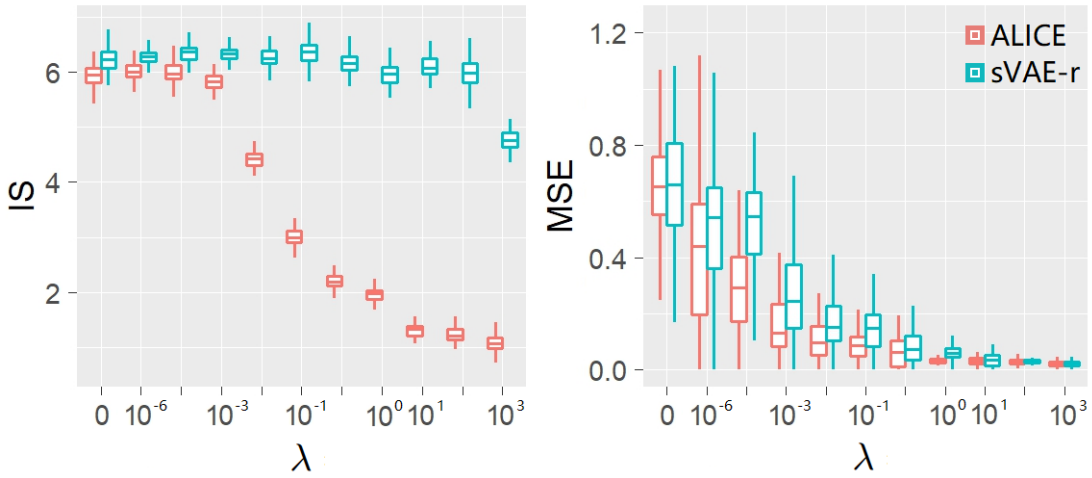



**Figure 2.7**: sVAE-r and ALICE CIFAR quantitative evaluation with different values of $\lambda$. Left: IS for generation; Right: MSE for reconstruction. The result is the average of multiple tests

criminator becomes too sensitive to the regularization term. On the other hand, by using the symmetric KL (2.14) as the cost function, we are able to alleviate this issue, which makes sVAE-r a more stable model than ALICE.

### 2.6.4 CIFAR-10

The trade-off of ALICE [LLC+17] mentioned in Sec. 2.6.3 is also manifested in the results for the CIFAR-10 dataset. In Figure 2.7, we show quantitative results in terms of inception score and mean squared error of sVAE-r and ALICE with different values of $\lambda$. As can be seen, both models are able to reconstruct images when $\lambda$ increases. However, when $\lambda$ is larger than $10^{-3}$, we observe a decrease in the inception score of ALICE, in which the model fails to generate images.

The CIFAR-10 dataset is also used to evaluate the generation ability of our model. The quantitative results, i.e., the inception scores, are listed in Table 2.2. Our model shows improved performance on image generation compared to ALI and DCGAN. Note that sVAE also gets comparable result as WGAN-GP [GAA+17] achieves. This

**Table 2.2**: Unsupervised Inception Score on CIFAR-10

| **Model** | IS |
|---|---|
| ALI [DBP+17] | 5.34 ± .05 |
| DCGAN [RMC16] | 6.16 ± .07 |
| WGAN-GP | 6.56 ± .05 |
| WGAN-GP ResNet [GAA+17] | 7.86 ± .07 |
| **sVAE (ours)** | 6.76 ± .046 |
| **sVAE-r (ours)** | 6.96 ± .066 |

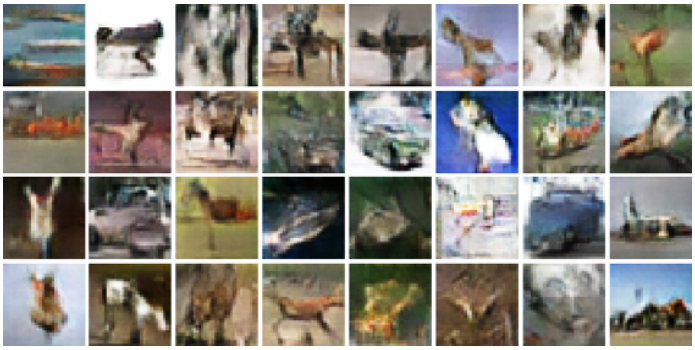


**Figure 2.8**: sVAE CIFAR unsupervised generation.

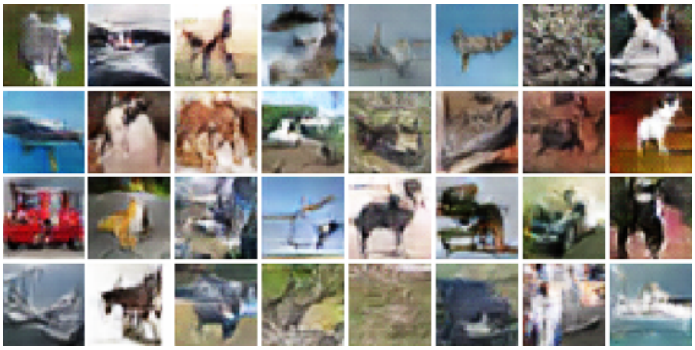


**Figure 2.9**: sVAE-r (with $\lambda = 1$) CIFAR unsupervised generation.

can be interpreted using the similarity between (2.23) and (2.25) as summarized in the Sec. 2.4. The generated images are shown in Figures 2.8 and 2.9. The reconstruction images and more generated images are in the Appendix.

## 2.7 Conclusions

We present the symmetric variational autoencoder (sVAE), a novel framework which can match the joint distribution of data and latent code using the *symmetric* Kullback-Leibler divergence. The experiment results show the advantages of sVAE, in that it not only overcomes the missing mode problem, but also is very stable to train. With excellent performance in image generation and reconstruction, we will apply sVAE on semi-supervised learning tasks in future work.

# Chapter 3

# Representation Learning in Hyperbolic Space

## 3.1 Introduction

The Variational Autoencoder (VAE) [KW13, RMW14] is a generative model widely applied to language-generation tasks, which propagates latent codes drawn from a simple prior to manifest data samples through a decoder. The generative model is augmented by an inference network, which feeds observed data samples through an encoder to yield a distribution on the corresponding latent codes. Since natural language often manifests a latent hierarchical structure, it is desirable for the latent code in a VAE to reflect such inherent language structure, so that the generated text can be more natural and expressive. An example of language structure is illustrated in Figure 3.1, where sentences are organized into a tree structure. The root node corresponds to simple sentences (*e.g.*, "*Yes*"), while nodes on outer leaves represent sentences with more complex syntactic structure and richer, more specific semantic meaning (*e.g.*, "*The food in the restaurant is awesome*")[1].

In existing VAE-based generative models, such structures are not *explicitly* considered. The latent code often employs a simple Gaussian prior, and the posterior is approximated as a Gaussian with diagonal covariance matrix. Such embeddings assume Euclidean structure, which is inadequate in capturing geometric structure illustrated in Figure 3.1. While some variants have been proposed to enrich the prior distributions [XD18, WGX+19, WTG+19, SZM+19], there is no evidence that

[1]Another possible way to organize sentences is a hierarchy of topics, *e.g.*, a parent node can be a sentence on "*sports*", while its children are sentences on "*basketball*" and "*skiing*".

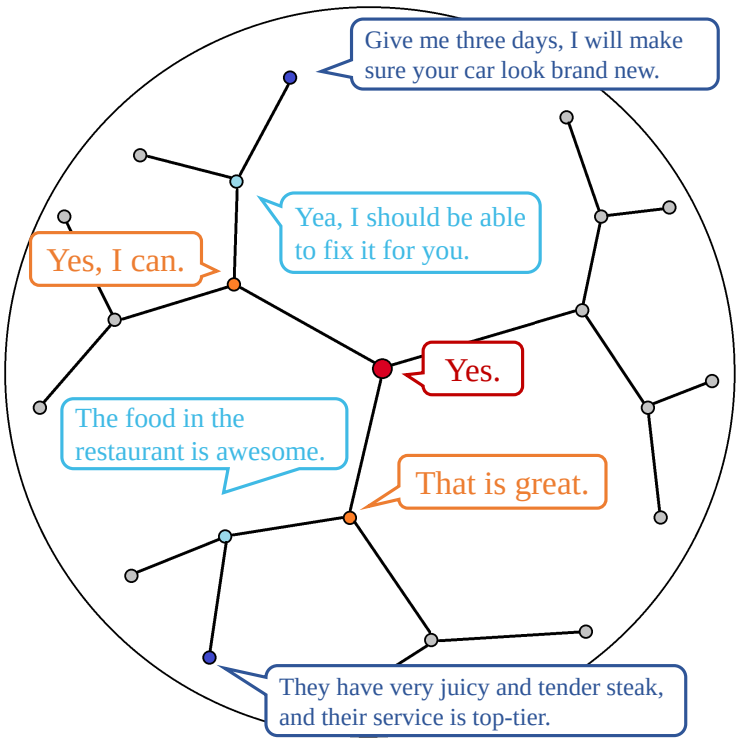



**Figure 3.1**: Illustration of the latent hierarchy in natural language. Each tree node is a latent code of its corresponding sentence.

structural information in language can be recovered effectively by the model.

Hyperbolic geometry has recently emerged as an effective method for representation learning from data with hierarchical structure [MLM+19, NK17]. Informally, hyperbolic space can be considered as a continuous map of trees. For example, a Poincaré disk (a hyperbolic space with two dimensions) can represent any tree with arbitrary low distortion [DSGRS18, Sar11]. In Euclidean space, however, it is difficult to learn such structural representation even with infinite dimensions [LLR95].

Motivated by these observations, we propose Adversarial Poincaré Variational Autoencoder (APo-VAE), a text embedding and generation model based on hyperbolic representations, where the latent code is encouraged to capture the underlying tree-like structure in language. Such latent structure provides more control of the generated sentences, *i.e.*, an increase of sentence complexity and diversity can be

achieved along some trajectory from a root to its children. In practice, we define both the prior and the variational posterior of the latent code over a Poincaré ball, via the use of a wrapped normal distribution [NYFK19b]. To obtain more stable model training and learn more flexible representation of the latent code, we exploit the primal-dual formulation of Kullback-Leibler (KL) divergence [DDH+18] based on the Fenchel duality [R+66], to adversarially optimize the variational bound. Unlike the primal form that relies on Monte Carlo approximation [MLM+19], our dual formulation bypasses the need for tractable posterior likelihoods via the introduction of an auxiliary dual function.

We apply the proposed approach to language modeling, unaligned style transfer and dialog-response generation. For language modeling, in order to enhance the distribution complexity of the prior, we use an additional "variational mixture of posteriors" prior (VampPrior) design [TW18] for the wrapped normal distribution. Specifically, VampPrior uses a mixture distribution with components from variational posteriors, coupling the parameters of the prior and variational posterior. For unaligned style transfer, we add a sentiment classifier to our model, and disentangle content and sentiment information by using adversarial training [ZKZ+17]. For dialog-response generation, a conditional model variant of APo-VAE is designed to take into account the dialog context.

Experiments also show that the proposed model addresses *posterior collapse* [BVV+16], a major obstacle preventing efficient learning of a VAE on text data. In posterior collapse, the encoder learns an approximate posterior similar to the prior, and the decoder tends to ignore the latent code for generation. Experiments show that our proposed model can effectively avoid posterior collapse. We hypothesize that this is due to the use of a more informative prior in hyperbolic space that enhances the complexity of the latent representation, which aligns well with previous

work [TW18, WGX+19] that advocates a better prior design.

Our main contributions are summarized as follows. (*i*) We present Adversarial Poincaré Variational Autoencoder (APo-VAE), a novel approach to text embedding and generation based on hyperbolic latent representations. (*ii*) In addition to the use of a wrapped normal distribution, an adversarial learning procedure and a VampPrior design are incorporated for robust model training. (*iii*) Experiments on language modeling, unaligned style transfer, and dialog-response generation benchmarks demonstrate the superiority of the proposed approach compared to Euclidean VAEs, as it benefits from capturing informative latent hierarchies in natural language.

## 3.2 Preliminaries

### 3.2.1 Variational Autoencoder

Let $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ be a dataset of sentences, where each $\boldsymbol{x}_i = [x_{i,1}, ..., x_{i,T_i}]$ is a sequence of tokens of length $T_i$. Our goal is to learn $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ that best models the observed sentences so that the expected log-likelihood is maximized, *i.e.*, $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N}\sum_i \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$.

The variational autoencoder (VAE) [KW13, CTZ+18] considers a latent-variable model $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$ to represent sentences, with an auxilary encoder that draws samples of latent code $\boldsymbol{z}$ from the conditional density $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, known as the approximate posterior. Given a latent code $\boldsymbol{z}$, the decoder samples a sentence from the conditional density $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) = \prod_t p(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{z})$, where the "decoding" pass takes an auto-regressive form. Together with prior $p(\boldsymbol{z})$, the model is given by the joint $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$. The VAE leverages the approximate posterior to derive an *evidence lower bound* (ELBO) to the (intractable) marginal log-likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) =$

$\log \int p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})\, \mathrm{d}\boldsymbol{z}$:

$$\mathcal{L}(\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[ \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \right] , \tag{3.1}$$

where $(\boldsymbol{\theta}, \boldsymbol{\phi})$ are jointly optimized during training, and the gap is given by the decomposition

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathcal{L}(\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\phi}) + \mathbb{D}_{\mathrm{KL}}(p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) \parallel q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})) , \tag{3.2}$$

where $\mathbb{D}_{\mathrm{KL}}$ denotes Kullback-Leibler divergence. Alternatively, the ELBO can be written as:

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} [\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] \\ - \mathbb{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z})) , \end{aligned} \tag{3.3}$$

where the first conditional likelihood and second KL terms respectively characterize reconstruction and generalization capabilities. Intuitively, a good model is expected to strike a balance between good reconstruction and generalization. In most cases, both the prior and variational posterior are assumed to be Gaussian for computational convenience. However, such over-simplified assumptions may not be ideal for capturing the intrinsic characteristics of data that have unique geometrical structure, such as natural language.

### 3.2.2 Hyperbolic Space

Riemannian manifolds can provide a more powerful and meaningful embedding space for complex data with highly non-Euclidean structure, that cannot be effectively captured in a vectorial form (*e.g.*, social networks, biology and computer graphics). Of particular interest is the hyperbolic space [GBH18], where $(i)$ the relatively simple geometry allows tractable computations, and $(ii)$ the exponential growth of distance in finite dimensions naturally embeds rich hierarchical structure in a compact form.

**Riemannian Geometry.** An $n$-dimensional Riemannian manifold $\mathcal{M}^n$ is a set of points locally similar to a linear space $\mathbb{R}^n$. At each point $\boldsymbol{x}$ of the manifold $\mathcal{M}^n$, we can define a real vector space $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n$ that is tangent to $\boldsymbol{x}$, along with an associated *metric tensor* $g_{\boldsymbol{x}}(\cdot,\cdot) : \mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n \times \mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n \to \mathbb{R}$ which is an inner product on $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n$. Intuitively, a Riemannian manifold behaves like a vector space only in its infinitesimal neighborhood, allowing the generalization of common notation like angle, straight line and distance to a smooth manifold. For each tangent space $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n$, there exists a specific one-to-one map $\exp_{\boldsymbol{x}}(\boldsymbol{v}) : \mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n \to \mathcal{M}^n$ from an $\epsilon$-ball at the origin of $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n$ to a neighborhood of $\boldsymbol{x}$ on $\mathcal{M}^n$, called the *exponential map*. We refer to the inverse of an exponential map as the *logarithm map*, denoted $\log_{\boldsymbol{x}}(\boldsymbol{y}) : \mathcal{M}^n \to \mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n$. In addition, a parallel transport $P_{\boldsymbol{x}\to\boldsymbol{x}'} : \mathcal{T}_{\boldsymbol{x}}\mathcal{M}^n \to \mathcal{T}_{\boldsymbol{x}'}\mathcal{M}^n$ intuitively transports tangent vectors along a "straight" line between $\boldsymbol{x}$ and $\boldsymbol{x}'$, so that they remain "parallel." This is the basic machinery that allows us to generalize distributions and computations in the hyperbolic space, as detailed in later sections.

**Poincaré Ball Model.** Hyperbolic geometry is one type of non-Euclidean geometry with a constant negative curvature. As a classical example of hyperbolic space, an $n$-dimensional Poincaré ball, with curvature parameter $c \geq 0$ (*i.e.*, radius $\frac{1}{\sqrt{c}}$), can be denoted as $\mathbb{B}_c^n := \{\boldsymbol{z} \in \mathbb{R}^n \,|\, c\|\boldsymbol{z}\|^2 < 1\}$ with its metric tensor given by $g_{\boldsymbol{z}}^c = \lambda_{\boldsymbol{z}}^2 g^E$, where $\lambda_{\boldsymbol{z}} = \frac{2}{1-c\|\boldsymbol{z}\|^2}$ and $g^E$ denotes the regular Euclidean metric tensor. Intuitively, as $\boldsymbol{z}$ moves closer to the boundary $\frac{1}{\sqrt{c}}$, the hyperbolic distance between $\boldsymbol{z}$ and a nearby $\boldsymbol{z}'$ diverges at a rate of $\frac{1}{1-c\|\boldsymbol{z}\|^2} \to \infty$. This implies significant representation capacity, as very dissimilar objects can be encoded on a compact domain. Note that as $c \to 0$, the model recovers the Euclidean space $\mathbb{R}^n$, *i.e.*, the lack of hierarchy. In comparison, a larger $c$ implies a stronger hierarchical organization.[2]

[2]The fact that APo-VAE outperforms standard VAE evidences the existence of the hierarchical organization in NLP data.

**Mathematical Operations.** We review the closed-form mathematical operations that enable differentiable training for hyperbolic space models, namely the hyperbolic algebra (vector addition) and tangent space computations (exponential/logarithm map and parallel transport). The hyperbolic algebra is formulated under the framework of *gyrovector spaces* [Ung08], with the addition of two points $\boldsymbol{z}, \boldsymbol{z}' \in \mathbb{B}_c^n$ given by the *Möbius addition*:

$$\boldsymbol{z} \oplus_c \boldsymbol{z}' := \tag{3.4}$$
$$\frac{(1 + 2c\langle \boldsymbol{z}, \boldsymbol{z}' \rangle + c\|\boldsymbol{z}'\|^2)\boldsymbol{z} + (1 - c\|\boldsymbol{z}\|^2)\boldsymbol{z}'}{1 + 2c\langle \boldsymbol{z}, \boldsymbol{z}' \rangle + c^2\|\boldsymbol{z}\|^2\|\boldsymbol{z}'\|^2}.$$

For any point $\boldsymbol{\mu} \in \mathbb{B}_c^n$, the exponential map and the logarithmic map are given for $\boldsymbol{u} \neq \boldsymbol{0}$ and $\boldsymbol{y} \neq \boldsymbol{\mu}$ by

$$\exp_{\boldsymbol{\mu}}^c(\boldsymbol{u}) := \boldsymbol{\mu} \oplus_c (\tanh(\sqrt{c}\frac{\lambda_{\boldsymbol{\mu}}^c \|\boldsymbol{u}\|}{2})\frac{\boldsymbol{u}}{\sqrt{c}\|\boldsymbol{u}\|}),$$
$$\log_{\boldsymbol{\mu}}^c(\boldsymbol{y}) := \frac{2}{\sqrt{c}\lambda_{\boldsymbol{\mu}}^c} \tanh^{-1}(\sqrt{c}\|\boldsymbol{\kappa}_{\boldsymbol{\mu},\boldsymbol{y}}\|)\frac{\boldsymbol{\kappa}_{\boldsymbol{\mu},\boldsymbol{y}}}{\|\boldsymbol{\kappa}_{\boldsymbol{\mu},\boldsymbol{y}}\|}, \tag{3.5}$$

where $\boldsymbol{\kappa}_{\boldsymbol{\mu},\boldsymbol{y}} := (-\boldsymbol{\mu}) \oplus_c \boldsymbol{y}$. Note that the Poincaré ball model is *geodesically complete* in the sense that $\exp_{\boldsymbol{\mu}}^c$ is well-defined on the full tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathbb{B}_c^n$. The parallel transport map from a vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{0}}\mathbb{B}_c^n$ to another tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathbb{B}_c^n$ is given by

$$P_{\boldsymbol{0}\to\boldsymbol{\mu}}^c(\boldsymbol{v}) = \log_{\boldsymbol{\mu}}^c(\boldsymbol{\mu} \oplus_c \exp_{\boldsymbol{0}}^c(\boldsymbol{v})) = \frac{\lambda_{\boldsymbol{0}}^c}{\lambda_{\boldsymbol{\mu}}^c}\boldsymbol{v}. \tag{3.6}$$

## 3.3 Adversarial Poincaré VAE

We first introduce our hyperbolic encoder and decoder, and how to apply reparametrization. We then provide detailed descriptions on model implementation, explaining how the primal-dual form of KL divergence can help stabilize training. Finally, we describe how to adopt VampPrior [TW18] to enhance performance. A summary of our model scheme is provided in Figure 3.2.

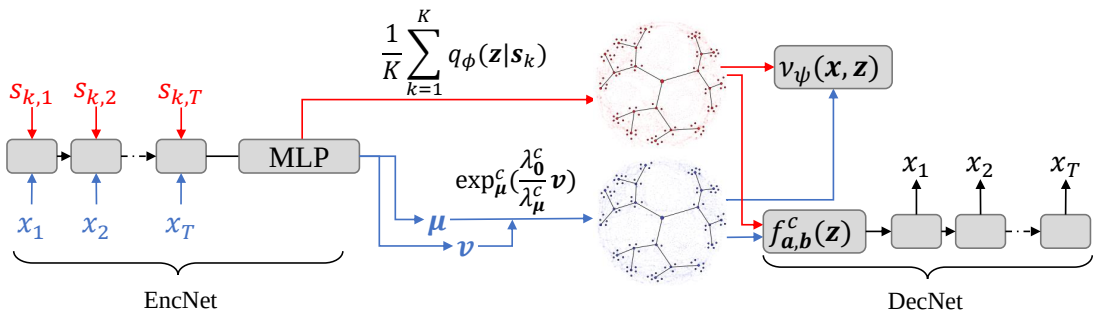



**Figure 3.2**: Model framework of the proposed APo-VAE (red is the prior and blue is the posterior). $\boldsymbol{x} = [x_1, ..., x_T]$ is text sequential data, and $\boldsymbol{s}_k = [s_{k,1}, ..., s_{k,T}]$ is the pseudo-input. The posterior (blue) is obtained by (3.7), and VampPrior (red) is achieved by (3.12). $\nu_\psi(\boldsymbol{x}, \boldsymbol{z})$ is the dual function.

### 3.3.1 Flexible Wrapped Distribution Encoder

We begin by generalizing the standard normal distribution to a Poincaré ball [GBH18]. While there are a few competing definitions of the hyperbolic normal, we choose the wrapped normal as our prior and variational posterior, largely due to its flexibility for more expressive generalization. A wrapped normal distribution $\mathcal{N}_{\mathbb{B}_c^n}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is defined as follows: $(i)$ sample vector $\boldsymbol{v}$ from $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, $(ii)$ parallel transport $\boldsymbol{v}$ to $\boldsymbol{u} := P_{\mathbf{0}\to\boldsymbol{\mu}}^c(\boldsymbol{v})$, and $(iii)$ using exponential map to project $\boldsymbol{u}$ back to $\boldsymbol{z} := \exp_{\boldsymbol{\mu}}^c(\boldsymbol{u})$. Putting these together, a latent sample has the following reparametrizable form:

$$\boldsymbol{z} = \exp_{\boldsymbol{\mu}}^c\left(\frac{\lambda_{\mathbf{0}}^c}{\lambda_{\boldsymbol{\mu}}^c}\boldsymbol{v}\right), \boldsymbol{v} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}). \tag{3.7}$$

For approximate posteriors, $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ depends on $\boldsymbol{x}$. We further generalize the (restrictive) hyperbolic wrapped normal by acknowledging that under the implicit VAE [FLG+19] framework, one does not need the approximate posterior $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to be analytically tractable. This allows us to replace the tangent space sampling step $\boldsymbol{v} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ in (3.7) with a more flexible implicit distribution from which we draw samples as $\boldsymbol{v} := G(\boldsymbol{x}, \boldsymbol{\xi}; \boldsymbol{\phi}_1)$ for $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Note that now $\boldsymbol{\mu} := F(\boldsymbol{x}; \boldsymbol{\phi}_2)$ can be regarded as a deterministic displacement vector that anchors embeddings to the correct se-

mantic neighborhood, allowing the stochastic $\boldsymbol{v}$ to only focus on modeling the local uncertainty of the semantic embedding. The synergy between the deterministic and stochastic parts enables efficient representation learning relative to existing alternatives. For simplicity, we denote the encoder neural network as $\text{EncNet}_{\boldsymbol{\phi}}$, which contains $G$ and $F$, with parameters $\boldsymbol{\phi} = \{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2\}$.

### 3.3.2 Poincaré Decoder

To build a geometry-aware decoder for a hyperbolic latent code, we follow [GBH18], and use a generalized linear function analogously defined in the hyperbolic space. A Euclidean linear function takes the form $f(\boldsymbol{z}) = \langle \boldsymbol{a}, \boldsymbol{z} - \boldsymbol{b} \rangle = \text{sign}(\langle \boldsymbol{a}, \boldsymbol{z} - \boldsymbol{b} \rangle) \|\boldsymbol{a}\| d^E(\boldsymbol{z}, H_{\boldsymbol{a},\boldsymbol{b}})$, where $\boldsymbol{a}$ is the coefficient, $\boldsymbol{b}$ is the intercept, $H_{\boldsymbol{a},\boldsymbol{b}}$ is a hyperplane passing through $\boldsymbol{b}$ with $\boldsymbol{a}$ as the normal direction, and $d^E(\boldsymbol{z}, H)$ is the distance between $\boldsymbol{z}$ and hyperplane $H$. The counterpart in Poincaré ball analogously writes

$$f^c_{\boldsymbol{a},\boldsymbol{b}}(\boldsymbol{z}) = \text{sign}(\langle \boldsymbol{a}, \log^c_{\boldsymbol{b}}(\boldsymbol{z}) \rangle_{\boldsymbol{b}}) \|\boldsymbol{a}\|_{\boldsymbol{b}} \, d^{\mathbb{B}}_c(\boldsymbol{z}, H^c_{\boldsymbol{a},\boldsymbol{b}}), \tag{3.8}$$

where $H^c_{\boldsymbol{a},\boldsymbol{b}} = \{\boldsymbol{z} \in \mathbb{B}^n_c | \langle \boldsymbol{a}, \log^c_{\boldsymbol{b}}(\boldsymbol{z}) \rangle_{\boldsymbol{b}} = 0\}$, and $d^{\mathbb{B}}_c(\boldsymbol{z}, H^c_{\boldsymbol{a},\boldsymbol{b}}) = \frac{1}{\sqrt{c}} \sinh^{-1}\left(\frac{2\sqrt{c}|\langle \boldsymbol{\kappa}_{\boldsymbol{b},\boldsymbol{z}}, \boldsymbol{a} \rangle|}{(1 - c\|\boldsymbol{\kappa}_{\boldsymbol{b},\boldsymbol{z}}\|^2)\|\boldsymbol{a}\|}\right)$ are the the gyroplane and the distance between $\boldsymbol{z}$ and the gyroplane, respectively. Specifically, we use the hyperbolic linear function in (3.8) to extract features from the Poincaré embedding $\boldsymbol{z}$. The feature $f^c_{\boldsymbol{a},\boldsymbol{b}}(\boldsymbol{z})$ will be the input to the RNN decoder. We denote the combined network of $f^c_{\boldsymbol{a},\boldsymbol{b}}$ and the RNN decoder as $\text{DecNet}_{\boldsymbol{\theta}}$, where parameters $\boldsymbol{\theta}$ contain $\boldsymbol{a}$ and $\boldsymbol{b}$.

### 3.3.3 Implementing APo-VAE

While it is straightforward to compute the ELBO (3.3) via Monte Carlo estimates using the explicit wrapped normal density [MLM+19], we empirically observe that: $(i)$ the normal assumption restricts the expressiveness of the model, and $(ii)$ the

wrapped normal likelihood makes the training unstable. Therefore, we appeal to a primal-dual view of VAE training to overcome such difficulties [R+66, DDH+18, TCD+19, FLG+19]. Specifically, the KL term in (3.3) can be reformulated as:

$$\mathbb{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z})) = \max_{\boldsymbol{\psi}} \tag{3.9}$$

$$\left\{\mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\exp\nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})\right\},$$

where $\nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})$ is the (auxiliary) dual function (*i.e.*, a neural network) with parameters $\boldsymbol{\psi}$. The primal-dual view of the KL term enhances the approximation ability, while also being tractable computationally. Meanwhile, since the density function in the original KL term in (3.3) is replaced by the dual function $\nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})$, we can avoid direct computation with respect to the probability density function of the wrapped normal distribution.

To train our proposed APo-VAE with the primal-dual form of the VAE objective, we follow the training schemes of coupled variational Bayes (CVB) [DDH+18] and implicit VAE [FLG+19], which optimize the objective adversarially. Specifically, we update $\boldsymbol{\psi}$ in the dual function $\nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})$ to maximize:

$$\begin{aligned}\mathcal{L}_1 = \mathbb{E}_{\boldsymbol{x}\sim\boldsymbol{X}}[\,&\mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z}) \\ &- \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\exp\nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})\,],\end{aligned} \tag{3.10}$$

where $\mathbb{E}_{\boldsymbol{x}\sim\boldsymbol{X}}[\cdot]$ denotes the expectation over empirical distribution on observations. Accordingly, parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are updated to maximize:

$$\begin{aligned}\mathcal{L}_2 = \mathbb{E}_{\boldsymbol{x}\sim\boldsymbol{X}}\mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\,&\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) \\ &- \nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})\,].\end{aligned} \tag{3.11}$$

Note that the term $\mathbb{E}_{\boldsymbol{x}\sim\boldsymbol{X}}\mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\nu_{\boldsymbol{\psi}}(\boldsymbol{x},\boldsymbol{z})$ is maximized in (3.10) while it is minimized in (3.11), *i.e.*, adversarial learning. In other words, one can consider the dual function as a discriminative network that distinguishes between the prior $\boldsymbol{z}\sim p(\boldsymbol{z})$

**Algorithm 1** Training procedure of APo-VAE.

**Input**: Data samples $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^N$, Poincaré curvature $c$, and number of pseudo-input $K$.
Initialize $\boldsymbol{\theta}$, $\boldsymbol{\phi}$, $\boldsymbol{\psi}$, and $\boldsymbol{\delta}$.
**for** *iter* from 1 to *max_iter* **do**
  Sample a mini-batch $\{\boldsymbol{x}_m\}_{m=1}^M$ from $\boldsymbol{X}$ of size $M$.
  ***# Sampling in the Hyperbolic Space.***
  Obtain $\boldsymbol{\mu}_m$ and $\boldsymbol{v}_m$ from $\text{EncNet}_{\boldsymbol{\phi}}(\boldsymbol{x}_m)$.
  Move $\boldsymbol{v}_m$ to $\boldsymbol{u}_m = P^c_{\boldsymbol{0}\to\boldsymbol{\mu}_m}(\boldsymbol{v}_m)$ by (3.6).
  Map $\boldsymbol{u}_m$ to $\boldsymbol{z}_m = \exp^c_{\boldsymbol{\mu}_m}(\boldsymbol{u}_m)$ by (3.5).
  ***# Update the dual function and the pseudo-input.***
  Sample $\tilde{\boldsymbol{z}}_m$ by (3.12).
  Update $\boldsymbol{\psi}$ and $\boldsymbol{\delta}$ by gradient ascent on (3.10)
  ***# Update the encoder and decoder networks.***
  Update $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ by gradient ascent on (3.11).
**end for**

and the variational posterior $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, both of which are paired with the input data $\boldsymbol{x} \sim \boldsymbol{X}$.

### 3.3.4 Data-driven Prior

While the use of a standard normal prior is a simple choice in Euclidean space, we argue that it induces bias in the hyperbolic setup. This is because natural sentences have specific meaning, and it is unrealistic to have the bulk of mass concentrated in the center (this is for low dimension; for high dimensions, it will concentrate near the surface of a sphere, which may partly explain why cosine similarity works favorably compared with Euclidean distance for NLP applications).

To reduce the induced bias from a pre-fixed prior, we adopt the VampPrior framework [TW18], which is a mixture of variational posteriors conditioned on learnable pseudo-data points. Specifically, we consider the prior as a learnable distribution given by

$$p_{\boldsymbol{\delta}}(\boldsymbol{z}) = \frac{1}{K}\textstyle\sum_{k=1}^K q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{s}_k), \tag{3.12}$$

where $q_{\boldsymbol{\phi}}$ is the learned approximate posterior, and we call the parameter $\boldsymbol{\delta} := \{\boldsymbol{s}_k\}_{k=1}^K$ pseudo inputs. Intuitively, $p_{\boldsymbol{\delta}}(\boldsymbol{z})$ seeks to match the aggregated posterior [MSJ+15]: $q(\boldsymbol{z}) = \frac{1}{N}\sum_{i=1}^N q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}_i)$ in a cost-efficient manner via parameterizing the pseudo inputs. By replacing the prior distribution $p(\boldsymbol{z})$ in (3.10) with $p_{\boldsymbol{\delta}}(\boldsymbol{z})$, we complete the final objective of the proposed APo-VAE. The detailed training procedure is summarized in Algorithm 1.

## 3.4 Related Work

**VAE for Text Generation.** Many VAE models have been proposed for text generation, most of which focus on solving the issue of posterior collapse. The most popular strategy is to alter the training dynamics, keeping the encoder away from bad local optima. For example, variants of KL annealing [BVV+16, ZSE18, FLL+19] dynamically adjust the weight on the KL penalty term as training progresses. Lagging VAE [HSNBK19] aggressively optimizes the encoder before each decoder update, to overcome the imbalanced training issue between the encoder and decoder. Alternative strategies have also been proposed based on competing theories or heuristics. $\delta$-VAE [ROPV19] tackles this issue by enforcing a minimum KL divergence between the posterior and the prior. [YHSBK17] blames mode-collapse on the auto-regressive design of the decoder and advocates alternative architectures. A semi-amortized inference network is considered by [KWM+18] to bridge the amortization gap between log-likelihood and the ELBO.

Recent work has also shown that posterior collapse can be ameliorated by using more expressive priors and variational posteriors other than Gaussian. Flow-based VAE is considered in [ZR19] to enhance the flexibility of prior distributions. A topic-guided prior is proposed in [WGX+19] to achieve more controllable text generation. [FLG+19] explores implicit sample-based representations, without requiring

an explicit density form for the approximate posterior. [XD18] considers replacing the Gaussian with the spherical von Mises-Fisher (vMF) distribution. Compared to these prior arts, our model features structured representation in hyperbolic space, which not only captures latent hierarchies but also combats posterior collapse.

**Hyperbolic Space Representation Learning.** There has been a recent surge of interest in representation learning in hyperbolic space, largely due to its exceptional effectiveness modeling data with underlying graphical structure [CCD17], such as relation nets [NK17]. In the context of NLP, hyperbolic geometry has been considered for word embeddings [TBG18]. A popular vehicle for hyperbolic representation learning is the auto-encoder (AE) framework [GLA19, Ovi19], where the decoders are built to efficiently exploit the hyperbolic geometry [GBH18]. Closest to our APo-VAE are the works of hyperbolic VAEs [MLM+19, NYFK19b], where wrapped normal distributions have been used. Drawing power from the dual form of the KL, the proposed APo-VAE highlights an implicit posterior and data-driven prior, showing improved training stability.

## 3.5 Experiments

We evaluate the proposed model on three tasks: (*i*) language modeling, (*ii*) unaligned style transfer, and (*iii*) dialog-response generation, with quantitative results, human evaluation and qualitative analysis.

### 3.5.1 Experimental Setup

**Datasets.** We use three datasets for language modeling: *Penn Treebank* (PTB) [MSM93], *Yahoo* and *Yelp* corpora [YHSBK17]. PTB contains one million words of 1989 Wall Street Journal material annotated in Treebank II style, with 42k sentences of varying

lengths. Yahoo and Yelp are much larger datasets, each containing 100k sentences with greater average length.

For unaligned style transfer, we use the Yelp restaurant reviews dataset [SLBJ17], which is obtained by pre-processing the Yelp dataset, *i.e.*, sentences are shortened for more feasible sentence level sentiment analysis. Overall, the dataset includes 350k positive and 250k negative reviews (based on user rating).

Following [GCHK19], we use the Switchboard [GH97b] dataset for dialogue-response generation. The former contains 2.4k two-sided telephone conversations, manually transcribed and aligned. We split the data into training, validation and test sets following the protocol described in [ZZE17].

**Evaluation Metrics.** To benchmark language modeling performance, we report the ELBO and *Perplexity* (PPL) of APo-VAE and baselines. In order to verify our proposed Apo-VAE is more resistant to posterior collapse, we also report the KL-divergence $\mathbb{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\|p(\boldsymbol{z}))$ and *mutual information* (MI) between $\boldsymbol{z}$ and $\boldsymbol{x}$ [HSNBK19]. The number of active units (AU) of the latent code is also reported, where the activity of a latent dimension $z$ is measured as $A_{\boldsymbol{z}} = \mathrm{Cov}_{\boldsymbol{x}}\mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[z]$, and defined as active if $A_{\boldsymbol{z}} > 0.01$.

To evaluate our model on unaligned style transfer, we consider the transfer accuracy from one sentiment to another, the BLEU scores between original and transferred sentences, the reconstruction perplexity of original sentences, and the reverse perplexity (RPPL) based on a language model from the transferred sentences.

For dialogue-response generation, we adopt the evaluation metrics used in previous studies [ZZE17, GCHK19], including BLEU [PRWZ02], BOW [LLS$^+$16], and *intra/inter-dist* values [GCHK19]. The first two metrics are used to assess the relevance of the generated response, and the third is for diversity evaluation.

**Model Implementation.** For language modeling, we adopt the LSTM [HS97] for both the encoder and decoder, with dimension of the latent code set to 32. Following [MLM+19], the hyper-parameter $c$ is set to 0.7. For unaligned style transfer, we extend our model in the same fashion as [FLG+19]. For dialogue-response generation, we modify APo-VAE following the conditional VAE framework [ZZE17]. Specifically, an extra input of context embedding $\boldsymbol{s}$ is supplied to the model (*i.e.*, $p_\theta(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{s}), q_\phi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{s})$). The prior $p(\boldsymbol{z}|\boldsymbol{s})$ is a wrapped normal conditioned on context embedding, learned together with the posterior.

### 3.5.2 Experimental Results

**Language Modeling.** Table 3.1 shows results on language modeling. We compare APo-VAE with other VAE-based solutions, including $\beta$-VAE [HMP+17], SA-VAE [KWM+18], lagging VAE (LAG-VAE) [HSNBK19], vMF-VAE [XD18], Poincaré VAE ($\mathcal{P}$-VAE) [MLM+19] and iVAE[3] [FLG+19]. On all three datasets, the proposed model achieves lower negative ELBO and PPL than other models, demonstrating its strong ability to better model sequential text data. Meanwhile, the larger KL term and higher mutual information (between $\boldsymbol{z}$ and $\boldsymbol{x}$) of APo-VAE model indicate its robustness in handling posterior collapse. In addition, the introduction of a data-driven prior (denoted as APo-VAE+VP) further boosts the performance, especially on negative ELBO and PPL.

**Visualization.** To verify our hypothesis that the proposed model is capable of learning latent tree structure in text data, we visualize the two-dimensional projection of the learned latent code in Figure 3.3. For visualization, we randomly draw 5k samples from PTB-test, and encode them to the latent space using the APo-VAE encoder. We color-code each sentence based on its length (*i.e.*, blue for long sentences

[3] We report $\text{iVAE}_{\text{MI}}$ results in all our experiments.

**Table 3.1**: Results on PTB, Yahoo, and Yelp datasets. A better language model achieves lower negative ELBO and PPL. Higher KL and MI indicate a better utilization of the latent space.

| Model | -ELBO | PPL | KL | MI | AU |
|---|---|---|---|---|---|
| | PTB | | | | |
| VAE | 102.6 | 108.26 | 1.1 | 0.8 | 2 |
| $\beta$-VAE | 104.5 | 117.92 | 7.5 | 3.1 | 5 |
| SA-VAE | 102.6 | 107.71 | 1.2 | 0.7 | 2 |
| vMF-VAE | 95.8 | 93.70 | 2.9 | 3.2 | 21 |
| $\mathcal{P}$-VAE | 91.4 | 76.13 | 4.5 | 2.9 | 23 |
| iVAE | 87.2 | 53.44 | **12.5** | **12.2** | **32** |
| APo-VAE | 87.2 | 53.32 | 8.4 | 4.8 | **32** |
| APo-VAE+VP | **87.0** | **53.02** | 8.9 | 4.5 | **32** |
| | Yahoo | | | | |
| VAE | 328.6 | 61.21 | 0.0 | 0.0 | 0 |
| $\beta$-VAE | 328.7 | 61.29 | 6.3 | 2.8 | 8 |
| SA-VAE | 327.2 | 60.15 | 5.2 | 2.9 | 10 |
| LAG-VAE | 326.7 | 59.77 | 5.7 | 2.9 | 15 |
| vMF-VAE | 318.5 | 53.92 | 6.3 | 3.7 | 23 |
| $\mathcal{P}$-VAE | 313.4 | 50.57 | 7.2 | 3.3 | 27 |
| iVAE | 309.1 | 47.93 | **11.4** | **10.7** | **32** |
| APo-VAE | 286.2 | 47.00 | 6.9 | 4.1 | **32** |
| APo-VAE+VP | **285.6** | **46.61** | 8.1 | 4.9 | **32** |
| | Yelp | | | | |
| VAE | 357.9 | 40.56 | 0.0 | 0.0 | 0 |
| $\beta$-VAE | 358.2 | 40.69 | 4.2 | 2.0 | 4 |
| SA-VAE | 357.8 | 40.51 | 2.8 | 1.7 | 8 |
| LAG-VAE | 355.9 | 39.73 | 3.8 | 2.4 | 11 |
| vMF-VAE | 356.2 | 51.03 | 4.1 | 3.9 | 13 |
| $\mathcal{P}$-VAE | 355.4 | 50.64 | 4.3 | 4.8 | 19 |
| iVAE | 348.7 | 36.88 | 11.6 | **11.0** | **32** |
| APo-VAE | 319.7 | 34.10 | 12.1 | 7.5 | **32** |
| APo-VAE+VP | **316.4** | **32.91** | **12.7** | 6.2 | **32** |

and red for short sentences). Note that only a small portion of data have a length longer than 32 ($< 10\%$), and human inspection verified that most of them contain multiple sub-sentences. We exclude these samples from our analysis.

As shown in Figure 3.3, longer sentences (dark blue) tend to occupy the outer rim of the Poincaré ball, while the shorter ones (dark red) are concentrated in the

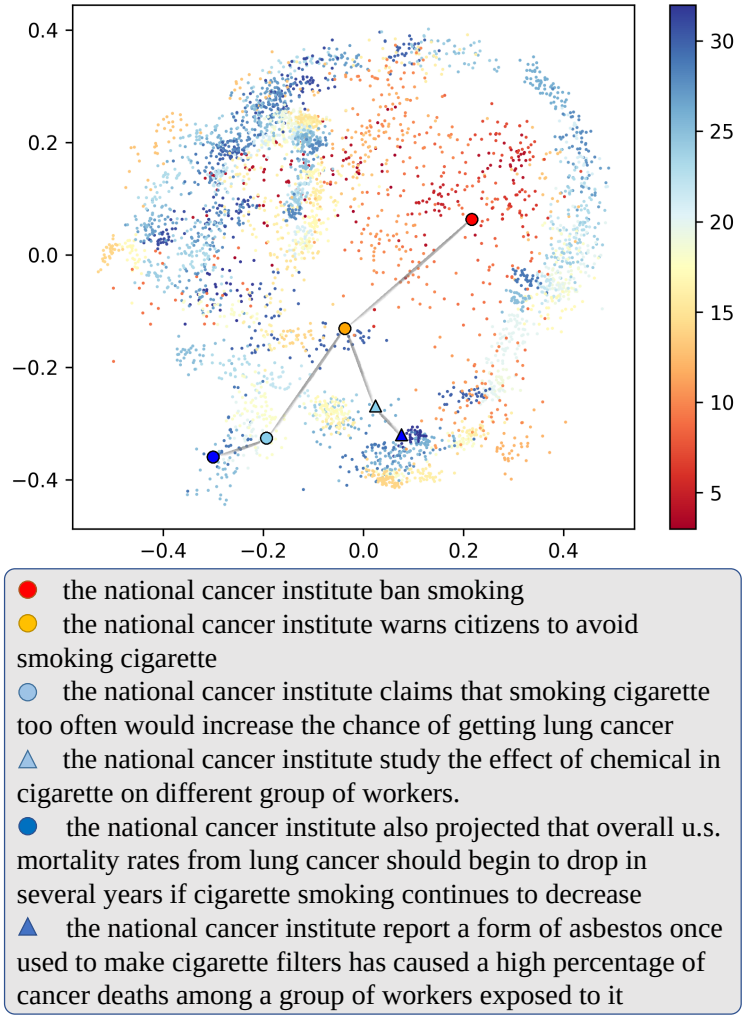



**Figure 3.3**: Visualization of the hyperbolic latent space of 5,000 randomly sampled sentences from the test set of PTB. The lengths of the samples are color-coded (red for short ones and blue for longer ones). The six listed sentences are created by modifying data samples.

inner area. We also select some long sample sentences (dark blue), and manually shorten them to create several variants of different lengths (ranging from 6 to 27), which are related in a hierarchical manner based on human judgement. We visualize their latent codes projected by the trained APo-VAE. The resulting plot is consistent with a hierarchical structure for APo-VAE: as the sentence becomes more specific, the embedding moves outward. We also decode from the neighbours of these latent

codes, the outputs (see the Appendix) of which demonstrate a similar hierarchical structure.

**Unaligned Style Transfer.** Table 3.2 shows the results on the Yelp restaurant reviews dataset. APo-VAE achieves over 1 point increased BLEU scores than iVAE, capturing a more informative and structured feature space. Comparable performance is achieved for the other evaluation metrics.

**Table 3.2**: Unaligned style transfer on the Yelp restaurant reviews dataset.

| Model | ACC | BLEU | PPL | RPPL |
|---|---|---|---|---|
| ARAE | **95** | 32.5 | 6.8 | 395 |
| iVAE | 92 | 36.7 | 6.2 | 285 |
| APo-VAE | **95** | **37.8** | **6.1** | **273** |

**Dialogue Response Generation.** Results on SwitchBoard are summarized in Table 3.3. Our proposed model generates comparable or better responses than the baseline models in terms of both relevance (BLEU and BOW) and diversity (intra/inter-dist). APo-VAE improves the average recall from 0.427 (by iVAE) to 0.438, while significantly enhancing generation diversity (*e.g.*, from 0.692 to 0.792 for intra-dist-2).

**Table 3.3**: Results on SwitchBoard (P: precision, R: recall, A: average, E: extreme, G: greedy). Higher BLEU and BOW Embedding indicate better quality of generated responses. Higher intra/inter-dist means better generation diversity.

| Model | BLEU | | | BOW | | | Intra-dist | | Inter-dist | |
|---|---|---|---|---|---|---|---|---|---|---|
| | R | P | F1 | A | E | G | dist-1 | dist-2 | dist-1 | dist-2 |
| CVAE | 0.295 | 0.258 | 0.275 | 0.836 | 0.572 | 0.846 | 0.803 | 0.415 | 0.112 | 0.102 |
| CVAE-BOW | 0.298 | **0.272** | 0.284 | 0.828 | 0.555 | 0.840 | 0.819 | 0.493 | 0.107 | 0.099 |
| CVAE-CO | 0.299 | 0.269 | 0.283 | 0.839 | 0.557 | 0.855 | 0.863 | 0.581 | 0.111 | 0.110 |
| DialogWAE | 0.394 | 0.254 | 0.309 | 0.897 | 0.627 | 0.887 | 0.713 | 0.651 | 0.245 | 0.413 |
| iVAE | 0.427 | 0.254 | 0.319 | 0.930 | 0.670 | 0.900 | 0.828 | 0.692 | 0.391 | 0.668 |
| APo-VAE | **0.438** | 0.261 | **0.328** | **0.937** | **0.683** | **0.904** | **0.861** | **0.792** | **0.445** | **0.717** |

**Human Evaluation.** We further perform human evaluation via Amazon Mechanical Turk. We asked the turkers to compare generated responses from two models, and assess each model's informativeness, relevance to the dialog context (coherence), and diversity. We use 500 randomly sampled contexts from the test set, each assessed by three judges. In order to evaluate diversity, 5 responses are generated for each dialog context. For quality control, only workers with a lifetime task approval rating greater than 98% were allowed to participate in our study. Table 3.4 summarizes the human evaluation results. The responses generated by our model are clearly preferred by the judges compared with other competing methods.

**Table 3.4**: Human evaluation results. Win/loss/tie indicates the percentage of responses generated by APo-VAE being better/worse/equal to the compared model.

| | vs iVAE | | | vs DialogWAE | | |
|---|---|---|---|---|---|---|
| | win | loss | tie | win | loss | tie |
| Informativeness | 52.8 | 27.9 | 19.3 | 63.7 | 27.1 | 19.2 |
| Coherence | 41.7 | 35.5 | 22.8 | 41.2 | 34.4 | 24.4 |
| Diversity | 51.2 | 26.4 | 22.4 | 62.1 | 25.1 | 12.8 |

## 3.6 Conclusions

We present APo-VAE, a novel model for text generation in hyperbolic space. Our model can learn latent hierarchies in natural language via the use of wrapped normals for the prior. A primal-dual view of KL divergence is adopted for robust model training. Extensive experiments on language modeling, text style transfer, and dialog response generation demonstrate the superiority of the model. For future work, we plan to combine APo-VAE with the currently prevailing large-scale pre-trained language models.

## 3.7 Supplementary Material

### 3.7.1 Basics of Riemannian Geometry (Extended)

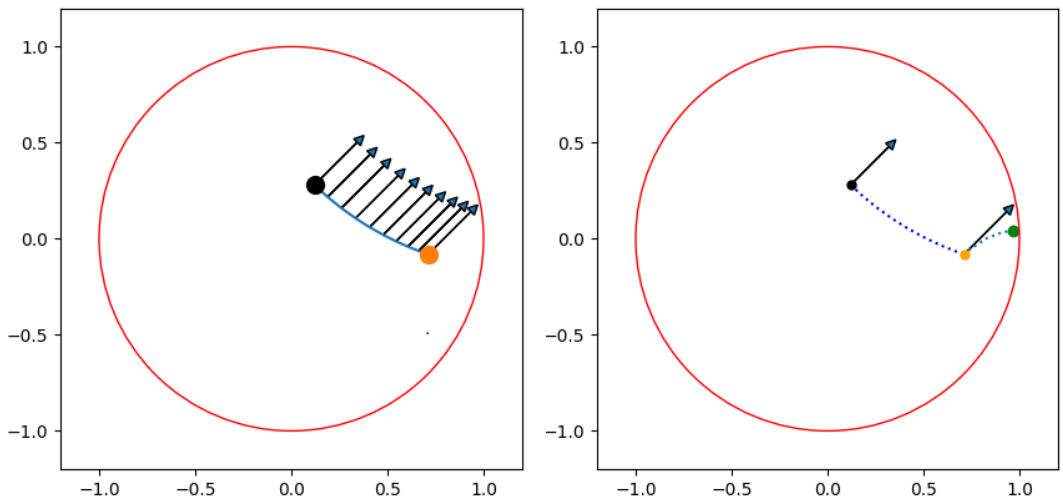
This section provides additional coverage of the basic mathematical concepts used in APo-VAE. For more detailed mathematics expositions, readers are referred to [GBH18].

A real, smooth $n$-dimensional manifold $\mathcal{M}$ is a set of points locally similar to a linear space $\mathbb{R}^n$. At each point $\boldsymbol{x}$ of the manifold $\mathcal{M}$ is defined a real vector space of the space of the same dimensionality as $\mathcal{M}$, called the tangent space in $\boldsymbol{x}$: $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}$. Intuitively it contains all the possible directions in which one can tangentially pass through $\boldsymbol{x}$. For each point $\boldsymbol{x}$ there also defines a *metric tensor* $g_{\boldsymbol{x}}(\cdot,\cdot) : \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \times \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \to \mathbb{R}$, which defined an inner product on the associated tangent space $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}$. More specifically, given a coordinate system, the inner product is given in the quadratic form $g_{\boldsymbol{x}}(\boldsymbol{u},\boldsymbol{v}) = \langle \boldsymbol{u},\boldsymbol{v}\rangle_{g_{\boldsymbol{x}}} = \boldsymbol{u}^T\boldsymbol{G}_{\boldsymbol{x}}\boldsymbol{v}$, where by slight abuse of notation $\boldsymbol{u},\boldsymbol{v} \in \mathbb{R}^n$ are vector representations of the tangent vectors wrt the coordinate system and $\boldsymbol{G}_{\boldsymbol{x}} \in \mathbb{R}^{n\times n}$ is a positive-definite matrix. Collectively, $(\mathcal{M}, g)$ defines a *Riemannian manifold*.

The metric tensor is used to generalize the notations such as distance and volume in Euclidean space to the Riemannian manifold. Given a curve $\gamma(t) : [0,1] \to \mathcal{M}$, its length is given by

$$L(\gamma) = \int_0^1 \|\gamma'(t)\|_{\gamma(t)}\,\mathrm{d}t\,. \tag{3.13}$$

The concept of straight lines can then be generalized to *geodesics*, which is the shortest path between pairs of points $\boldsymbol{x},\boldsymbol{y}$ on the manifold $\gamma^*(\boldsymbol{x},\boldsymbol{y}) = \arg\min_\gamma L(\gamma)$ such that $\gamma(0) = \boldsymbol{x}$ and $\gamma(1) = \boldsymbol{y}$ with $\gamma$ traveling at constant speed (*i.e.*, $\|\gamma'(t)\|_{\gamma(t)} = c$, where $c$ is the distance). The concept of moving along a straight line with constant speed defines the *exponential map*, where for $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathcal{M}$ there is a unique unit speed geodesic $\gamma$ satisfying $\gamma(0) = \boldsymbol{x}$ with initial tangent vector $\boldsymbol{v}$, and the corresponding




(a) Parallel transport a tangent vector (arrow) from one location (black) to another (orange).

(b) Map the transported tangent vector (orange) to a point (green) in the hyperbolic space by using the exponential map.

**Figure 3.4**: Visualization of different mathematical operations in a hyperbolic space, that are used to define the wrapped distribution.

exponential map is defined by $\exp_{\boldsymbol{x}}(\boldsymbol{v}) = \gamma(1)$. We call the inverse of exponential map the *logarithm map* $\log_{\boldsymbol{x}} = \exp_{\boldsymbol{x}}^{-1} : \mathcal{M} \to \mathcal{T}_{\boldsymbol{x}}\mathcal{M}$, mapping from the manifold to the tangent space. For the Poincáre ball model, it is *geodesically complete* in the sense that $\exp_{\boldsymbol{x}}$ is well-defined on the full tangent space $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}$.

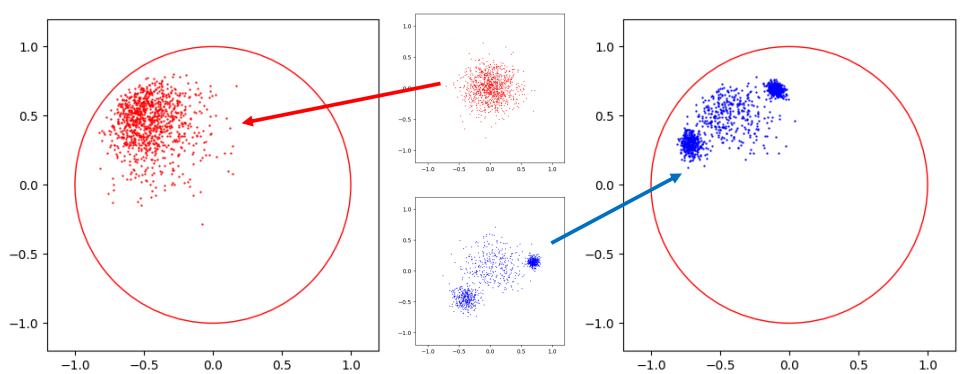



**Figure 3.5**: Mapping a Gaussian distribution (red) and a implicit distribution (blue) to the hyperbolic space.

### 3.7.2 Additional Related Work

**VAE with Adversarial Learning.** One of the first to apply adversarial learning to VAE is Adversarial Variational Bayes (AVB) [MNG17, PWH+17]. Motivated by Generative Adversarial Network (GAN) [GPAM+14], AVB introduces an auxiliary discriminator that transforms the maximum-likelihood-problem into a two-player game. Similarly, Adversarial Autoencoder (AAE) [MSJ+15] uses adversarial learning to match aggregated posterior with the prior. Based on this, Coupled Variational Bayes (CVB) [DDH+18] connects the primal-dual view of ELBO with adversarial learning, where the discriminator takes both data sample and latent code as input. This approach is also adopted in implicit VAE [FLG+19] for text generation. However, the prior used in implicit VAE is still standard Gaussian, while our proposed model uses hyperbolic geometry.

### 3.7.3 Metrics (Dialogue Response Generation)

BLEU [PRWZ02] is used to measure the amount of $n$-gram overlap between a generated response with the reference. Specifically, BLEU scores of $n < 4$ are computed; the average and the maximum scores are considered as $n$-gram precision and $n$-gram recall, respectively. In addition, the BOW embedding metric [LLS+16] is used to measure cosine similarity between bag-of-word embeddings of response and reference. Three metrics are considered for cosine distance: ($i$) computed by **greedily** matching words in two utterances; ($ii$) between the **averaged** embeddings in two utterances; and ($iii$) between the largest **extreme** values among the embeddings in two utterances. We also follow [GCHK19] and use the *distinct* metric to measure the diversity of generated text. $Dist-n$ is the ratio of unique $n$-grams over all $n$-grams in the generated sentences. *Intra-dist* and *inter-dist* are the average *distinct* values within each generated sample and among all generated samples, respectively.

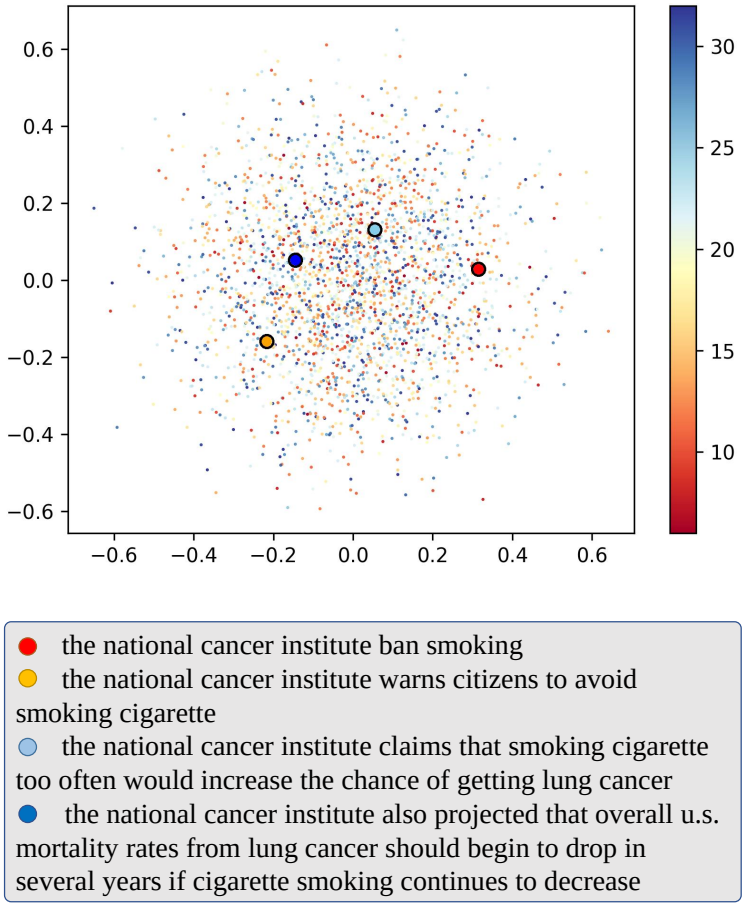
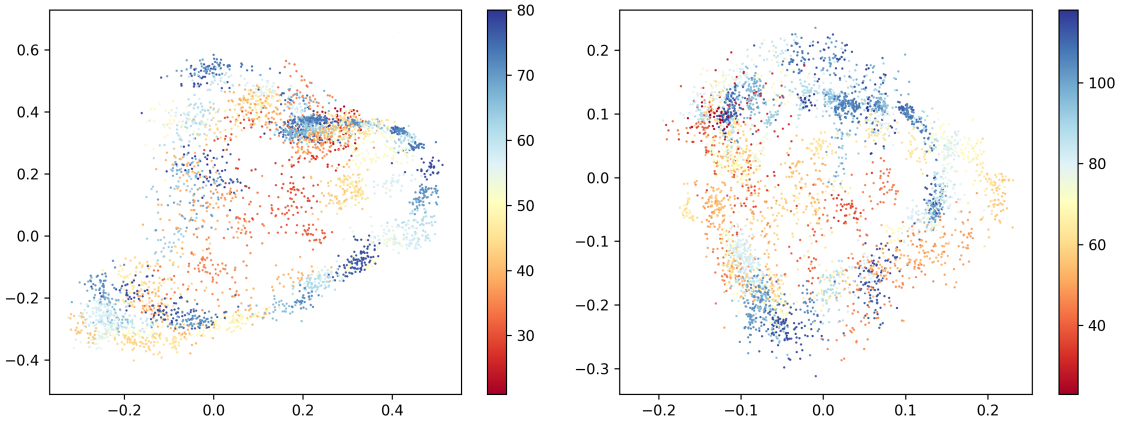



**Figure 3.6**: Visualization of the Euclidean space (VAE) of 5,000 randomly sampled sentences from the test set of PTB. The four listed sentences are created by modifying data samples.

### 3.7.4 Additional Implementation Details

For language modeling, we adopt the LSTM [HS97] for both the encoder and decoder, which have 256 hidden units for PTB, and 1024 hidden units for Yahoo and Yelp. The dimension of the latent code is set to 32. Following [MLM+19], the hyper-parameter $c$ is set to 0.7. We set the vocabulary size to 10,000 for PTB, and 20,000 for both Yahoo and Yelp. The word embedding size is 256 for PTB, and 512 for Yahoo and Yelp. For dialogue response generation, we follow [GCHK19], and use the GRU [CVMG+14] with 300 hidden units in each direction for both the response encoder and context




(a) Hyperbolic latent space for Yahoo.

(b) Hyperbolic latent space for Yelp.

**Figure 3.7**: Visualization of the hyperbolic latent space of 5,000 randomly sampled sentences from different datasets.

**Table 3.5**: Generate a sample based on the latent code of a specific input (PTB).

| Input | Sample |
|---|---|
| the national cancer institute ban smoking | the ⟨unk⟩ and drug administration were talking |
| the national cancer institute warns citizens to avoid smoking cigarette | the ⟨unk⟩ and drug administration officials also are used to ⟨unk⟩ |
| the national cancer institute claims that smoking cigarette too often would increase the chance of getting lung cancer | the u.s. and drug administration officials say they are n't ⟨unk⟩ to be used by the government |
| the national cancer institute also projected that overall u.s. mortality rates from lung cancer should begin to drop in several years if cigarette smoking continues to decrease | the u.s. and drug administration officials ⟨unk⟩ they are looking for ways to ¡unk¿ their own accounts for some of their assets to be sold by some companies |

encoder, and 300 hidden units for decoder. The latent code $\boldsymbol{z}$ has a dimension of 200. The size of the vocabulary is set to 10,000, and the word-embedding size is 200, initialized by GloVe [PSM14].

**Table 3.6**: Results on DailyDialog (P: precision, R: recall, A: average, E: extreme, G: greedy). Higher BLEU and BOW Embedding indicate better quality of generated responses. Higher intra/inter-dist means better generation diversity.

| Model | BLEU | | | BOW | | | Intra-dist | | Inter-dist | |
|---|---|---|---|---|---|---|---|---|---|---|
| | R | P | F1 | A | E | G | dist-1 | dist-2 | dist-1 | dist-2 |
| CVAE | 0.265 | 0.222 | 0.242 | 0.923 | 0.543 | 0.811 | 0.938 | 0.973 | 0.177 | 0.222 |
| CVAE-BOW | 0.256 | 0.224 | 0.239 | 0.923 | 0.540 | 0.812 | **0.947** | 0.976 | 0.165 | 0.206 |
| CVAE-CO | 0.259 | 0.244 | 0.251 | 0.914 | 0.530 | 0.818 | 0.821 | 0.911 | 0.106 | 0.126 |
| DialogWAE | 0.341 | **0.278** | **0.306** | 0.948 | 0.578 | 0.846 | 0.830 | 0.940 | 0.327 | 0.583 |
| iVAE | 0.355 | 0.239 | 0.285 | 0.951 | 0.609 | 0.872 | 0.897 | 0.975 | 0.501 | 0.868 |
| APo-VAE | **0.359** | 0.265 | 0.305 | **0.954** | **0.616** | **0.873** | 0.919 | **0.989** | **0.511** | **0.869** |

### 3.7.5 Additional Results

For language modeling, we plot the hyperbolic latent space for Yahoo and Yelp as shown in Figure 3.7. To demonstrate the hierarchical structure in the generated sentences (*i.e.*, the decoder), we choose 4 sentences (from short to long) with some hierarchy, listed on the left hand side of Table 3.5. These sentences are encoded to hyperbolic space by using a well trained APo-VAE. Then, we decode by randomly select a neighbor of each of the 4 latent codes. The output sentences are shown on the right hand side of Table 3.5, demonstrating similar hierarchy as the input sentences. Moreover, we directly measure the generation quality by using PPL and reverse PPL, shown in Table 3.7. Our APo-VAE achieves consistently better performance.

**Table 3.7**: Results on forward and reverse PPL on the PTB dataset.

| Model | Forward | Reverse |
|---|---|---|
| VAE | 18494 | 10149 |
| Cyc-VAE | 3390 | 5587 |
| $\beta$-VAE | 625 | 1897 |
| SA-VAE | 341 | 10651 |
| vMF-VAE | 274 | 2364 |
| iVAE | 116 | 1520 |
| APo-VAE | 109 | 1385 |

For dialog response generation, we include additional results on the DailyDialog dataset [LSS+17], which contains 13k daily conversations for an English learner in

**Table 3.8**: Human evaluation results on DailyDialog. Win/loss/tie indicates the percentage of responses generated by APo-VAE being better/worse/equal to the compared model.

| | vs iVAE | | | vs DialogWAE | | |
|---|---|---|---|---|---|---|
| | win | loss | tie | win | loss | tie |
| Informativeness | 45.4 | 26.9 | 17.7 | 46.1 | 26.5 | 27.4 |
| Coherence | 40.1 | 25.9 | 24.0 | 40.7 | 24.2 | 25.1 |
| Diversity | 43.9 | 30.8 | 25.3 | 47.5 | 31.4 | 21.1 |

daily life. We also provide examples of generated responses along with their corresponding dialog context in Table 3.9. Samples generated by APo-VAE are more relevant to the contexts than the baseline models. In addition, APo-VAE is capable of providing both positive and negative responses, demonstrating better generation diversity. The human evalutation results for DailyDialog can be found in Table 3.8.

**Table 3.9**: Examples of context-response pairs for different models. 'Eg.i' means the $i$-th sample.

| Context | DialogWAE | iVAE | APo-VAE |
|---|---|---|---|
| i would wondering if you like to stay here for a few days? | Eg.1: ok. | Eg.1: I'm not going to take a look. | Eg.1: good idea. how about shopping tomorrow morning? |
| | Eg.2: great, thanks for asking. | Eg.2: is it a problem? | Eg.2: of course. would you like to have lunch with me? |
| | Eg.3: next week is a good idea. | Eg.3: it is a day vacation for you here in the us. | Eg.3: i'm afraid i can't make it to the next few days. |
| i have to check out today. i'd like my bill ready by 10. | Eg.1: that's a long time. | Eg.1: that's the problem, sir? | Eg.1: i'm sorry to hear that. are you looking for another one? |
| | Eg.2: yes. | Eg.2: i was the only child in the room. | Eg.2: no problem. may i have it now? |
| | Eg.3: i'm afraid you can't do that. | Eg.3: you can get me a little time, and the red will take you get off. | Eg.3: i see, sir. what is the room number? |
| i wonder whether i could possibly borrow your new bicycle now. | Eg.1: that's a great idea. what did you do? | Eg.1: a problem. what do you think of this one? | Eg.1: of course. let's see. |
| | Eg.2: i'd like to tell you a fees, but i'm not sure i can do that. | Eg.2: yes sure, thanks. you are very tough. | Eg.2: i sorry to hear that. anything else? do you have any other account? |
| | Eg.3: oh, what is your problem? | Eg.3: what is the problem? what time? oh yes, it's been a long time. let me see. | Eg.3: i would love to do that. there have been more 1 for you. |

# Chapter 4

# Contrastively Learnt Latent Representation for Domain Adaptation

## 4.1 Introduction

Deep neural networks (DNNs) have significantly improved the state of the art on many supervised tasks [DJV+14, YCBL14, SZ15, HZRS16]. However, without sufficient training data, DNNs often generalize poorly to new tasks or new environments [TE11]. This is known as dataset bias or a domain-shift problem [GSH+09]. Unsupervised domain adaptation (UDA) [PY+10, GUA+16] aims to generalize a model learned from a source domain with rich annotated data to a new target domain without any labeled data. Recently, many approaches have been proposed to learn transferable representations, by simultaneously matching feature distributions across different domains [HFMC17, THDS15].

Motivated by [GPAM+14], [THSD17, GL14] introduced a min-max game: a domain discriminator is learned by minimizing the error of distinguishing data samples from the source and the target domains, while a feature generator learns transferable features that are indistinguishable by the domain discriminator. This imposes that the learned features are domain-invariant. Additionally, a feature classifier ensures that the learned features are discriminative in the source domain. Despite promising results, these adversarial methods suffer from inherent algorithmic weaknesses [SBNE18]. Specifically, the generator may manifest ambiguous features near class boundaries [SWUH18]: while the generator manages to fool the discriminator, some target-domain features may still be misclassified. In other words, the model

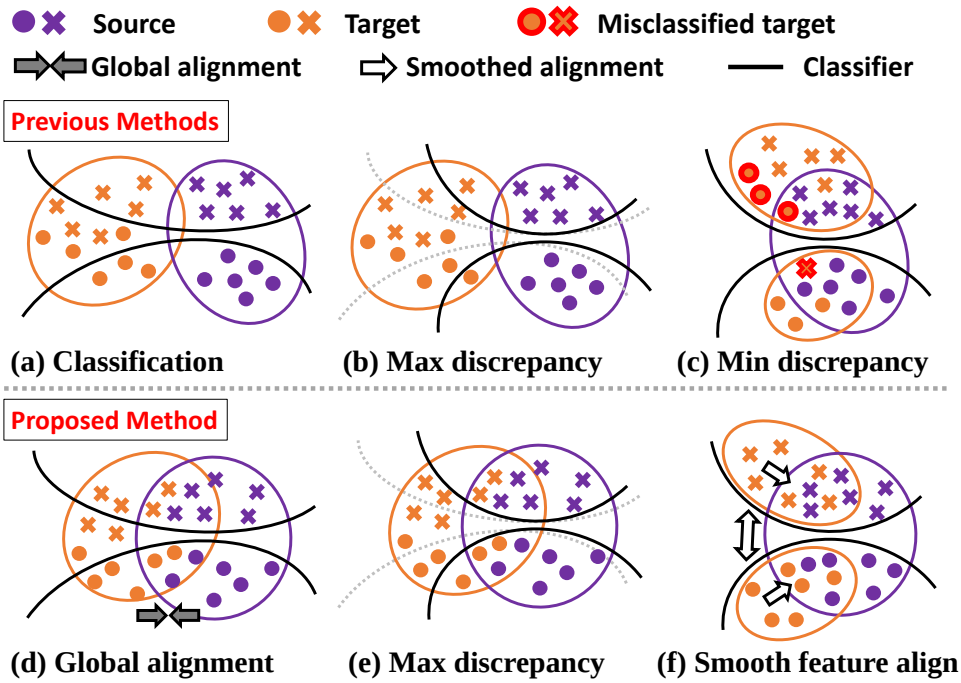



**Figure 4.1**: Comparison between previous classifier-discrepancy-based methods and our proposed CoSCA in the *feature* space. **Top:** The region of vacancy created by maximum discrepancy reduces the smoothness of alignment between ambiguous target samples and source samples, leading to sub-optimal solutions. This problem becomes more severe when global domain alignment is not considered. **Bottom:** Demonstration of global alignment and class-conditional adaptation by using the proposed CoSCA. After classifier discrepancy is maximized, the proposed contrastive loss moves ambiguous target samples near the decision boundary towards their neighbors and separates them from non-neighbors.

merely aligns the global marginal distribution of the two domains and ignores the class-conditional decision boundaries.

To overcome this issue, recent UDA models further align class-level distributions by taking the decision boundary into consideration. These methods either rely on iteratively refining the decision boundary with empirical data [SBNE18, SUHS17], or utilizing multi-view information [KSW+18]. Alternatively, the maximum classifier discrepancy (MCD) [SWUH18] model conducts a min-max game between a feature generator and two classifiers. Ambiguous target samples that are far from source-

domain samples can be detected when the discrepancy between the two classifiers is maximized, as shown in Figure 4.1(b). Meanwhile, as the generator fools the classifiers, the generated target features may fall into the source feature regions. However, the target samples may not be smooth on the low-dimensional manifold [CSZ09, LZL+17], meaning that neighboring samples may not belong to the same class. As a result, some generated target features could be miscategorized as shown in Figure 4.1(c).

In this paper, we propose the **Co**ntrastively **S**moothed **C**lass **A**lignment (CoSCA) model to improve the latent alignment of class-conditional feature distributions between source and target domains, by alternatively estimating the underlying label hypothesis of target samples to map them into tighter clusters, and adapt feature representations based on a proposed contrastive loss. Specifically, by aligning ambiguous target samples near the decision boundaries with their neighbors and distancing them from non-neighbors, CoSCA enhances the alignment of each class in a contrastive manner. Figure 4.1(f) demonstrates an enhanced and smoothed version of the class-conditional alignment. Moreover, as shown in Figure 4.1(d), Maximum Mean Discrepancy (MMD) is included to better merge the source and target domain feature representations. The overall framework is trained end-to-end in an adversarial manner.

Our principal contributions are summarized as follows:

- We propose CoSCA, a novel approach that smooths class alignment for maximizing classifier discrepancy with a contrastive loss. CoSCA also provides better global domain alignment via the use of MMD loss.

- We validate the proposed approach on several domain adaptation benchmarks. Extensive experiments demonstrate that CoSCA achieves state-of-the-art results on several benchmarks.

## 4.2 Related Work

**Unsupervised Domain Adaptation** A practical solution for domain adaptation is to learn domain-invariant features whose distribution is similar across the source and the target domains. For example, Sener *et al.* [SSSS16] proposed using clustering techniques and pseudo-labels to obtain discriminative features. Long *et al.* proposed DAN [LCWJ15] and JAN [LZWJ16] to minimize the MMD or variations of MMD between two domains. Adversarial domain adaptation integrates adversarial learning and domain adaptation in a two-player game [GL14, THSD17, THDS15]. Following this idea, most existing adversarial learning methods reduce feature differences by fooling a domain discriminator [LCWJ18a, GUA+16]. However, these methods fail to consider the relationship between target samples and the class-conditional decision boundaries when aligning features [SWUH18], while only merging the source and the target domains.

**Class-conditional Alignment** To address the aforementioned issue, recent work enforces class-level alignment while aligning global marginal distributions. Associative domain adaptation (ADA) [HFMC17] reinforces associations across domains directly in the embedding space, to extract features that are statistically domain-invariant and class-discriminative. Adversarial Dropout Regularization (ADR) [SUHS17] and Maximum Classifier Discrepancy (MCD) [SWUH18] were proposed to train a neural network in an adversarial manner, avoiding generating non-discriminative features lying in the region near the decision boundary. In [PCLW18, LZWJ16] the authors considered class information when measuring domain discrepancy. Co-regularized Domain Adaptation (Co-DA) [KSW+18] utilized multi-view information to match the marginal feature distributions corresponding to the class-conditional distributions. Compared with previous work that executed the alignment by optimizing "hard" metrics [SWUH18, KSW+18], we propose to smooth the alignment iteratively,

with explicitly defined loss.

**Contrastive Learning** The intuition for contrastive learning is to let the model understand the difference between one set (*e.g.*, data points) and another, instead of only characterizing a single set [ZHPA13]. This idea has been explored in previous works that model intra-class compactness and inter-class separability (*e.g.*, distinctiveness loss [DL17], contrastive loss [HCL06], triplet loss [WSL+14]) and tangent distance [RDV+11]. It has also been extended to consider several assumptions in semi-supervised and unsupervised learning [LZL+17, CXZZ17], such as the low-density region (or cluster) assumption [LZL+17, RDV+11] that the decision boundary should lie in the low-density region, rather than crossing the high-density region. Recently, contrastive learning was applied in UDA [KJYH19], in which the intra/inter-class domain discrepancy were modeled. In comparison, our work is based on the MCD framework, utilizing the low-density assumption and focusing on separating the ambiguous target data points by optimizing the contrastive objective, allowing the decision boundary to sit in the low-density region, *i.e.*, region of vacancy, and smoothness assumption.

## 4.3 Approach

Unsupervised domain adaptation seeks to generalize a learned model from a source domain to a target domain, the latter following a different (but related) data distribution from the former. Specifically, the source- and target-domain samples are denoted $\mathcal{S} = \{(\mathbf{x}_1^s, y_1^s), ..., (\mathbf{x}_i^s, y_i^s), ..., (\mathbf{x}_{N_s}^s, y_{N_s}^s)\}$, and $\mathcal{T} = \{\mathbf{x}_1^t, ..., \mathbf{x}_i^t, ..., \mathbf{x}_{N_t}^t\}$, respectively, where $\mathbf{x}_i^s$ and $\mathbf{x}_i^t$ are the input, and $y_i^s \in \{1, 2, ..., K\}$ represents the data labels of $K$ classes in the source domain. The target domain shares the same label types as the source domain, but we possess no labeled examples from the target domain. We are interested in learning a deep network $G$ that reduces domain shift in the data

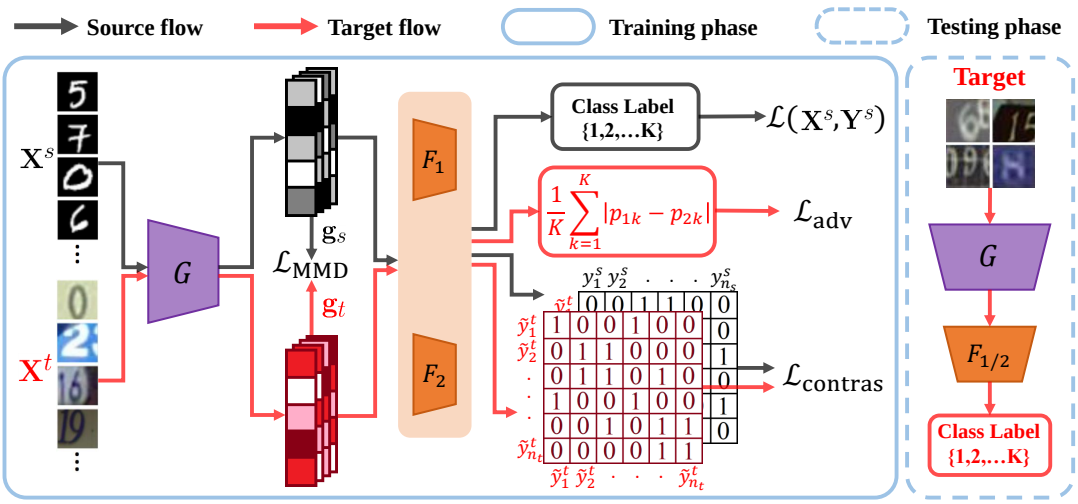



**Figure 4.2**: Framework of the proposed CoSCA. The inputs are $\mathbf{X}^s$ with label $\mathbf{Y}^s$ from the source domain and unlabeled $\mathbf{X}^t$ from the target domain. The model contains a shared feature generator $G$ and two feature classifiers $F_1$ and $F_2$. $\mathcal{L}_{\text{MMD}}$ is calculated using the generated feature mean of the source and target, *i.e.*, $g_s$ and $g_t$ respectively. $\mathcal{L}_{\text{adv}}$ is the classifier discrepancy calculated based on the probability outputs $p_1$ and $p_2$ of $F_1(G(\mathbf{X}^t))$ and $F_2(G(\mathbf{X}^t))$, respectively. $\mathcal{L}_{\text{contras}}$ is the contrastive loss calculated for both source-and-target and target-and-target samples.

distribution across $\mathcal{S}$ and $\mathcal{T}$, in order to make accurate predictions for $y_i^t$. We use the notation $(\mathbf{X}^s, \mathbf{Y}^s)$ to describe the source-domain samples and labels, and $\mathbf{X}^t$ for the unlabeled target-domain samples.

Adversarial domain adaptation approaches such as [SWUH18, KSGP19] achieve this goal via a two-step procedure: *i*) train a feature generator $G$ and the feature classifiers $F_1$, $F_2$ with the source-domain data, to ensure the generated features are class-conditional; *ii*) train $F_1$ and $F_2$ so that the prediction discrepancy between the two classifiers is maximized, and train $G$ to generate features that are distinctively separated. The maximum classifier discrepancy detects the target features that are far from the support of the source domain. As the generator tries to fool the classifiers (*i.e.*, minimizing the discrepancy), these target-domain features are enforced to be categorized and aligned with the source-domain features.

However, only measuring divergence between $F_1$ and $F_2$ can be considered first-order moment matching, which may be insufficient for adversarial training. Previous work also observed similar issues [AGL+17, THS+18]. We address this challenge by adding the Maximum Mean Discrepancy (MMD) loss, that matches the difference via higher-order moments. Also, the class alignment in existing UDA methods takes into account the intra-class domain discrepancy only, which makes it difficult to separate samples within the same class that are close to the decision boundary. Thus, in addition to the discrepancy loss, we also measure both intra- and inter-class discrepancy across domains. Specifically, we propose to minimize the distance among target-domain features that fall into the same class based on decision boundaries, and separate those features from different categories. During this process, ambiguous target features are simultaneously kept away from the decision boundaries and mapped into the high-density region, achieving better class alignment.

### 4.3.1 Global Alignment with MMD

Following [SWUH18], we first train a feature generator $G(\cdot)$ and two classifiers $F_1(G(\cdot))$ and $F_2(G(\cdot))$ to minimize the softmax cross-entropy loss using the data from the labeled source domain $\mathcal{S}$, defined as:

$$\mathcal{L}(\mathbf{X}^s, \mathbf{Y}^s) = -\,\mathbb{E}_{(\mathbf{x}^s, y^s)\sim(\mathbf{X}^s, \mathbf{Y}^s)} \Big[ \sum_{k=1}^{K} \mathbb{1}_{[k=y^s]} \log p_1(\mathbf{y}|\mathbf{x}^s) + \sum_{k=1}^{K} \mathbb{1}_{[k=y^s]} \log p_2(\mathbf{y}|\mathbf{x}^s) \Big] \tag{4.1}$$

where $p_1(\mathbf{y}|\mathbf{x})$ and $p_2(\mathbf{y}|\mathbf{x})$ are the probabilistic output of the two classifiers $F_1(G(\mathbf{x}))$ and $F_2(G(\mathbf{x}))$, respectively.

In addition to (4.1), we explicitly minimize the distance between the source and target feature distributions with MMD. The main idea of MMD is to estimate the

distance between two distributions as the distance between sample means of the projected embeddings in a Hilbert space. Minimizing MMD is equivalent to minimizing all orders of moments [GBR+12]. In practice, the squared value of MMD is estimated with empirical kernel mean embeddings:

$$\begin{aligned} \mathcal{L}_{\text{MMD}}(\mathbf{X}^s, \mathbf{X}^t) &= \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} k(\phi(\frac{\mathbf{g}_s}{||\mathbf{g}_s||}), \phi(\frac{\mathbf{g}_t}{||\mathbf{g}_t||})) \\ \mathbf{g}_s = \frac{1}{n_s} \sum_{i=1}^{n_s} G(\mathbf{x}_i^s), &\quad \mathbf{g}_t = \frac{1}{n_t} \sum_{i=1}^{n_t} G(\mathbf{x}_i^t) \end{aligned} \tag{4.2}$$

where $\phi(\cdot)$ is the kernel mapping, $\mathbf{g}_s \in \mathcal{R}^n$, $\mathbf{g}_t \in \mathcal{R}^n$, with $n_t$ and $n_s$ denoting the size of a training mini-batch of the data from the source domain $\mathcal{S}$ and the target domain $\mathcal{T}$, respectively; $|| \cdot ||$ denotes the $\ell_2$-norm. With the MMD loss $\mathcal{L}_{\text{MMD}}$, the normalized features in the two domains are encouraged to be distributed identically, leading to better global domain alignment.

### 4.3.2 Contrastively Smoothed Class Alignment

**Discrepancy Loss** The discrepancy loss represents the level of disagreement between the two feature classifiers in prediction for target-domain samples. Specifically, the discrepancy loss between $F_1$ and $F_2$ is defined as:

$$d(p_1(\mathbf{y}|\mathbf{x}), p_2(\mathbf{y}|\mathbf{x})) = \frac{1}{K} \sum_{k=1}^{K} \Big| p_{1_k}(\mathbf{y}|\mathbf{x}) - p_{2_k}(\mathbf{y}|\mathbf{x}) \Big| \tag{4.3}$$

where $| \cdot |$ denotes the $\ell_1$-norm, and $p_{1_k}(\cdot)$ and $p_{2_k}(\cdot)$ are the probability output of $p_1$ and $p_2$ for the $k$-th class, respectively. Accordingly, we can define the discrepancy loss over the target domain $\mathcal{T}$:

$$\mathcal{L}_{\text{adv}}(\mathbf{X}^t) = \mathbb{E}_{\mathbf{x}^t \sim \mathbf{X}^t} \big[ d(p_1(\mathbf{y}|\mathbf{x}^t), p_2(\mathbf{y}|\mathbf{x}^t)) \big] \tag{4.4}$$

Adversarial training is conducted in the MCD [SWUH18] setup:

$$\begin{aligned} &\min_{F_1,F_2} \mathcal{L}(\mathbf{X}^s, \mathbf{Y}^s) - \lambda \mathcal{L}_{\text{adv}}(\mathbf{X}^t) \\ &\min_{G} \mathcal{L}_{\text{adv}}(\mathbf{X}^t) \end{aligned} \tag{4.5}$$

where $\lambda$ is a hyper-parameter. Minimizing the discrepancy between the two classifiers $F_1$ and $F_2$ induces smoothness for the clearly classified target-domain features, while the region in the vacancy among the ambiguous ones remains non-smooth. Moreover, MCD only utilizes the unlabeled target-domain samples, while ignoring the labeled source-domain data when estimating the discrepancy.

**Contrastive Loss** To further optimize $G$ to estimate the underlying label hypothesis of target-domain samples, we propose to measure the intra- and inter-class discrepancy across domains, conditional on class information. By using an indicator defined as $c(y, y') = \begin{cases} 1, & y = y' \\ 0, & y \neq y' \end{cases}$, we define the contrastive loss between $\mathcal{S}$ and $\mathcal{T}$ as:

$$\mathcal{L}_{\text{contras}}^{\mathcal{S}\leftrightarrow\mathcal{T}} = \sum_{\mathbf{x}_i^s \in \mathcal{S}, \mathbf{x}_j^t \in \mathcal{T}} L_{\text{dis}}(G(\mathbf{x}_i^s), G(\mathbf{x}_j^t), c(y_i^s, \widetilde{y}_j^t)) \tag{4.6}$$

where $L_{\text{dis}}$ is a distance measure (defined below), and $\widetilde{y}_j^t$ is the predicted target label for $\mathbf{x}_j^t$. Specifically, (4.6) covers two types of class-aware domain discrepancies: *i*) intra-class domain discrepancy ($y_i^s = \widetilde{y}_j^t$); and *ii*) inter-class domain discrepancy ($y_i^s \neq \widetilde{y}_j^t$). Note that $y_i^s$ is known, providing some supervision for parameter learning. Similarly, we can define the constrastive loss between $\mathcal{T}$ and $\mathcal{T}$ as:

$$\mathcal{L}_{\text{contras}}^{\mathcal{T}\leftrightarrow\mathcal{T}} = \sum_{\mathbf{x}_i^t, \mathbf{x}_j^t \in \mathcal{T}} L_{\text{dis}}(G(\mathbf{x}_i^t), G(\mathbf{x}_j^t), c(\widetilde{y}_i^t, \widetilde{y}_j^t)) \tag{4.7}$$

To obtain the indicator $c(y, y')$, estimated target label $\widetilde{y}_i^t$ is required. Specifically, for each data sample $\mathbf{x}_j^t$, a pseudo label is predicted based on the maximum posterior

probability of the two classifiers:

$$\widetilde{y}_j^t = \underset{k \in \{1,2,\ldots,K\}}{\arg\max} \Big\{ p(F_1(G(\mathbf{x}_j^t)) = k|\mathbf{x}) + p(F_2(G(\mathbf{x}_j^t)) = k|\mathbf{x}) \Big\} \tag{4.8}$$

Ideally, based on the indicator, $L_{\text{dis}}$ should ensure the gathering of features that fall in the same class, while separating those in different categories. Following [LZL+17], we utilize contrastive Siamese networks [BGL+94], which can learn an invariant mapping to a smooth and coherent feature space and perform well in practice:

$$L_{\text{dis}} = \begin{cases} ||G(\mathbf{x}_i) - G(\mathbf{x}_j)||^2 & c_{ij} = 1 \\ \max(0, m - ||G(\mathbf{x}_i) - G(\mathbf{x}_j)||)^2 & c_{ij} = 0 \end{cases} \tag{4.9}$$

where $c_{ij} = c(y_i, y_j)$ and $m$ is a pre-defined margin. The margin loss constrains the neighboring features to be consistent. Based on the above definitions of source-and-target and target-and-target contrastive losses, the overall objective is:

$$\mathcal{L}_{\text{contras}}(\mathbf{X}^s, \mathbf{Y}^s, \mathbf{X}^t) = \mathcal{L}_{\text{contras}}^{\mathcal{S} \leftrightarrow \mathcal{T}} + \mathcal{L}_{\text{contras}}^{\mathcal{T} \leftrightarrow \mathcal{T}} \tag{4.10}$$

Minimizing the contrastive loss $\mathcal{L}_{\text{contras}}$ encourages features in the same class to aggregate together while pushing unrelated pairs away from each other. In other words, the semantic feature approximation is enhanced to induce smoothness between data in the feature space.

### 4.3.3 Training Procedure

We optimize $G$, $F_1$ and $F_2$ by combining all of the aforementioned losses, performed in an adversarial training manner. Specifically, we first train the classifiers $F_1$ and $F_2$ and the generator $G$ to minimize the objective:

$$\min_{F_1, F_2, G} \mathcal{L}(\mathbf{X}^s, \mathbf{Y}^s) + \lambda_1 \mathcal{L}_{\text{MMD}}(\mathbf{X}^s, \mathbf{X}^t) \tag{4.11}$$

**Algorithm 2** Training procedure of CoSCA.

**Input**: Source domain samples $\{\mathbf{x}_i^s, y_i^s\}$, and target domain samples $\{\mathbf{x}_j^t\}$. Hyper-parameters $\lambda_1$, $\lambda_2$, $\lambda_3$, and inner-loop iteration $\tau$ and $\delta$.
**Output**: Classifiers $F_1$ and $F_2$, and generator $G$.

---

**for** *iter* from 1 to *max_iter* **do**
  Sample a mini-batch of source samples $[\mathbf{x}_i^s, y_i^s]$ and target samples $\left[\mathbf{x}_j^t\right]$.
  ***# Update both the generator and the classifiers***
  Compute $\mathcal{L}(\mathbf{X}^s, \mathbf{Y}^s)$ on $[\mathbf{x}_i^s, y_i^s]$.
  Compute $\mathcal{L}_{\text{MMD}}(\mathbf{X}^s, \mathbf{X}^t)$ on $\left[\mathbf{x}_i^s, \mathbf{x}_j^t\right]$.
  Update $G$, $F_1$ and $F_2$ using (4.11).
  ***# Update the classifiers***
  **for** *inner_loop_iter*$_1$ from 1 to $\tau$ **do**
    Compute $\mathcal{L}(\mathbf{X}^s, \mathbf{Y}^s)$ on $[\mathbf{x}_i^s, y_i^s]$.
    Compute $\mathcal{L}_{\text{adv}}(\mathbf{X}^t)$ on $\mathbf{x}_j^t$.
    Fix $G$, update $F_1$ and $F_2$ using (4.12).
  **end for**
  ***# Update the feature generator***
  **for** *inner_loop_iter*$_2$ from 1 to $\delta$ **do**
    Compute $\mathcal{L}_{\text{adv}}(\mathbf{X}^t)$ on $\mathbf{x}_j^t$.
    Compute $\mathcal{L}_{\text{contras}}(\mathbf{X}^s, \mathbf{Y}^s, \mathbf{X}^t)$ on $\left[\mathbf{x}_i^s, y_i^s, \mathbf{x}_j^t\right]$.
    Fix $F_1$ and $F_2$, update $G$ using (4.13).
  **end for**
**end for**

---

We then train the classifiers $F_1$ and $F_2$ while keeping the generator $G$ fixed. The objective is:

$$\min_{F_1, F_2} \ \mathcal{L}(\mathbf{X}^s, \mathbf{Y}^s) - \lambda_2 \mathcal{L}_{\text{adv}}(\mathbf{X}^t) \tag{4.12}$$

Lastly, we train the generator $G$ with the following objective, while keeping both $F_1$ and $F_2$ fixed:

$$\min_{G} \ \lambda_2 \mathcal{L}_{\text{adv}}(\mathbf{X}^t) + \lambda_3 \mathcal{L}_{\text{contras}}(\mathbf{X}^s, \mathbf{Y}^s, \mathbf{X}^t) \tag{4.13}$$

where $\lambda_1$, $\lambda_2$ and $\lambda_3$ are hyper-parameters that balance the different objectives. These steps are repeated, with the full approach summarized in Algorithm 2. In our experiments, the inner-loop iteration numbers $\tau$ and $\delta$ are both set to 2.

**Class-aware sampling** When training with the contrastive loss, it is important to sample a mini-batch of data with all the classes, to allow (4.10) to be fully trained. Following [KJYH19], we use a class-aware sampling strategy. Specifically, we randomly select a subset of each class, from which a mini-batch is sampled. Consequently, in each mini-batch, we are able to estimate the intra/inter-class discrepancy.

**Dynamic parameterization of $\lambda_3$** In our implementation, we adapt a dynamic $\omega(t)$ to parameterize $\lambda_3$. We set $\omega(t) = \exp[-\theta(1 - \frac{t}{\text{max-epochs}})]\lambda_3$, which is a Gaussian curve ranging from 0 to $\lambda_3$; this is employed to prevent unlabeled target features gathering in the early stage of training, as the pseudo labels might be unreliable.

## 4.4 Experiments

We evaluate the proposed model primarily on image datasets. To compare with MCD [SWUH18] as well as the state-of-the-art results in [SBNE18, KSW+18], we evaluate on the same datasets used in those studies: the digit datasets (*i.e.*, MNIST, MNISTM, Street View House Numbers (SVHN), and USPS), CIFAR-10, and STL-10. We also conduct experiments on the VisDA dataset, *i.e.*, large-scale images. Our model can also be applied to non-visual domain adaptation tasks. Specifically, to show the flexibility of our model, we also evaluate it on the Amazon Reviews dataset.

For visual domain adaptation tasks, the proposed model is implemented based on VADA [SBNE18] and Co-DA [KSW+18] to avoid any incidental difference caused by network architecture. However, different from these methods, our model does not require a discriminator, and only adopts the architecture for the feature generator $G$ and the classifier $F$. Specifically, $G$ has 9 convolutional layers with several dropout, max-pool, Gaussian noise and global pool layers. Both $F_1$ and $F_2$ are one-layer MLPs. We also include instance normalization [SBNE18, UVL16], achieving superior results on several benchmarks. For the VisDA dataset, we implemented our model based on

(a) The Digit dataset.

(b) The CIFAR-10 dataset and the STL dataset.

**Figure 4.3**: Sample images from the Digit, CIFAR-10 and STL datasets. Images from each column belong to the same class, while each row corresponds to a domain.

Self-ensembling Domain Adaptation (SEDA) [FMF18]. To compare with MCD and Contrastive Adaptation Network (CAN) [KJYH19] (codebase not available) in both experiments, we re-implemented them using the exact architecture as our model.

In addition to the aforementioned baseline models, we also include results from recently proposed unsupervised domain adaptation models. Note that standard domain adaptation methods (such as Transfer Component Analysis (TCA) [PTKY11] and Subspace Alignment (SA) [FHST13]) are not included; these models only work on pre-extracted features, and are often not scalable to large datasets. Instead, we mainly compare our model with methods based on adversarial neural networks.

For the non-visual task, we adopt a one-layer CNN structure from previous work [Kim14]. The feature generator $G$ consists of three components, including a 300-dimensional word embedding layer using GloVe [PSM14], a one-layer CNN with ReLU, and a max-over-time pooling through which the final sentence representation is obtained. The classifiers $F_1$ and $F_2$ can be decomposed into one dropout layer and one fully connected output layer.

**Table 4.1**: Results on visual domain adaptation tasks. Source-Only corresponds to training a classifier in the source domain and applying it directly to the target domain, without any adaptation. Models with instance-normalized input are implemented using the same network architecture. Results with † are reported in [SWUH18].

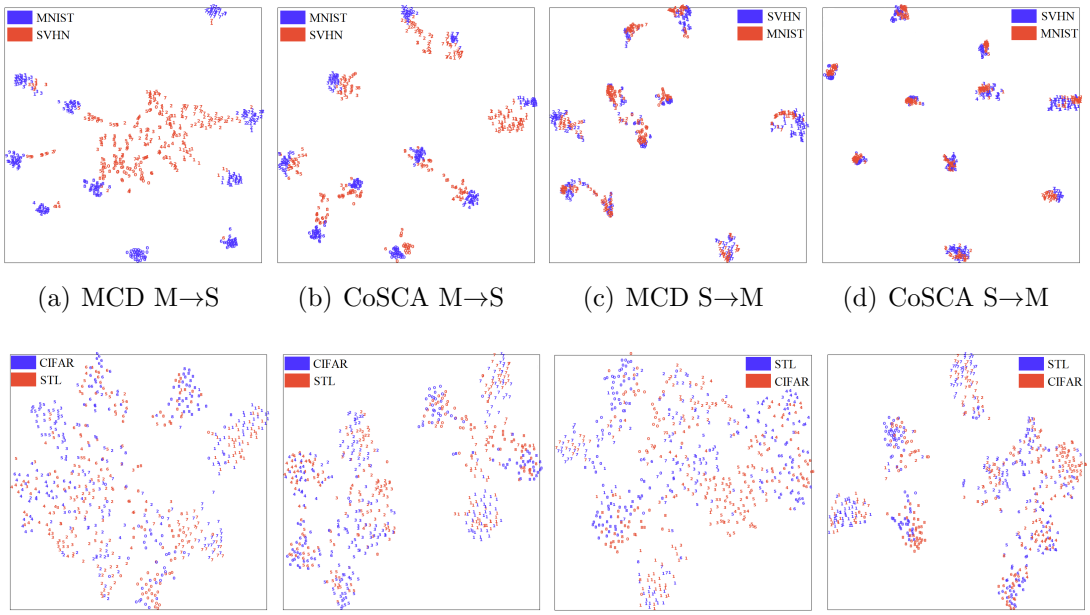
| Source Domain<br>Target Domain | MNIST<br>SVHN | SVHN<br>MNIST | MNIST<br>MNISTM | MNIST<br>USPS | CIFAR<br>STL | STL<br>CIFAR |
|---|---|---|---|---|---|---|
| MMD [LCWJ15] | - | 71.1 | 76.9 | 81.1 | - | - |
| DANN [GUA+16] | 35.7 | 71.1 | 81.5 | 77.1 | - | - |
| DSN [BTS+16] | 40.1 | 82.7 | 83.2 | 91.3 | - | - |
| ATT [SUH17] | 52.8 | 86.2 | 94.2 | - | - | - |
| With Instance-Normalized Input: | | | | | | |
| Souce-Only | 40.9 | 82.4 | 59.9 | 76.7 | 77.0 | 62.6 |
| VADA [SBNE18] | 73.3 | 94.5 | 95.7 | - | 78.3 | 71.4 |
| Co-DA [KSW+18] | **81.3** | 98.6 | 97.3 | - | 80.3 | 74.5 |
| MCD [SWUH18] | 68.7 | 96.2† | 96.7 | 94.2† | 78.1 | 69.2 |
| SEDA [FMF18] | 37.5 | **99.2** | - | 98.2 | 80.1 | 74.2 |
| CAN [KJYH19] | 67.1 | 94.8 | 96.2 | 97.5 | 77.3 | 70.4 |
| CoSCA | 80.7 | 98.7 | **98.9** | **99.3** | **81.7** | **75.2** |

### 4.4.1 Digit Datasets

There are four types of digit images (*i.e.*, four domains). MNIST and USPS are both hand-written gray-scale images, with relatively small domain difference. MNISTM [GUA+16] is a dataset built upon MNIST by adding randomly colored image patches from BSD500 dataset [AMFM11]. SVHN includes colored images of street numbers. All images are rescaled to $32 \times 32 \times 3$. Sample images of all four digit datasets are presented in Figure 4.3(a).

**MNIST→SVHN** While both MNIST and SVHN include images of digits, there exists a large domain gap between these two datasets. As gray-scale handwritten digits, MNIST has much lower dimensionality than SVHN, which contains cropped street-view images of house numbers. Specifically, each image from SVHN has a colored background, which sometimes contains multiple digits, and might be blurry. This makes MNIST→SVHN a much harder adaptation task than other digit datasets.

It is shown recently in [SBNE18] that instance normalization allows the classifier to be invariant to channel-wide scaling and shifting of the input pixel intensities, greatly improving the adaptation performance on MNIST→SVHN (73.3%). With instance normalization, our proposed CoSCA achieves test accuracy of 80.7%, as shown in Table 4.1, competitive with state-of-the-art results from [KSW+18].




(a) MCD M→S (b) CoSCA M→S (c) MCD S→M (d) CoSCA S→M

(e) MCD C→STL (f) CoSCA C→STL (g) MCD STL→C (h) CoSCA STL→C

**Figure 4.4**: t-SNE embedding of the features $G(x)$ for MNIST (M) → SVHN (S) and STL → CIFAR (C). Color indicates domain, and the digit number is the label. The ideal situation is to mix the two colors with the same label, representing domain-invariant features.

Notice that MCD does not provide adequate performance. Figure 4.4(a) plots the t-SNE embedding of the features learned by MCD. Domains are indicated by different colors, and classes are indicated by different digit numbers. MCD fails to align the features of the two domains globally due to the large domain gap. In other words, the maximized discrepancy provides too many ambiguous target-domain samples. As a result, the feature generator may not be able to properly align them with the source-domain samples. In comparison, as shown in Figure 4.4(b), CoSCA utilizes the

MMD between the source and the target domain features, thus maintaining a better global domain alignment. With further smoothed class-conditional adaptation, it outperforms MCD.

**SVHN→MNIST** Classification with the MNIST dataset is easier than others. As shown in Table 4.1, source-only achieves 82.4% on SVHN→MNIST with instance normalization. Therefore, even with the same amount of domain difference, performance on SVHN→MNIST is much better than MNIST→SVHN across all compared models. The test accuracy of our model achieves 98.7%.

**MNIST→MNISTM** Since MNISTM is a colored version of MNIST, there exists a one-to-one matching between the two datasets, *i.e.*, a domain adaptation model would perform well as long as domain-invariant features are properly extracted. CoSCA provides better results than Co-DA, yielding a test accuracy of 98.9%.

**MNIST→USPS** Evaluation on MNIST and USPS datasets is also conducted to compare our model with other baselines. The proposed method achieves an excellent result of 99.3%.

### 4.4.2 CIFAR-10 and STL-10 Datasets

CIFAR-10 and STL-10 are both 10-class datasets, with each image containing an animal or a type of transportation. Images from each class are much more diverse than the digit datasets, with higher intrinsic dimensionality, which makes it a harder domain adaptation task. There are 9 overlapping classes between these two datasets. Figure 4.3(b) shows some sample images from each class. CIFAR provides images of size 32×32 and a large training set of 50,000 image samples, while STL contains higher quality images of size 96×96, but with a much smaller training set of 5,000 samples. Following [FMF18, SBNE18, KSW$^+$18], we remove non-overlapping classes from these two datasets and resize the images from STL to 32×32.

**Table 4.2**: Test accuracy of ResNet101 model fine-tuned on the VisDA dataset. Results with † are reported in [SWUH18], while the others are implemented using the same network architecture.

| Model | plane | bcycl | bus | car | horse | knife | mcycl | person | plant | sktbrd | train | truck | mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Source Only | 55.1 | 53.3 | 61.9 | 59.1 | 80.6 | 17.9 | 79.7 | 31.2 | 81.0 | 26.5 | 73.5 | 8.5 | 52.4† |
| MMD | 87.1 | 63.0 | 76.5 | 42.0 | 90.3 | 42.9 | 85.9 | 53.1 | 49.7 | 36.3 | 85.8 | 20.7 | 61.1† |
| DANN | 81.9 | 77.7 | 82.8 | 44.3 | 81.2 | 29.5 | 65.1 | 28.6 | 51.9 | 54.6 | 82.8 | 7.8 | 57.4† |
| MCD | 89.1 | 80.8 | 82.9 | 70.9 | 91.6 | 56.5 | 89.5 | 79.3 | 90.9 | 76.1 | 88.3 | 29.3 | 77.1 |
| CAN | 91.4 | 78.9 | 79.1 | 72.8 | 93.2 | 63.4 | 82.4 | 68.6 | 93.2 | 88.3 | 84.1 | **39.2** | 77.9 |
| SEDA | 95.3 | 87.1 | 84.2 | 58.3 | 94.4 | **89.6** | 87.9 | 79.1 | 92.8 | **91.3** | **89.6** | 37.4 | 82.2 |
| CoSCA | **95.7** | **87.4** | **85.7** | **73.5** | **95.3** | 72.8 | **91.5** | **84.8** | **94.6** | 87.9 | 87.9 | 36.8 | **82.9** |

STL→CIFAR is more difficult than CIFAR→STL, due to the small training set in STL. For the latter, the source-only model with no adaptation involved achieves an accuracy of 77.0%. With adaptation, the margin-of-improvement is relatively small, while CoSCA provides the best improvement of 4.7% among all the models (Table 4.1). For STL→CIFAR, our model yields a 12.6% margin-of-improvement and an accuracy of 75.2%. Figures 4.4(e), 4.4(f), 4.4(g), and 4.4(h) provide t-SNE plots for MCD and our model, respectively, which shows our model achieves much better alignment for each class.

### 4.4.3 VisDA Dataset

The VisDA dataset is a large-scale image dataset that evaluates the adaptation from synthetic-object to real-object images. Images from the source domain are synthetic renderings of 3D models from different angles and lighting conditions. There are 152,397 image samples in the source domain, and 55,388 image samples in the target domain. The image size, after rescaling as in [SWUH18], is $224 \times 224 \times 3$. A model architecture with ResNet101 [HZRS16] pre-trained on Imagenet is required. There are 12 different object categories in VisDA, shared by the source and the target domains.

**Table 4.3**: Results on the Amazon Reviews dataset. Results with † are reported by [HLND18, ZR18].

| | Source-Only | DANN [GUA+16] | PBLM [ZR18] | MCD [SWUH18] | DAS [HLND18] | CoSCA |
|---|---|---|---|---|---|---|
| Accuracy | 79.13 | 80.29† | 80.40† | 81.35 | 81.96† | **83.17** |

Table 4.2 shows the test accuracy of different models in all object classes. The class-aware methods, namely MCD [SWUH18], SEDA [FMF18] and our proposed CoSCA, outperform the source only model in all categories. In comparison, the methods that are mainly based on distribution matching do not perform well in some of the categories. CoSCA outperforms MCD, showing the effectiveness of contrastive loss and MMD global alignment. In addition, it performs better than SEDA in most categories, demonstrating its robustness in handling large scale images.

### 4.4.4 Amazon Reviews Dataset

We also evaluate CoSCA on the Amazon Reviews dataset collected by Blitzer *et al.* [BDP07]. It contains reviews from several different domains, with 1000 positive and 1000 negative reviews in each domain.

Table 4.3 shows the average classification accuracy of different methods, including DANN [GUA+16] DAS [HLND18] and PBLM [ZR18]. We use the same model architecture and parameter setting for MCD and the source-only model. Results show that the proposed CoSCA outperforms all other methods. Specifically, it improves the performance from test accuracy of 81.96% to 83.17%, when compared to the state-of-the-art method DAS. MCD achieves 81.35%, which is also outperformed by CoSCA.

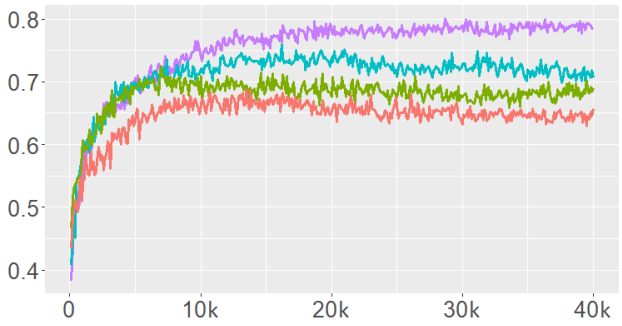
**Table 4.4**: Ablation study on CoSCA with different variations of MCD on MNIST→SVHN, STL→CIFAR, and Amazon Reviews.

| Model | MNIST SVHN | STL CIFAR | Amazon Reviews |
|---|---|---|---|
| MCD [SWUH18] | 68.7 | 69.2 | 81.35 |
| MCD+MMD | 72.1 | 70.2 | 81.73 |
| MCD+Contras | 75.9 | 73.4 | 82.56 |
| CoSCA | **80.7** | **75.2** | **83.17** |

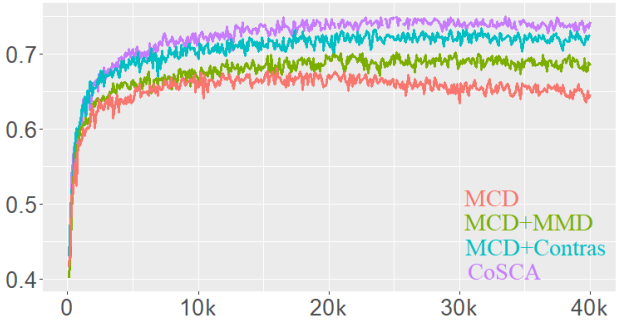
### 4.4.5 Ablation Study

To further demonstrate the improvement of CoSCA over MCD [SWUH18], we conduct an ablation study. Specifically, with the same network architecture and setup, we compare model performance among 1) MCD, 2) MCD with smooth alignment (MCD+Contras), 3) MCD with global alighnment (MCD+MMD), and 4) CoSCA, to validate the effectiveness of adding contrastive loss $\mathcal{L}_{\text{contras}}$ and MMD loss $\mathcal{L}_{\text{MMD}}$ to MCD. As MCD has already achieved superior performance on some of the benchmark datasets, we mainly choose those tasks on which MCD does not perform very well, in order to better analyze the margin of improvement. Therefore, MNIST→SVHN, STL→CIFAR and Amazon Reviews are selected for this experiment (Table 4.4).

**Effect of Contrastive Alignment** We compare CoSCA with MCD as well as its few variations, to validate the effectiveness of the proposed contrastive alignment. Table 4.4 provides the test accuracy for every model across the selected benchmark datasets. For MNIST→SVHN, MCD+Contrastive outperforms MCD by 7.2%. For STL→CIFAR and Amazon Reviews, the margin of improvement is 4.2% and 1.21%, respectively (less significant than MNIST→SVHN, possibly due to the smaller domain difference). Note that the results of MCD+Contras are still worse than CoSCA, demonstrating the effectiveness of the global domain alignment and the framework design of our model.




(a) Validation accuracy vs number of iterations for MNIST→SVHN.




(b) Validation accuracy vs number of iterations for STL→CIFAR.

**Figure 4.5**: Ablation study on CoSCA and different variations of MCD.

**Effect of MMD** We further investigate how the MMD loss can impact the performance of our proposed CoSCA. Specifically, for MNIST→SVHN, MCD+MMD achieves a test accuracy of 72.1%, only lifting the original result of MCD by 3.4%. For STL→CIFAR and Amazon Reviews, the margin-of-improvement is 1.0% and 0.38%, respectively. While this validates the effectiveness of having global alignment in the MCD framework, the improvement is small. Without a smoothed class-conditional alignment, MCD still encounters misclassified target features during training, leading to a sub-optimal solution. Notice that when comparing CoSCA with MCD+Contras, the improvement for MNIST→SVHN is significant (as shown in Figure 4.5(a)), with validation accuracy and training stability enhanced. This demonstrates the importance of global alignment when there exists a large domain difference.

## 4.5 Conclusions

We have proposed Contrastively Smoothed Class Alignment (CoSCA) for the UDA problem, by explicitly combining intra-class and inter-class domain discrepancy and optimizing class alignment through end-to-end mini-batch training. Experiments on several benchmarks demonstrate that our model can outperform state-of-the-art baselines. Our experimental analysis shows that CoSCA learns more discriminative target domain features, and the introduced MMD feature matching improves the global domain alignment. For future work, we will extend our model to other domain-adaptation tasks. Another direction to explore concerns development of a theoretical interpretation of contrastive learning for domain adaptation, particularly characterizing its effects on the alignment of source and target domain feature distributions.

# Chapter 5

# Bridging Domain Gap by Learning Intermediate Latent Representation

## 5.1 Introduction

With advances in supervised deep learning, many vision problems have realized significant performance improvements [KSH12, SZ15, SLJ+15, HZRS16, GDDM14, RHGS15, SLD17, CPK+18]. While the success is driven by several factors, such as improved deep learning architectures [HZRS16, HSS18] or optimization techniques [DHS11, KB14, IS15], it is strongly dependent on the existence of large-scale labeled training data [DDS+09]. Unfortunately, such a dataset may not be available for each application domain. This demands new ways of knowledge transfer from existing labeled data to individual target applications, potentially with access to large-scale *unlabeled* data from the application domain.

Unsupervised domain adaptation (UDA) [BDBCP07, BDBC+10] has been proposed to improve the generalization ability of classifiers, using unlabeled data from the target domain. Deep domain adaptation that realizes UDA in a deep learning framework has been successful in several vision tasks [GUA+16, CLS+18, IFYA18, HTP+17, TSSC19, PTS+20]. The core idea is to reduce the discrepancy metric between the two domains, measured by the domain discriminator [GUA+16] or MMD kernel [THZ+14] at certain representation of deep networks. Ideally, the discriminator learns the transformation mechanisms between the two domains. However, it could be difficult to model such dynamics when there are many factors of inter- and intra-domain variation applied to transform the source domain into the target domain.

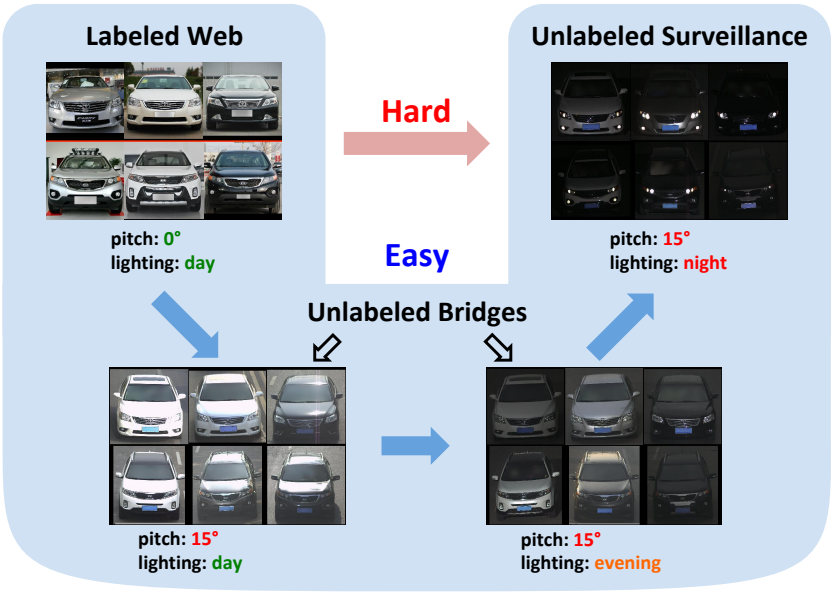



**Figure 5.1**: Unsupervised domain adaptation is challenging when the target domain is significantly different from the source domain due to many convoluted factors of variation. We introduce *bridging domains* composed of unlabeled images with some common factors to the source (e.g., lighting) and the target domain (e.g., viewpoint, image resolution).

In this paper, we aim to solve unsupervised domain adaptation challenges when domain discrepancy is large due to variation across the source and the target domains. Figure 5.1 provides an illustrative example of adapting from labeled images of cars from the internet to recognize cars for surveillance applications at night. Two dominant factors, the perspective and illumination, make this a difficult adaptation task. As a step towards solving these problems, we introduce *unlabeled domain bridges* whose factors of variation are partially shared with the source domain, while the others are in common with the target domain. As in Figure 5.1, the domain on the bottom left shares a consistent lighting condition (day) with the source, while the viewpoint is similar to that of the target domain. We note that there could be multiple bridging domains, such as the one on the bottom right of Figure 5.1, whose lighting intensity is between that of the first bridging domain and the target domain.

To utilize unlabeled bridging domains, we propose to extend the domain adversarial neural network [GUA+16] using multiple domain discriminators, each of which

accounts for learning and reducing the discrepancy between unlabeled (bridging, target) domains and the mix of all precedent domains. We justify our learning framework by deriving a bound on the target error, that contains the source error and a list of discrepancies between unlabeled domain and the mix of precedent domains, including the source. This bound captures the intuition that judicious choices of bridge domains should not introduce large discrepancies. We hypothesize that the decomposition of a single, large discrepancy into multiple, small ones leads to a series of easier optimization problems, culminating in better alignment of source and target domains. We illustrate this intuition in Figure 5.2 on a variant of the two-moons dataset.

While works on unsupervised discovery of latent domains exist [GLC11, GGS13b, GGS14], it still remains a hard, unsolved problem. Firstly, we focus on the complementary and also unsolved problem of devising adversarial formulations that exploit given bridging domains. We observe that such domain information is often easily available in practice, for example, image meta-data such as timestamps, geotags and calibration parameters suffice to inform about illumination, weather or perspective. Moreover, we exploit different methods on measuring domain discrepancy [GBR+12, GUA+16] or out-of-distribution (OOD) sample detection [HG17] to discover latent domains in an unsupervised manner, i.e., without domain information.

## 5.2 Related Work

**Unsupervised Domain Adaptation.** The proper reduction of discrepancy across domains [PY+10] is a longstanding challenge. Specifically, an appropriate metric is required in order to measure the difference in between domains [BDBCP07]. Recent works use kernel-based methods such as maximum mean discrepancy (MMD) [THDS15] and optimal transport (OT) [CFTR15] to measure the domain difference in

the feature space. Others adopt the idea of adversarial training [GUA+16, SLZ+17] which is inspired by the generative adversarial network (GAN) [GPAM+14]. This training procedure allows the feature representations to be indistinguishable between the source and target domain, aligning the two. One example of using adversarial training on UDA problems is the domain adversarial neural network (DANN) [GUA+16]. It trains a discriminator that distinguishes domains, while also learning a feature extractor to fool the discriminator by providing domain-invariant feature.

**Multiple Domains.** In [ZZW+18], models are proposed for multiple-source UDA problems based on a domain adversarial learning. While the intuition is to utilize extra source domains that are available, the adaptation process is in practice favored toward the source domain that is closely related to the target domain [MMR09]. Our method shares the similar high-level idea, in which relevant domains should guide the adaptation. In contrast, *unlabeled* bridging domains that share factors of variation with both source and target domains are utilized to guide the two domains, aligned with the bridging domain. Similar to our proposed approach, the benefit of having intermediate domains to guide transfer learning is shown in [TSZY15], but in the context of semi-supervised label propagation, requiring labeled data from the target domains.

## 5.3 Method

Our proposed domain adaptation framework is built atop DANN, utilizing *unlabeled bridging domains* to enhance the adaptation performance when the source and target domains are significantly different due to factors of variations.

**Notation.** Denote $\mathcal{D}_S$ and $\mathcal{D}_T$ as the source and target domains, respectively, from which data $x$ are drawn. Output label $y \sim \mathcal{Y}$ has $N$ categories. The model contains: 1) a feature extractor $f : \mathcal{D} \rightarrow \mathbb{R}^K$, with parameter $\theta_f$, that maps $x$ into a feature

vector $f(x)$; 2) the domain discriminator $d:\mathbb{R}^K \to (0,1)$, with parameter $\theta_d$, that tells whether $f(x)$ is from $\mathcal{D}_S$ or $\mathcal{D}_T$; and 3) the classifier $C:\mathbb{R}^K \to \mathcal{Y}$, with parameter $\theta_C$, that gives a predicted label $\hat{y} = C(f(x))$.

### 5.3.1 Domain Adversarial Neural Network

The domain adversarial neural network transfers a classifier learned from the labeled source domain to the unlabeled target domain by learning domain-invariant features. It is realized by first learning the domain-related information and leveraging it with features extracted from the input. DANN uses a domain discriminator $d$ to control the amount of domain-related information in the extracted feature. The discriminator is updated by maximizing the following:

$$\mathcal{L}_d = \mathbb{E}_{x\sim\mathcal{D}_S} \log d(f(x)) + \mathbb{E}_{x\sim\mathcal{D}_T} \log(1 - d(f(x))). \tag{5.1}$$

In comparison, the feature extractor $f$ wants to confuse the discriminator $d$ to remove any domain-specific information. Moreover, to make sure the extracted feature is task-related, $f$ is trained to generate features that can be correctly classified by the classifier $C$ trained by minimizing the following:

$$\mathcal{L}_C = \mathbb{E}_{(x,y)\sim\mathcal{D}_S\times\mathcal{Y}}[-y \log C(f(x))], \tag{5.2}$$

and a learning objective for feature extractor is as follows:

$$\min_{\theta_f,\theta_C} \mathcal{L}_C + \lambda\mathcal{L}_d. \tag{5.3}$$

While [GUA+16] introduces a gradient reversal layer to jointly train all parameters, we do alternating update of GANs [GPAM+14] between $\theta_d$ and $\{\theta_f, \theta_C\}$ in our implementation.

### 5.3.2 Challenge in Domain Adversarial Learning

While deep domain adaptation algorithms are realized in different forms [THDS15, THZ+14, GUA+16, SLZ+17, BTS+16, STS+20], their theoretical motivation largely derives from the seminal work of [BDBCP07]. In short, a theorem from that work states that the target domain task error $\epsilon_T$ is bounded by the source error $\hat{\epsilon}_S$ and the domain discrepancy:

$$\epsilon_T(h) \leq \hat{\epsilon}_S(h) + d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T), \tag{5.4}$$

where $h \in \mathcal{H}$ is a hypothesis and $d_{\mathcal{H}\Delta\mathcal{H}}$ is written as:

$$\sup_{h,h' \in \mathcal{H}} |P_{\mathcal{D}_S}(h(x) \neq h'(x)) - P_{\mathcal{D}_T}(h(x) \neq h'(x))|.$$

Adversarial loss can be used to minimize the domain discrepancy to obtain a tighter bound. While it provides flexibility on the types of discrepancy, it is challenging to learn the right transformation from the source domain to the target domain when the two are far apart.

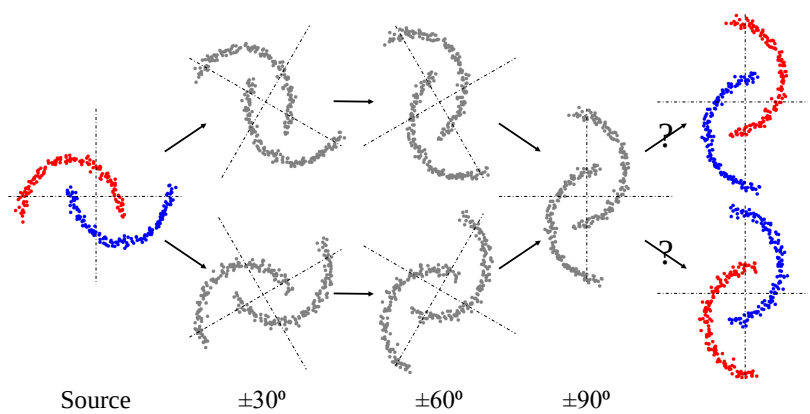



**Figure 5.2**: Translated two moons. The inter-twinning moons (left) are considered as the source domain. Two moons are rotated and translated to the right to generate the target domain (right in gray).

As motivation, consider a variant of the two-moons dataset, whose data points are translated to the right by the amount proportional to the rotation angle, as in Figure 5.2. The source domain is centered at the origin, while the target domain is

moved to the right after being rotated by 90°, and given without labels. Adapting from source to target directly is difficult due to a significant change. Moreover, there are many ways to generate the same unlabeled target data points (e.g., rotate counterclockwise instead of clockwise, as in the bottom of Figure 5.2). In such a case, knowing what happens in the middle of the entire transformation process from source to target domains is critical, as these data points in the middle, even if they are unlabeled, can guide learning algorithms to easily disentangle transformation factors (e.g., clockwise rotation and translation to the right) from task-relevant factors.

### 5.3.3 Adaptation with Bridging Domain

We introduce additional sets of unlabeled examples, which we call bridging domains, that reside in the transformation pathway from labeled source to unlabeled target domains.

**DANN with a Single Bridging Domain.** Besides $\mathcal{D}_S$ and $\mathcal{D}_T$, we denote $\mathcal{D}_B$ as a bridging domain. Our framework is composed of feature extractor $f(x)$ from an input $x \in \mathcal{D}_S \cup \mathcal{D}_B \cup \mathcal{D}_T$ and classifier $C(f(x))$ trained using classification loss in (5.2). Unlike DANN, which directly aligns $\mathcal{D}_S$ and $\mathcal{D}_T$, we decompose the adaptation into two steps. First, $\mathcal{D}_S$ and $\mathcal{D}_B$ are aligned. This is an easier task than direct adaptation as in DANN, since there are less discriminating factors between $\mathcal{D}_S$ and $\mathcal{D}_B$. Second, we adapt $\mathcal{D}_T$ to the union of $\mathcal{D}_S$ and $\mathcal{D}_B$. Similarly, the task is easier since it needs to discover remaining factors between $\mathcal{D}_T$ and $\mathcal{D}_S$ or $\mathcal{D}_B$, as some factors are already found from the previous step. To accommodate the two adaptation steps, we use two binary domain discriminators, $d_1$ for learning discrepancy between $\mathcal{D}_S$ and $\mathcal{D}_B$, and $d_2$ between $\mathcal{D}_S \cup \mathcal{D}_B$ and $\mathcal{D}_T$. Finally, this is realized with the following objectives:

$$\mathcal{L}_{d_1} = \mathbb{E}_{\mathcal{D}_S} \log d_1(f) + \mathbb{E}_{\mathcal{D}_B} \log(1 - d_1(f)), \tag{5.5}$$

$$\mathcal{L}_{d_2} = \mathbb{E}_{\mathcal{D}_S \cup \mathcal{D}_B} \log d_2(f) + \mathbb{E}_{\mathcal{D}_T} \log(1 - d_2(f)). \tag{5.6}$$

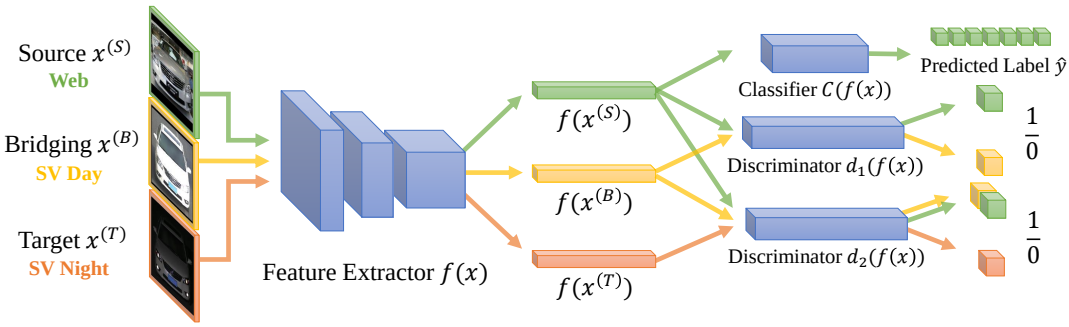



**Figure 5.3**: The learning framework with labeled source, unlabeled target, and unlabeled bridging domains for our extension of DANN using multiple discriminators. The model is composed of shared feature extractor $f$, classifier $C$, which is trained using labeled source examples, and two domain discriminators $d_1$ and $d_2$.

Both $\mathcal{L}_{d_1}$ and $\mathcal{L}_{d_2}$ are minimized to update their respective model parameters $\theta_{d_1}$ and $\theta_{d_2}$. We update the classifier using (5.2) and the feature extractor to confuse discriminators as follows:

$$\min_{\theta_f, \theta_C} \mathcal{L}_C + \lambda_1 \mathcal{L}_{d_1} + \lambda_2 \mathcal{L}_{d_2}, \tag{5.7}$$

with two hyperparameters $\lambda_1$ and $\lambda_2$ to adjust the strengths of adversarial loss. We alternate updates between $d_1, d_2$ and $f, C$. The proposed framework is visualized in Figure 5.3.

**Theoretical Insights.** To provide insights on how our learning objectives are constructed, we derive a bound on the target error while considering the unlabeled bridging domain:

$$\epsilon_T(h) \leq \epsilon_T(h_T^*) + \tfrac{1}{2}\epsilon_B(h_B^*) + 2\gamma_\alpha + \eta + d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_\alpha, \mathcal{D}_T) + \tfrac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_B), \tag{5.8}$$

where $h^* = \arg\min_{h\in\mathcal{H}} \epsilon(h)$, $\mathcal{D}_\alpha = \mathcal{D}_S \cup \mathcal{D}_B$, and

$$\gamma_\alpha = \min_{h\in\mathcal{H}} \{\epsilon_T(h) + \textstyle\sum_{j=1}^{N} \alpha_j \epsilon_j(h)\} \text{ with } \alpha = (0.5, 0.5).$$

Note that $\epsilon_T(h_T^*) + \frac{1}{2}\epsilon_B(h_B^*) + 2\gamma_\alpha \approx \hat{\epsilon}_S(h)$, making (5.8) similar to (5.4). The derivation is provided in the Supplementary Material.

The implications of (5.8) are two-fold: Firstly, to keep the bound tight, we need to assure that both domain discrepancies are small. This motivates the design of our proposed adversarial learning framework discussed earlier. More importantly, we argue that the individual components of *decomposed discrepancies are much easier to optimize* than the one in (5.4) when the bridging domain is chosen properly.

**Unsupervised Bridging Domain Discovery.** While there are many real-world problems where the bridging domains naturally arise (*e.g.*, the illumination condition of the surveillance images, which can be obtained from the mean pixel values), it is not always available. In such cases, one may resort to the unsupervised discovery of latent domains [GLC11, GGS13b, TSZY15].

To find out whether an unlabeled image of the target domain belongs to the bridging domain, one may measure the closeness of individual target examples to the source domain. For example, we propose to pretrain a standard DANN and exploit the discriminator score $d_{\text{pre}}(f_{\text{pre}}(x)), x \in \mathcal{D}_T$ to quantify the closeness. Since the discriminator converges at equilibrium of source and target distributions [GPAM+14], this requires an early stopping in practice [SLZ+17].

Alternatively, we can use off-the-shelf algorithms to compute the distance between individual target examples to the source domain. Given a feature extractor $f_{\text{pre}}$ trained on the source examples, one may compute the MMD between $f_{\text{pre}}(x), x \in \mathcal{D}_T$ and $\{f_{\text{pre}}(x)\}_{x \in \mathcal{D}_S}$. In addition, out-of-distribution (OOD) sample detection methods [HG17] are good candidates as they provide the score quantifying how likely an example belongs to the source domain.

**DANN with Multiple Bridging Domains.** Our framework can be extended to the case for which multiple unlabeled bridging domains exist, which is desirable to span larger discrepancies between source and target domains. To formalize, we denote $\mathcal{D}_0 = \mathcal{D}_S, \mathcal{D}_{M+1} = \mathcal{D}_T$ as source and target domains, and $\mathcal{D}_m$, $m = 1, ..., M$ as

unlabeled bridging domains with $\mathcal{D}_m$ closer to source than $\mathcal{D}_{m+1}$. We introduce $M+1$ domain discriminators $d_1, ..., d_{M+1}$, each of which is trained by maximizing the following objective:

$$\mathcal{L}_{d_m} = \mathbb{E}_{\bigcup_{i=0}^{m-1} \mathcal{D}_i} \log d_m(f) + \mathbb{E}_{\mathcal{D}_m} \log(1 - d_m(f)), \tag{5.9}$$

and the learning objective for $f$ and $C$ is given as follows:

$$\min_{\theta_f, \theta_C} \mathcal{L}_C + \textstyle\sum_{m=1}^{M+1} \lambda_m \mathcal{L}_{d_m}. \tag{5.10}$$

## 5.4 Experiments

We evaluate our methods mainly on three adaptation tasks: digit classification, object recognition, and semantic scene segmentation. For the recognition task, we use the Comprehensive Cars (CompCars) [YLCLT15] dataset to recognize car models in the surveillance domain at night using labeled images from the web domain. For the scene segmentation task, synthetic images of the GTA5 dataset [RVRK16] are given as the source domain and the task is to perform adaptation on Foggy Cityscapes [SDVG18]. We also provide results on the two-moons toy dataset as described previously and the digit classification task.

### 5.4.1 Toy Experiment with Two Moons

Created for binary classification problem, the inter-twinning moons 2D dataset suits our model if we consider different rotated versions of the standard two entangled moons as different domains. In this experiment, we consider a hard adaptation from the original data to the ones that are rotated 90° (clockwise or counter-clockwise), while intermediate rotation such as 30° and 60° can be considered as bridging domains. Moreover, as discussed in Section 5.3.2, the domains do not share the same centers and are proportionally translated according to the rotated angle. We follow

the same network architecture as in [GUA+16], with one hidden layer of 15 neurons followed by sigmoid non-linearity. The performance is summarized in Table 5.1.

**Table 5.1**: Average classification accuracy on test set of each domain. Results for the baseline and different bridging domain combinations are included.

| **Model** | 0° | 30° | 60° | 90° |
|---|---|---|---|---|
| 0→90 | 80.88±1.71 | - | - | 56.98±4.47 |
| 0→30→90 | 87.23±3.64 | 95.66±4.18 | - | 60.98±7.41 |
| 0→60→90 | 79.19±1.21 | - | 89.66±3.61 | 80.67±9.47 |
| 0→30→60→90 | 78.75±1.56 | 82.33±8.71 | 87.33±3.83 | **86.97**±2.17 |

One observation is that when 90° is involved as a target domain, the source domain accuracy is sacrificed a lot, which may be because of the limited network capacity. While the adaptation achieves only 56.90%, which is almost a random guess, with source-to-target model (0→90), the proposed method clearly demonstrates its effectiveness, achieving 86.97% on the target domain.

### 5.4.2 Digit Classification

Different digit datasets are considered as separated domains. MNIST [LBBH98] provides a large amount of hand written digit images in gray scale. SVHN [NWC+11] contains colored digit images of house numbers from street view. MNIST-M [GUA+16] is enriched from MNIST using randomly selected colored image patches in BSD500 [AMFM11] as background. We consider adaptation from labeled MNIST to unlabeled SVHN, while using MNIST-M as an unlabeled bridging domain. Given the differences between MNIST and SVHN, MNIST-M seems appropriate bridging domain (similar appearance of foreground digits to MNIST but color statistics to SVHN).

We compare our model with the baseline model, i.e., a standard DANN from source to target without bridging domain. A DANN model that adapts to the mixture of bridge and target domains as a single target is included for comparison. We

**Table 5.2**: Accuracy on MNIST-M and SVHN test sets averaged over 10 runs. We report the performance of the standard DANN, the DANN model using mixture of unlabeled domains as a single target (MNIST→MNIST-M+SVHN), and our proposed model.

| **Model** | MNIST-M | SVHN |
|---|---|---|
| MNIST→SVHN | - | 71.02 |
| MNIST→MNIST-M+SVHN | 96.27 | 78.07 |
| MNIST→MNIST-M→SVHN | 97.07 | **81.28** |

present results in Table 5.2. When the bridging domain is involved, the average accuracy on SVHN (target) significantly improves upon the baseline model. Moreover, our proposed model achieves higher performance than the model with mixture of unlabeled domains, demonstrating benefits from the bridging domain.

### 5.4.3 Recognizing Cars in SV Domain at Night

**Dataset and Experimental Setting.** Two sets of images are provided in the CompCars dataset: 1) the web-nature images are collected from car forums, public websites and search engines, and 2) the surveillance-nature images are collected from surveillance cameras. The dataset is composed of $52,083$ web images across 431 car models and $44,481$ SV images across 181 car models, with these categories of the SV set being inclusive of 431 categories from the web set. We consider a set of adaptation problems from labeled web to unlabeled SV images. This is challenging as SV images have different perspective and illumination variations from web images.

We use an illumination condition as a metric for adaptation difficulty and partition the SV set into SV1–5 based on the illumination condition of each image.[1] SV1 contains the brightest images, whereas SV5 contains the darkest ones. We visualize samples from SV1–5 in Figure 5.4, and confirm the domain discrepancy between web

[1]We compute the mean pixel-intensity and sort/threshold images to construct SV1–5 with roughly the same sizes. In practice, the illumination condition may be obtained from metadata, such as recorded time.

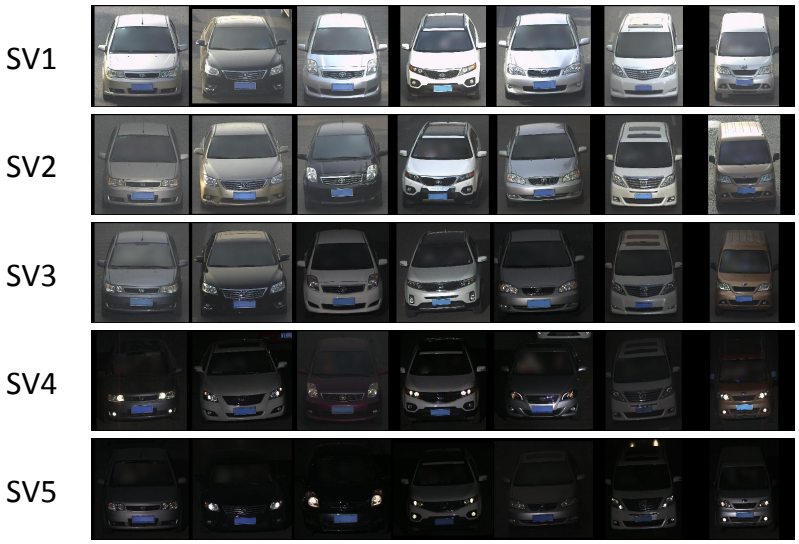



**Figure 5.4**: Sample images of CompCars surveillance (SV) domain from light (SV1) to dark (SV5) illumination conditions.

**Figure 5.5**: t-SNE plots of CompCars web and SV domains from light (SV1) to dark (SV5) illumination conditions using baseline features.

and SV1–5 domains through t-SNE plots in Figure 5.5.

We present two experimental protocols. First, we evaluate on an adaptation task from web to SV night (SV4–5) using SV day (SV1–3) as one domain bridge. We demonstrate the difficulty of adaptation when two domains are far from each other, and show the importance of bridging domain and the effectiveness our adaptation method. Second, we adapt to extreme SV domain (SV5) using different combinations of one or multiple bridging domains (SV1–4) and characterize the properties of an effective bridging domain.

**Evaluation with a Single Bridging Domain.** We demonstrate the difficulty of adaptation when domains are far apart and show that the performance of adversarial DA can be enhanced using bridging domains. In particular, night images (SV4–5) are

considered as unlabeled target domain and day images (SV1–3) as unlabeled bridging domain. We compare the following models in Table 5.3: baseline model trained on labeled web images, DANN from source to target (Web→SV4–5), from source to mixture of bridge and target (Web→SV1–5), and the proposed model from source to bridge to target (Web→SV1–3→SV4–5).

**Table 5.3**: Accuracy and standard error over 5 runs on SV test sets for models without (Web→SV4–5) and with (Web→SV4→5, Web→SV1–3→4–5) bridging domain. Baselines include a model using mixture of bridge and target domains as a single target domain (Web→SV1–5).

| **Model** | SV1–3 | SV4–5 |
|---|---|---|
| Web (source only) | 72.67 | 19.87 |
| Web→SV4–5 | 68.90±1.28 | 49.83±0.70 |
| Web→SV4→5 | **74.03**±0.71 | **61.37**±0.30 |
| Web→SV1–5 | **83.29**±0.14 | 77.84±0.34 |
| Web→SV1–3→4–5 | **82.83**±0.40 | **78.78**±0.33 |

While the DANN adapted to the target domain (SV4–5) improves upon the baseline model, the performance is still far from adequate when compared to the performance of day images. By introducing unlabeled bridging domain, we observe significant improvement in accuracy on the target domain, achieving 77.84% using standard DANN adapted to the mixture of bridging and target domains and 78.78% using our proposed method.

We further conduct an experiment using SV4 as a bridge domain and SV5 as a target domain and compare with the naively trained model (Web→SV4–5). As in Table 5.3, the proposed model (Web→SV4→5) outperforms the DANN by a large margin (49.83% to 61.37% on SV4–5 test set). This is because it is difficult to determine the adaptation curriculum as both domains are distant from the source domain (see Figure 5.5), which is different from the previous experiment where there are sufficient amount of day images that are fairly close to the source domain, for

discriminator to figure out the curriculum.

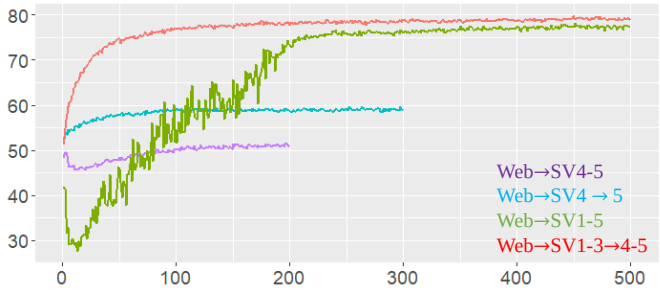



**Figure 5.6**: Validation accuracy on SV4–5 (night)

To better understand the advantage of our proposed training scheme, we monitor the validation accuracy on SV4–5 of the standard (Web→SV1–5) and the proposed (Web→SV1–3→4–5) models and plot curves in Figure 5.6. Interestingly, we observe a large fluctuation in the performance of night images using the standard DANN. In contrast, our method allows stable performance earlier in the training, which implies that knowing the curriculum [BLCW09] (*i.e.*, adaptation difficulty) is important. Our method with multiple discriminators effectively utilizes such information.

**Table 5.4**: Accuracy and standard error over 5 runs on SV5 test set for models with different bridging domain configurations.

| **Model** | SV5 |
|---|---|
| Web→SV5 | 37.83±0.51 |
| Web→SV4→5 | 58.40±0.60 |
| Web→SV1→5 | 69.69±0.99 |
| Web→SV3→4→5 | 74.01±0.52 |
| Web→SV2→3→4→5 | 75.15±0.18 |
| Web→SV1→2→3→4→5 | 75.47±0.20 |

**Which is a Good Bridging Domain?** We perform an ablation study to characterize the properties of a good bridging domain. Specifically, we would like to answer which is a more useful bridging domain: the one closer to the source domain or the one closer to the target domain. To this end, we compare two models,

namely, Web→SV1→5 and Web→SV4→5. Note that SV4 is more similar to the target domain (SV5) in terms of visual attributes than SV1.

The results are summarized in Table 5.4. We observe much higher accuracy on the target domain (SV5) for the model using SV1 as a bridging domain (69.69%) than the one using SV4 (58.40%). We believe that the optimization of the adversarial loss for the second model ($d_{\mathcal{H}\Delta\mathcal{H}}(\mathrm{W}, \mathrm{SV4}) + d_{\mathcal{H}\Delta\mathcal{H}}(\mathrm{W{+}SV4}, \mathrm{SV5})$) is more difficult than that of the first model ($d_{\mathcal{H}\Delta\mathcal{H}}(\mathrm{W}, \mathrm{SV1}) + d_{\mathcal{H}\Delta\mathcal{H}}(\mathrm{W{+}SV1}, \mathrm{SV5})$) as SV4 is farther from the web domain than SV1. This implies that a good bridging domain decomposes the domain discrepancy between the source and target domains, so that the decomposed discrepancy losses is easily optimized.

**Evaluation with Multiple Bridging Domains.** Our theoretical motivation suggests that, if we have many bridging domains whose generalization error between any two neighboring domains is small, we can also reduce the generalization error between source and target.

We test our hypothesis through experiments that adapt to SV5 with different bridging domain configurations. Specifically, bridging domains are included one by one from SV4 to SV1, and finally reach adaptation with four bridging domains. As in shown Table 5.4, DANN fails at adaptation without domain bridge (Web→SV5). While including SV4 as the target domain raises adaptation difficulty, using it as a bridging domain (Web→SV4→5) greatly improves the performance on the SV5 test set. Including SV3 as an additional bridging domain (Web→SV3→4→5) shows additional improvement, confirming our hypothesis. While adding SV2 and SV1 as bridging domains leads to an extra improvement, the margin is not as large as including SV3 and SV4. The reason is that SV3 is already close to the web domain as SV1 or SV2 (see Figure 5.4 and 5.5), and there is little benefit of introducing an extra bridge.

**Table 5.5**: Accuracy and standard error over 5 runs on SV4–5 test set for models with different bridging domain configurations. Split $m$-way means that we evenly split the unlabeled data into $m$ domains. Each column indicates whether the split is performed based on OOD [HG17], MMD, or discriminator scores. The number next to each method is the AUC between the predicted score and the ground-truth day/night labels.

| Split | OOD (0.69) | MMD (0.79) | $d$-Score (0.85) |
|---|---|---|---|
| 2-way | 76.43±0.28 | 78.32±0.27 | 78.62±0.39 |
| 3-way | 76.14±0.34 | 78.47±0.31 | 78.29±0.28 |
| 4-way | 76.67±0.32 | 78.68±0.39 | 77.93±0.43 |
| 5-way | 76.54±0.29 | 78.91±0.33 | 79.13±0.32 |

**Evaluation with Unsupervised Bridge Discovery.** We perform several unsupervised approaches for bridging domain discovery, including discriminator scores of the DANN, MMD, and OOD sample detection [HG17]. These approaches provide scores indicating the closeness to the source domain distribution for each target example, based on which we can split the target domain into multiple bridging domains (details in the Supplementary Material).

To evaluate the performance of our methods on unsupervised bridging domain discovery, we first compute the AUC between the predicted score and the ground-truth day/night labels and report results in Table 5.5. While we observe the highest AUC using discriminator of the DANN (0.85), it requires early stopping based on the AUC. In comparison, MMD (0.79) and OOD (0.69) show slightly lower AUC, but are preferred as additional side information is not required. Overall, we observe that using $d$-Score or MMD, the performance on SV4–5 test set is improved compared to not using any domain bridges (Table 5.3 Web→SV1–5).

### 5.4.4 Foggy Scene Segmentation

**Dataset and Experimental Setting.** We use the GTA5 dataset [RVRK16], a synthetic dataset of street-view images, containing 24,966 images of size 1914×1052,

as labeled source domain. Unlike previous works [HWYD16, THS+18], we adapt to Foggy Cityscapes [SDVG18], a derivative from the real scene images of Cityscapes [COR+16] with a fog simulation, and use Cityscapes as well as Foggy Cityscapes with lighter foggy levels (0.01) as unlabeled bridging domains.

The task is to categorize each pixel into one of 19 semantic categories on the test set images of Foggy Cityscapes with 0.02 foggy level. We consider several models for comparison, such as the traditional DANN (GTA5→$F_{0.02}$), with one bridging domain (GTA5→City→$F_{0.02}$), or with two of them (GTA5→City→$F_{0.01}$→$F_{0.02}$). One consideration is that we partition 2975 training images of Cityscapes equally for each domain to prevent the case where the algorithm finds an exact correspondence between images from different unlabeled domains.

**Evaluation on Semantic Segmentation.** We utilize the adaptation method by [THS+18] as our base model, which reduces the domain discrepancy at structured output spaces. The same discriminator architecture is used for multiple adversarial losses in our framework.

**Table 5.6**: mIoU and standard error over 5 runs on Foggy Cityscapes 0.02 ($F_{0.02}$) test set. We partition unlabeled training data into 2 for models in rows 2–4, resulting in 1487 images per domain, while for models in rows 5–7, we partition them into 3, resulting in 991 images per domain.

| **Model** | # images | mIoU on $F_{0.02}$ |
|---|---|---|
| GTA5 (source only) | – | 27.5 |
| GTA5→$F_{0.02}$ [THS+18] | | 33.08±0.32 |
| GTA5→City+$F_{0.02}$ | 1487 | 32.62±0.58 |
| GTA5→City→$F_{0.02}$ | | **34.82**±0.39 |
| GTA5→$F_{0.02}$ [THS+18] | | 33.16±0.35 |
| GTA5→City→$F_{0.02}$ | 991 | 34.13±0.78 |
| GTA5→City→$F_{0.01}$→$F_{0.02}$ | | **35.31**±0.06 |

We first conduct experiments with one bridging domain to adapt to Foggy Cityscapes

0.02 ($F_{0.02}$). We construct two partitions of unlabeled data for Cityscapes and $F_{0.02}$. Mean intersection-over-union (IoU) averaged over 5 runs, using different partition for each run, is reported in rows 2–4 of Table 5.6. While significant improvement in mIoU is observed by directly adapting to the target domain (GTA5→$F_{0.02}$), our framework using a bridging domain further enhances the performance on the final target domain from 33.08 to 34.82. We also conduct a baseline model by merging Cityscapes with the target domain (GTA5→City+$F_{0.02}$), but the performance is not good, indicating that naively merging two domains with different properties may be suboptimal for adversarial adaptation.

We then experiment with two bridging domains by introducing Foggy Cityscapes 0.01 ($F_{0.01}$) as a bridge between Cityscapes and $F_{0.02}$. The setting is similar, but we use $\frac{1}{3}$ of entire images for each unlabeled domain. Table 5.6 rows 5–7 validate our hypothesis that additional bridging domains are beneficial, improving mIoU from 34.13 to 35.31. While using the same number of overall unlabeled images during the training, we also observe benefit of using two bridging domains ($7^{\text{th}}$ row) than one ($4^{\text{th}}$ row).

## 5.5 Conclusions

This paper aims to simplify adaptation problems with extreme domain variations, using unlabeled bridging domains. A novel framework based on DANN is developed by introducing additional discriminators to account for decomposed many, but smaller discrepancies of the source-to-target domain discrepancy. Several adaptation tasks in computer vision are considered, demonstrating the effectiveness of our framework with bridging domains.

## 5.6 Supplementary Material

### 5.6.1 Proof of Eq. 4.8

With source and bridging domains, $h^* = \arg\min_{h\in\mathcal{H}} \epsilon(h)$, $h_1 = \arg\min_{h\in\mathcal{H}} \hat{\epsilon}_S(h)$, an empirical minimizer of source error, and weight vector $\alpha = (0.5, 0.5)$, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the target error can be bounded as follows:

$$\begin{aligned}\epsilon_T(h_1) \leq \epsilon_T(h_T^*) + \tfrac{1}{2}\epsilon_B(h_B^*) + 2\gamma + 2\eta \\ + d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_\alpha, \mathcal{D}_T) + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_B)\end{aligned} \tag{5.11}$$

*where $h \in \mathcal{H}$ is a hypothesis and*

$$d_{\mathcal{H}\Delta\mathcal{H}} = \sup_{h,h'\in\mathcal{H}} |P_{\mathcal{D}_S}(h(x) \neq h'(x)) - P_{\mathcal{D}_T}(h(x) \neq h'(x))| \tag{5.12}$$

$$\gamma = \min_{h\in\mathcal{H}} \epsilon_T(h) + \epsilon_S(h) + \epsilon_B(h) \tag{5.13}$$

$$\eta = 2\sqrt{\left(\frac{2d\log(2m+1) + \log(\frac{4}{\delta})}{m}\right)} \tag{5.14}$$

*Proof.* For the presentation clarity, we use $\epsilon_\alpha = \frac{1}{2}\epsilon_S + \frac{1}{2}\epsilon_B$ interchangeably. Let $h_1, h_2, h_3 \in \mathcal{H}$, which will be defined later. We begin by bounding the target error $\epsilon_T$ by the mixture error $\epsilon_\alpha$ and the divergence as follows:

$$\epsilon_T(h_1) \leq \epsilon_T(h_2) + \epsilon_\alpha(h_1, h_2) + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_\alpha, D_T) \tag{5.15}$$

The second term of RHS in (5.15) is further bounded as follows:

$$\epsilon_\alpha(h_1, h_2) = \tfrac{1}{2}\epsilon_S(h_1, h_2) + \tfrac{1}{2}\epsilon_B(h_1, h_2) \tag{5.16}$$

$$\leq \tfrac{1}{2}\big[\epsilon_S(h_1) + \epsilon_S(h_2)\big] + \tfrac{1}{2}\big[\epsilon_B(h_1) + \epsilon_B(h_2)\big] \tag{5.17}$$

and $\epsilon_B(h_1)$ is bounded as follows:

$$\epsilon_B(h_1) \leq \epsilon_B(h_3) + \epsilon_S(h_1, h_3) + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) \tag{5.18}$$

$$\leq \epsilon_B(h_3) + \epsilon_S(h_1) + \epsilon_S(h_3) + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) \tag{5.19}$$

Plugging (5.17) and (5.19) into (5.15), we get the following:

$$\epsilon_T(h_1) \leq \epsilon_T(h_2) + \Big[\tfrac{1}{2}\big[\epsilon_S(h_1) + \epsilon_S(h_2)\big] + \tfrac{1}{2}\big[\epsilon_B(h_1) + \epsilon_B(h_2)\big]\Big] + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_\alpha, D_T) \tag{5.20}$$

$$= \big[\epsilon_T(h_2) + \tfrac{1}{2}\epsilon_S(h_2) + \tfrac{1}{2}\epsilon_B(h_2)\big] + \tfrac{1}{2}\big[\epsilon_B(h_3) + \epsilon_S(h_3)\big] + \epsilon_S(h_1) + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_\alpha, D_T) + \tfrac{1}{4}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) \tag{5.21}$$

Assuming $h_2 = \arg\min_h\{\epsilon_T(h) + \frac{1}{2}\epsilon_S(h) + \frac{1}{2}\epsilon_B(h)\}$ and $h_3 = \arg\min_h\{\epsilon_B(h) + \epsilon_S(h)\}$, the RHS of (5.21) is written as follows:

$$\epsilon_T(h_1) \leq \gamma_1 + \tfrac{1}{2}\gamma_2 + \epsilon_S(h_1) + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_\alpha, D_T) + \tfrac{1}{4}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) \tag{5.22}$$

where $\gamma_1 = \min_h\{\epsilon_T(h) + \frac{1}{2}\epsilon_S(h) + \frac{1}{2}\epsilon_B(h)\}$ and $\gamma_2 = \min_h\{\epsilon_B(h) + \epsilon_S(h)\}$. We further assume that $h_1 = \arg\min_{\hat{\epsilon}_S(h)}$, an empirical minimizer of source error.

Now we are left with bounding the source error $\epsilon_S(h_1)$ by the empirical source error $\hat{\epsilon}_S(h_1)$, theoretical minimum errors of the target $\epsilon_T(h_T^*)$ and the bridging $\epsilon_B(h_B^*)$ domains. This is done by using Lemma 6 in [BDBC+10] as follows:

$$\epsilon_S(h_1) \leq \hat{\epsilon}_S(h_1) + \eta \tag{5.23}$$

$$\leq \tfrac{1}{2}\hat{\epsilon}_S(h_T^*) + \tfrac{1}{2}\hat{\epsilon}_S(h_B^*) + \eta \tag{5.24}$$

$$\leq \tfrac{1}{2}\epsilon_S(h_T^*) + \tfrac{1}{2}\epsilon_S(h_B^*) + 2\eta \tag{5.25}$$

$$\leq \tfrac{1}{2}\epsilon_S(h_T^*) + \tfrac{1}{2}\Big[\epsilon_B(h_B^*) + \underbrace{\epsilon_S(h_3) + \epsilon_B(h_3)}_{=\gamma_2} + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B)\Big] + 2\eta \tag{5.26}$$

$$\leq \tfrac{1}{2}\epsilon_S(h_T^*) + \tfrac{1}{2}\epsilon_B(h_B^*) + \tfrac{1}{2}\gamma_2 + \tfrac{1}{4}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) + 2\eta \tag{5.27}$$

$$\leq \underbrace{\tfrac{1}{2}\epsilon_S(h_T^*) + \tfrac{1}{2}\epsilon_B(h_T^*)}_{=\epsilon_\alpha(h_T^*)} + \tfrac{1}{2}\epsilon_B(h_B^*) + \tfrac{1}{2}\gamma_2 + \tfrac{1}{4}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) + 2\eta \tag{5.28}$$

$$\leq \Big[\epsilon_T(h_T^*) + \underbrace{\epsilon_T(h_2) + \epsilon_\alpha(h_2)}_{=\gamma_1} + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_\alpha, D_T)\Big] + \tfrac{1}{2}\epsilon_B(h_B^*) + \tfrac{1}{2}\gamma_2 + \tfrac{1}{4}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) + 2\eta \tag{5.29}$$

$$= \epsilon_T(h_T^*) + \tfrac{1}{2}\epsilon_B(h_B^*) + \gamma_1 + \tfrac{1}{2}\gamma_2 + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_\alpha, D_T) + \tfrac{1}{4}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) + 2\eta \tag{5.30}$$

where the second inequality is due to the fact that $h_1 = \arg\min_h \hat{\epsilon}_S$ and sixth is by adding $\frac{1}{2}\epsilon_B(h_T^*)$ to RHS. $\eta$ and $2\eta$ are introduced in the second and third inequalities using Lemma 6. Finally, plugging in (5.30) into (5.22), we get the following:

$$\epsilon_T(h_1) \leq \epsilon_T(h_T^*) + \tfrac{1}{2}\epsilon_B(h_B^*) + 2\gamma_1 + \gamma_2 + d_{\mathcal{H}\Delta\mathcal{H}}(D_\alpha, D_T) + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) + 2\eta \quad (5.31)$$

$$\leq \epsilon_T(h_T^*) + \tfrac{1}{2}\epsilon_B(h_B^*) + 2\gamma + d_{\mathcal{H}\Delta\mathcal{H}}(D_\alpha, D_T) + \tfrac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_B) + 2\eta \quad (5.32)$$

where the last inequality is given that $\min_h f(h) + \min_h g(h) \leq \min_h\{f(h) + g(h)\}$ for any $f$ and $g$.

□

### 5.6.2 Additional Experiments

Additional experiment results on Web→SV$x$:5, for $x = 1, 2, 3, 4$ are shown in Figure 5.7, e.g., SV3:5 denotes SV3→SV4→SV5.

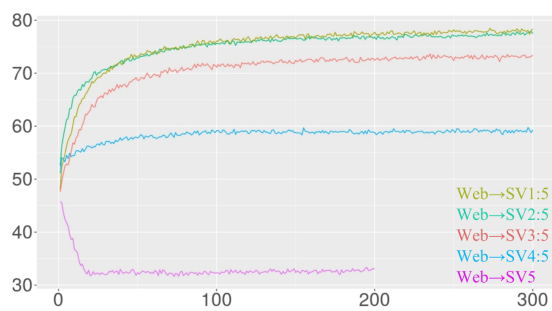



**Figure 5.7**: Validation accuracy over training epochs of our proposed domain adaptation framework with bridging domains. SV5 is used for validation set.

### 5.6.3 Unsupervised Discovery of Bridging Domains

While works on unsupervised discovery of latent domains exist [GLC11, GGS13b, GGS14], the choice of bridging domains remains a hard, unsolved problem. In this section, we present several approaches that we have exploited along this direction. Our initial approach is to quantify the closeness to the source domain of each image in the target domain by using the discriminator score $d_{\text{pre}}(f_{\text{pre}}(x))$ of pretrained

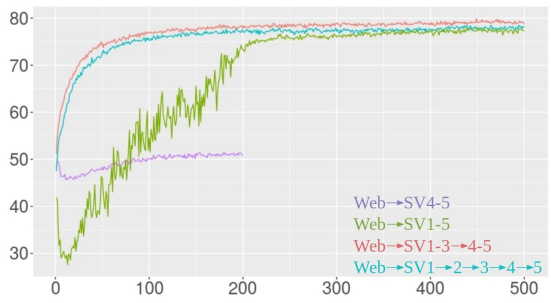



**Figure 5.8**: Performance on SV4–5 based on *supervised* bridging domain discovery using ground truth lighting conditions.

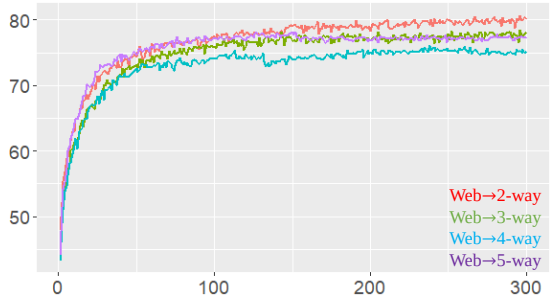



**Figure 5.9**: Performance on SV4–5 based on *unsupervised* bridging domain discovery using discriminator score of pretrained DANN.

DANN model as an indicator. This approach [SLZ+17] intuitively makes sense as discriminator is trained to distinguish source and target domains, and those images from the target domain predicted as source domain are likely to be more similar to those images in the source domain, thus qualified as a bridging domain. Unfortunately, this is not necessarily true since the DANN is trained in an adversarial way and the discriminator at convergence should not be able to distinguish images from source and target domains [GPAM+14]. Specifically, if we split the surveillance dataset into two domains based on the discriminator scores at each training epoch, and compute the AUC using the ground truth of day (SV1–3) and night (SV4–5) labels, we can see in Figure 5.13 that the AUC decreases as the number of training epochs increases. Meanwhile, as shown in Figure 5.10, 5.11 and 5.12, we visualize images from the surveillance domain based on the discriminator score from left-top the highest to right-bottom the lowest. With early stopping at the epoch 10, the discriminator of pretrained DANN model could be more discriminative in separating day and night images than those early stopped at epoch 50 and 150, which are closer to the convergence, thus cannot discriminate the images between the source and the target domains.

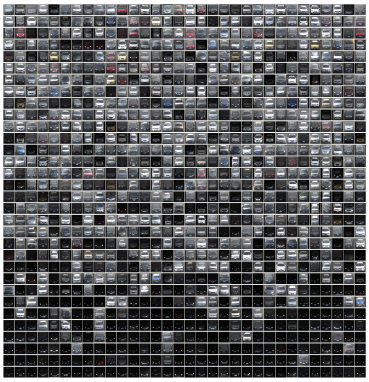


**Figure 5.10**: Epoch 10.

**Figure 5.11**: Epoch 50.

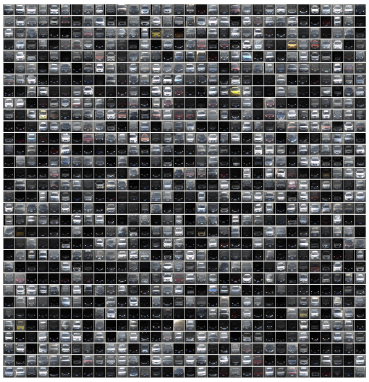


**Figure 5.12**: Epoch 150.

Based on our intuition and the visual inspection, we propose to construct bridging domains based on the discriminator score of the DANN model at epoch 10. By ranking the discriminator score $d_{\text{pre}}(f_{\text{pre}}(x))$ for $x \in \mathcal{D}_T$, we evenly split the unlabeled target data into $m$ sub-domains, denoted as $\mathcal{D}_1, \cdots, \mathcal{D}_m$ for $m = 2, \cdots, 5$; $\mathcal{D}_1$ has the highest discriminator score and $\mathcal{D}_m$ the lowest. We then apply our proposed framework on $\mathcal{D}_1, \cdots, \mathcal{D}_m$ with $\mathcal{D}_m$ as the target and the rest as the bridging domains. Results are shown in Figure 5.9 (also Table 5.5 from the main paper). Note that we use the SV4–5 for validation and testing so that the results are comparable with the reported ones in the main paper. The performance of our framework using unsupervised bridging domain discovery is highly competitive to those using ground truth lighting condition to construct bridging domains. Moreover, our proposed framework with discovered bridging domains demonstrates much more stable training curve (Figure 5.9) comparing to the baseline DANN model (Figure 5.8, Web→SV1–5).

In addition, we evaluate the performance of our proposed adaptation framework with discovered bridging domains using DANN models at epoch 50 and 150. Using Web→2-way (78.62%) as a reference, the results are 76.31% and 67.46% respectively. This confirms our observation in Figure 5.13 that our framework is the most effective

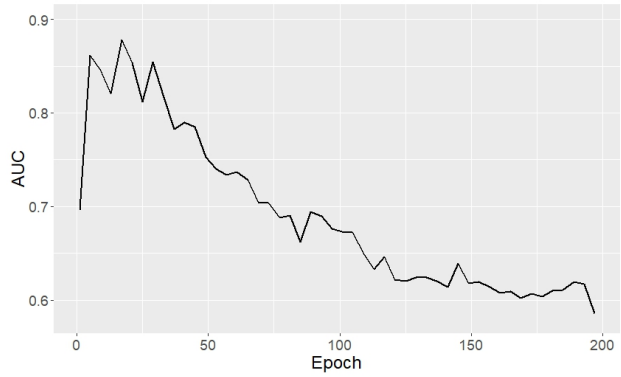



**Figure 5.13**: AUC between predicted closeness to the source domain using discriminator score and the ground-truth day/night labels at each training epoch.

when the bridging domains are retrieved by the discriminator of DANN stopped early.

While using discriminator scores demonstrates the effectiveness in unsupervised bridging domain detection, additional model selection stage (i.e., early stopping) is required to find a reliable discriminator $d_{\text{pre}}$. To avoid this, we can directly use different measure of closeness in the feature space between the source and the target domains. Specifically, we propose two measures, namely, the maximum mean discrepancy (MMD) [GBR+12] and the out-of-distribution (OOD) sample detection score [HG17]. To evaluate these metrics, we first pretrain a classification model on the source domain only, with feature extractor $f_{\text{pre}}$ and classifier $C_{\text{pre}}$. Then, the pretrained extractor is applied to each of the target domain data $f_{\text{pre}}(x), x \in \mathcal{D}_T$. We compute the MMD between a target feature $f_{\text{pre}}(x^{(T)})$ and the entire source domain distribution $\{f_{\text{pre}}(x^{(S)})\}$ as follows:

$$\text{MMD}(f_{\text{pre}}(x^{(T)}), \{f_{\text{pre}}(x^{(S)})\}) = \|\phi(f_{\text{pre}}(x^{(T)})) - \mathbb{E}_{x^{(S)} \sim \mathcal{D}_S}[\phi(f_{\text{pre}}(x^{(S)}))]\|_{\mathcal{H}}, \quad (5.33)$$

where $\phi : \mathcal{D} \to \mathcal{H}$ is the kernel mapping, and $\mathcal{H}$ is the a reproducing kernel Hilbert space (RKHS). By ranking the target domain based on the MMD values (the smaller the MMD, the closer the target feature is to the source domain), we can split target

domain into several sub domains, where the ones that are close to the source domain can be considered as the bridging domains.

Alternatively, we can use the out-of-distribution (OOD) sample detection methods [HG17]. Consider the output of the pretrained classifier for a target sample is $\hat{y}^{(T)} = C_{\text{pre}}(f_{\text{pre}}(x^{(T)}))$, such that $\hat{y}^{(T)} \sim \mathcal{Y}$ has $N$ categories, denoted as $\hat{y}^{(T)} = \{\hat{y}_1^{(T)}, \cdots, \hat{y}_N^{(T)}\}$. Each $\hat{y}_i^{(T)}$ is the probability of $x^{(T)}$ being in category $i$. The OOD sample detection algorithm basically calculates:

$$\text{OOD}(\hat{y}^{(T)}) = \max_i \sigma(\hat{y}_i^{(T)}), \tag{5.34}$$

where $\sigma(\cdot)$ is the softmax function. The lower the value of $\text{OOD}(\hat{y}^{(T)})$ is, the more likely $x^{(T)}$ would be an out-of-distribution sample, and the further it is from the source domain. Similar to the MMD based approach, we can split the target domain based on the OOD score of every target domain sample.

As shown in Table 5.5 from the main paper, the discriminator score based approach achieves the highest AUC of 0.85. Without any requirement of early stopping, the MMD based approach provides an AUC of 0.79, and competitive model performance to the one from the discriminator score. The AUC from the OOD approach is relatively low at 0.69, and the model performance is lower than the other two. Moreover, we observe that the classification accuracy is well correlated with the AUC score, suggesting the importance of more advanced algorithms [LLS18, LLLS18] for measuring the closeness sensibly of the target example to the source domain.

# Chapter 6

# Conclusion

In this dissertation, I focus on applying different representation learning methodologies on various applications, including image generation, text generation, and unsupervised domain adaptation. In most of the works that I have presented, I consider adversarial training as a tool for distribution matching, which, based on different problem setups (*i.e.*, generative models or domain adaptation), helps to align learnt feature representations between real and fake data (or source and target domains). Such a training mechanism provides guidance on merging the two data distributions into one latent space, with shared information aligned.

To address the limitation of the previously proposed variational autoencoder framework, we further develop two new methods with applications in both vision and language domains.

- While the original VAE yields a lower bound on the log likelihood of the observed data, such a learning tends to provide models with a high probability of generating unrealistic synthetic data. To alleviate such a problem, We present the symmetric variational autoencoder (sVAE), a novel framework which can match the joint distribution of data and latent code using the *symmetric* Kullback-Leibler divergence. Such an approach provides a close connection between the VAE framework and the previously developed adversarial training mechanism. The experiment results show the advantages of sVAE, in that it not only overcomes the missing mode problem, but also is very stable to train.

- When applying the original VAE on text generation tasks, one may often en-

counter the so-called posterior collapse problem, in which the decoder ignores the latent inputs and generates sentences with low diversity. We present APo-VAE, a novel model for text generation in hyperbolic space. The increase in latent distribution complexity helps to alleviate the posterior collapse problem by enriching the information flow in the latent distribution learning. Besides, our model can learn latent hierarchies in natural language via the use of wrapped normal for the prior. A primal-dual view of KL divergence is adopted for robust model training, which is realized using adversarial training mechanism. Extensive experiments on language modeling, text style transfer, and dialog response generation demonstrate the superiority of the model.

We also improve upon existing unsupervised domain adaptation models, by using feature representations learnt from contrastively constructed task-oriented classifiers, as well as intermediate adaptation segments.

- Previously proposed domain adversarial neural networks (DANN) does not focus on extracting class-orientated information, and thus, do not perform well on harder UDA problems such as MNIST→SVHN. We have proposed Contrastively Smoothed Class Alignment (CoSCA) for the UDA problem, by explicitly combining intra-class and inter-class domain discrepancy and optimizing class alignment through end-to-end mini-batch training. Experiments on several benchmarks demonstrate that our model can outperform state-of-the-art baselines. Our experimental analysis shows that CoSCA learns more discriminative target domain features, and the introduced MMD feature matching improves the global domain alignment.

- UDA problems could be relatively challenging when large domain gaps exist due many factors of variations. We propose to simplify adaptation problems with

extreme domain variations, using unlabeled bridging domains. A novel framework based on DANN is developed by introducing additional discriminators to account for decomposed many, but smaller discrepancies of the source-to-target domain discrepancy. Several adaptation tasks in computer vision are considered, demonstrating the effectiveness of our framework with bridging domains.

# Bibliography


[AB17] M. Arjovsky and L. Bottou. Toward principled methods for training generative adversarial networks. *ICLR*, 2017.

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[AGL+17] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In *ICML*, 2017.

[AMFM11] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *PAMI*, 2011.

[BAPM15] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *EMNLP*, 2015.

[BCB12] D. Bahdanau, K. Cho, and Y. Bengio. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 2012.

[BCB15] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[BCK+08] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. In *NeurIPS*, 2008.

[BCV13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *PAMI*, 2013.

[BDBC+10] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 2010.

[BDBCP07] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *NeurIPS*, 2007.

[BDP07] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007.

[BGL+94] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In *NeurIPS*, 1994.

[BGS16] Y. Burda, R. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. In *ICLR*, 2016.

[BJ06] D. M. Blei and M. I. Jordan. Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1(1):121–144, 2006.

[BKM17] D.M. Blei, A. Kucukelbir, and J.D. McAuliffe. Variational inference: A review for statisticians. In *arXiv:1601.00670v5*, 2017.

[BLCW09] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, 2009.

[BLP+12] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio. Theano: new features and speed improvements. In *NIPS Workshop*, 2012.

[BSD+17] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *CVPR*, 2017.

[BSM17] David Berthelot, Thomas Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

[BTS+16] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NeurIPS*, 2016.

[BVV+16] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *CoNLL*, 2016.

[BYG+19] Imad Eddine Ibrahim Bekkouch, Youssef Youssry, Rustam Gafarov, Adil Khan, and Asad Masood Khattak. Triplet loss network for unsupervised domain adaptation. *Algorithms*, 2019.

[Car16] Lawrence Carin. On kernelized stein discrepancy, sampling-based recognition models, and gan. In *Internal Note*, 2016.

[Car17] L. Carin. On kernelized Stein discrepancy, sampling-based recognition models, and GAN. *Internal Notes*, 2017.

[CCD17] Benjamin Paul Chamberlain, James Clough, and Marc Peter Deisenroth. Neural embeddings of graphs in hyperbolic space. *arXiv preprint arXiv:1705.10359*, 2017.

[CCK+18] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018.

[CCL+08] Carlos M Carvalho, Jeffrey Chang, Joseph E Lucas, Joseph R Nevins, Quanli Wang, and Mike West. High-dimensional sparse factor modeling: applications in gene expression genomics. *Journal of the American Statistical Association*, 103(484):1438–1456, 2008.

[CDH+16] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NeurIPS*, 2016.

[CDP+17] Liqun Chen, Shuyang Dai, Yunchen Pu, Chunyuan Li, Qinliang Su, and Lawrence Carin. Symmetric variational autoencoder and connections to adversarial learning. *arXiv preprint arXiv:1709.01846*, 2017.

[CDT+18] Liqun Chen, Shuyang Dai, Chenyang Tao, Dinghan Shen, Zhe Gan, Haichao Zhang, Yizhe Zhang, and Lawrence Carin. Adversarial text generation via feature-mover's distance. *arXiv preprint arXiv:1809.06297*, 2018.

[CFG+15] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. *arXiv preprint arXiv:1512.03012*, 2015.

[CFHR17] Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. In *NeurIPS*, 2017.

[CFK+97] James W Cannon, William J Floyd, Richard Kenyon, Walter R Parry, et al. Hyperbolic geometry. *Flavors of geometry*, 1997.

[CFTR15] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *arXiv preprint arXiv:1507.00504*, 2015.

[CKF11] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A Matlab-like environment for machine learning. In *NeurIPS Workshop*, 2011.

[CLS+18] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *CVPR*, 2018.

[CMG+13] K. Cho, B. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *ACL*, 2013.

[COR+16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[CPK+17] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *PAMI*, 2017.

[CPK+18] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *PAMI*, 2018.

[CSG16] K. Chwialkowski, H. Strathmann, and A. Gretton. A kernel test of goodness of fit. In *ICML*, 2016.

[Csu17] Gabriela Csurka. *Domain adaptation in computer vision applications*. Springer, 2017.

[CSZ09] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning. *IEEE Transactions on Neural Networks*, 2009.

[CTZ+18] Liqun Chen, Chenyang Tao, Ruiyi Zhang, Ricardo Henao, and Lawrence Carin Duke. Variational inference and model selection with generalized evidence bounds. In *International conference on machine learning*, pages 893–902. PMLR, 2018.

[CVMG+14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[CXZZ17] LI Chongxuan, Taufik Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. In *NIPS*, 2017.

[DBP+16] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[DBP+17] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. In *ICLR*, 2017.

[DDH+18] Bo Dai, Hanjun Dai, Niao He, Weiyang Liu, Zhen Liu, Jianshu Chen, Lin Xiao, and Le Song. Coupled variational bayes via optimization embedding. In *NeurIPS*, 2018.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[DHS11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 2011.

[DJV+14] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

[DKD16] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[DKD17] J. Donahue, . Krähenbühl, and T. Darrell. Adversarial feature learning. In *ICLR*, 2017.

[DL17] Bo Dai and Dahua Lin. Contrastive learning for image captioning. In *NeurIPS*, 2017.

[DSGRS18] Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. Representation tradeoffs for hyperbolic embeddings. *Proceedings of machine learning research*, 2018.

[DSTB15] A. Dosovitskiy, J. T. Springenberg, M. Tatarchenko, and T. Brox. Learning to generate chairs, tables and cars with convolutional networks. *CVPR*, 2015.

[DYY+17] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Ruslan Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *NeurIPS*, 2017.

[FdCK14] Nazli Farajidavar, Teofilo de Campos, and Josef Kittler. Transductive transfer machine. In *ACCV*, 2014.

[FHST13] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, 2013.

[FLG+19] Le Fang, Chunyuan Li, Jianfeng Gao, Wen Dong, and Changyou Chen. Implicit deep latent variable models for text generation. In *EMNLP*, 2019.

[FLL+19] Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. In *NAACL*, 2019.

[FMF18] Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for domain adaptation. In *ICLR*, 2018.

[FRL+18] William Fedus, Mihaela Rosca, Balaji Lakshminarayanan, Andrew M Dai, Shakir Mohamed, and Ian Goodfellow. Many paths to equilibrium: Gans do not need to decrease adivergence at every step. In *ICLR*, 2018.

[FTT15] Basura Fernando, Tatiana Tommasi, and Tinne Tuytelaars. Joint cross-domain classification and subspace learning for unsupervised adaptation. *Pattern Recognition Letters*, 2015.

[GAA+17] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of Wasserstein GANs. In *NIPS*, 2017.

[GB05] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *NeurIPS*, 2005.

[GB10] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

[GBH18] Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. In *NeurIPS*, 2018.

[GBR+12] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *JMLR*, 2012.

[GCH+15] Z. Gan, C. Chen, R. Henao, D. Carlson, and L. Carin. Scalable deep poisson factor analysis for topic modeling. In *ICML*, 2015.

[GCHK19] Xiaodong Gu, Kyunghyun Cho, Jung-Woo Ha, and Sunghun Kim. Dialogwae: Multimodal response generation with conditional wasserstein auto-encoder. In *ICLR*, 2019.

[GCW+17] Zhe Gan, Liqun Chen, Weiyao Wang, Yuchen Pu, Yizhe Zhang, Hao Liu, Chunyuan Li, and Lawrence Carin. Triangle generative adversarial networks. In *NeurIPS*, 2017.

[GDDM14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[GDGW15] K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. Draw: A recurrent neural network for image generation. In *ICML*, 2015.

[GGS13a] Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013.

[GGS13b] Boqing Gong, Kristen Grauman, and Fei Sha. Reshaping visual datasets for domain adaptation. In *NeurIPS*, 2013.

[GGS14] Boqing Gong, Kristen Grauman, and Fei Sha. Learning kernels for unsupervised domain adaptation with applications to visual object recognition. *IJCV*, 2014.

[GH97a] Z. Ghahramani and G.E. Hinton. The em algorithm for mixtures of factor analyzers. Technical report, University of Toronto, 1997.

[GH97b] John J Godfrey and Edward Holliman. Switchboard-1 release 2. *Linguistic Data Consortium, Philadelphia*, 926:927, 1997.

[GHP07] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. *Caltech Technical Report.*, 2007.

[Gir15] R. Girshick. Fast r-cnn. In *ICCV*, 2015.

[GKDFF17] Timnit Gebru, Jonathan Krause, Jia Deng, and Li Fei-Fei. Scalable annotation of fine-grained categories without experts. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2017.

[GKZ+16] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *ECCV*, 2016.

[GL14] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.

[GLA19] Daniele Grattarola, Lorenzo Livi, and Cesare Alippi. Adversarial autoencoders with constant-curvature latent manifolds. *Applied Soft Computing*, 81:105511, 2019.

[GLC11] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, 2011.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[GPH+17] Zhe Gan, Yunchen Pu, Ricardo Henao, Chunyuan Li, Xiaodong He, and Lawrence Carin. Learning generic sentence representations using convolutional neural networks. In *EMNLP*, 2017.

[GSH+09] Arthur Gretton, Alexander J Smola, Jiayuan Huang, Marcel Schmittfull, Karsten M Borgwardt, and Bernhard Schölkopf. Covariate shift by kernel mean matching. In *MIT press*, 2009.

[GSSG12] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012.

[GUA+16] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016.

[GZH+16] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016.

[HCL06] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006.

[HFMC17] Philip Haeusser, Thomas Frerix, Alexander Mordvintsev, and Daniel Cremers. Associative domain adaptation. In *ICCV*, 2017.

[HG17] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.

[HGB+07] Jiayuan Huang, Arthur Gretton, Karsten M Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *NeurIPS*, 2007.

[HGLC15] Ricardo Henao, Zhe Gan, James Lu, and Lawrence Carin. Deep Poisson factor modeling. In *Submitted to Advances in Neural Information Processing Systems*, 2015.

[HGT+14] Judy Hoffman, Sergio Guadarrama, Eric S Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. Lsda: Large scale detection through adaptation. In *NeurIPS*, 2014.

[HHG+13] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *CIKM*, 2013.

[HJ16] Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NeurIPS*, 2016.

[HJO+01] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001.

[HKS+17] Lanqing Hu, Meina Kan, Shiguang Shan, Xingguang Song, and Xilin Chen. Ldf-net: Learning a displacement field network for face recognition across pose. In *IEEE International Conference on Automatic Face & Gesture Recognition*, 2017.

[HLDC16] S. Han, X. Liao, D.B. Dunson, and L. Carin. Variational gaussian copula inference. In *AISTATS*, 2016.

[HLND18] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. Adaptive semi-supervised learning for cross-domain sentiment classification. In *ACL*, 2018.

[HMP+17] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2(5):6, 2017.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[HSNBK19] Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. Lagging inference networks and posterior collapse in variational autoencoders. In *ICLR*, 2019.

[HSS18] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.

[HSW89] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 1989.

[HTP+17] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017.

[HWYD16] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016.

[HZRJ16] K. He, X. Zhang, S. Ren, and Sun J. Deep residual learning for image recognition. In *CVPR*, 2016.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[IFYA18] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *CVPR*, 2018.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[IZZE16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

[IZZE17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[JBP+02] S. V. Jones, S. Brown, T. Parker, V. Stubblefield, E. Kössler, C. Simmons, and J. Wittner. Anomalous indeterminate amorphous transitory gradations observed in certain specific nongeneralizable locii attendant to unique fleeting unpredictable phenomena. *Journal of Non-specific Research*, 88:665–703, 2002.

[JGJS99] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 1999.

[JSD+14] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACMMM*, 2014.

[JSZ+15] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *NeurIPS*, 2015.

[JZ07] Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in nlp. In *ACL*, volume 7, pages 264–271, 2007.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KB15] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[KCK+17] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jungkwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192*, 2017.

[Kim14] Yoon Kim. Convolutional neural networks for sentence classification. In *EMNLP*, 2014.

[KJYH19] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *CVPR*, 2019.

[KRMW14] D.P. Kingma, D.J. Rezende, S. Mohamed, and M. Welling. Semi-supervised learning with deep generative models. In *NIPS*, 2014.

[KSDFF13] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *CVPR Workshop*, 2013.

[KSES14] Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 3d object manipulation in a single photograph using stock 3d models. *Transactions on Graphics*, 2014.

[KSGP19] Minyoung Kim, Pritish Sahu, Behnam Gholami, and Vladimir Pavlovic. Unsupervised visual domain adaptation: A deep max-margin gaussian process approach. *arXiv preprint arXiv:1902.08727*, 2019.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[KSJ+16] D. P. Kingma, T. Salimans, R. Jozefowicz, X.i Chen, I. Sutskever, and M. Welling. Improving variational inference with inverse autoregressive flow. In *NIPS*, 2016.

[KSW+18] Abhishek Kumar, Prasanna Sattigeri, Kahini Wadhawan, Leonid Karlinsky, Rogerio Feris, Bill Freeman, and Gregory Wornell. Co-regularized alignment for unsupervised domain adaptation. In *NeurIPS*, 2018.

[KW13] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[KW14] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *ICLR*, 2014.

[KWKT15] Tejas D Kulkarni, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In *NeurIPS*, 2015.

[KWM+18] Yoon Kim, Sam Wiseman, Andrew C Miller, David Sontag, and Alexander M Rush. Semi-amortized variational autoencoders. In *ICML*, 2018.

[LAE+16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single Shot MultiBox Detector. In *ECCV*, 2016.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[LBD+89] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1989.

[LBK17] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017.

[LCWJ15] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[LCWJ18a] Mingsheng Long, ZHANGJIE CAO, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *NeurIPS*, 2018.

[LCWJ18b] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Domain adaptation with randomized multilinear adversarial networks. In *NeurIPS*, 2018.

[LDG+17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017.

[LGG+17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. In *ICCV*, 2017.

[LL12] H. Larochelle and S. Laulyi. A neural autoregressive topic model. In *NIPS*, 2012.

[LLC+17] Chunyuan Li, Hao Liu, Changyou Chen, Yuchen Pu, Liqun Chen, Ricardo Henao, and Lawrence Carin. Alice: Towards understanding adversarial learning for joint distribution matching. In *NIPS*, 2017.

[LLJ16] Q. Liu, J.D. Lee, and M. Jordan. A kernelized stein discrepancy for goodness-of-fit tests. In *ICML*, 2016.

[LLLS18] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NIPS*, 2018.

[LLR95] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[LLS+16] Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023*, 2016.

[LLS18] Shiyu Liang, Yixuan Li, and R Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.

[LRP07] F. Li, F. Rob, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 2007.

[LSC+16] C. Li, Andrew Stevens, Changyou Chen, Yunchen Pu, Zhe Gan, and Lawrence Carin. Learning weight uncertainty with stochastic gradient mcmc for shape classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5666–5675, 2016.

[LSLW15] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015.

[LSLW16] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016.

[LSS+17] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*, 2017.

[LT16] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *NIPS*, 2016.

[LW16] Q. Liu and D. Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *NIPS*, 2016.

[LWD+13] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *ICCV*, 2013.

[LZHFF17] Zelun Luo, Yuliang Zou, Judy Hoffman, and Li Fei-Fei. Label efficient learning of transferable representations across domains and tasks. In *NeurIPS*, 2017.

[LZL+17] Yucen Luo, Jun Zhu, Mengxi Li, Yong Ren, and Bo Zhang. Smooth neighbors on teacher graphs for semi-supervised learning. In *CVPR*, 2017.

[LZSZ15] C. Li, J. Zhu, T. Shi, and B. Zhang. Max-margin deep generative models. In *NIPS*, 2015.

[LZT+17] Chi Li, M Zeeshan Zia, Quoc-Huy Tran, Xiang Yu, Gregory D Hager, and Manmohan Chandraker. Deep supervision with shape concepts for occlusion-aware 3d object parsing. In *CVPR*, 2017.

[LZW+17] Zheng Li, Yu Zhang, Ying Wei, Yuxiang Wu, and Qiang Yang. End-to-end adversarial memory network for cross-domain sentiment classification. In *IJCAI*, 2017.

[LZWJ16] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *NeurIPS*, 2016.

[MG14] A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. In *ICML*, 2014.

[MHN13] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013.

[MKCK13] Boris Moiseev, Artem Konev, Alexander Chigorin, and Anton Konushin. Evaluation of traffic sign recognition methods trained on synthetically generated data. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, 2013.

[MLM+19] Emile Mathieu, Charline Le Lan, Chris J Maddison, Ryota Tomioka, and Yee Whye Teh. Continuous hierarchical representations with poincaré variational auto-encoders. In *NeurIPS*, 2019.

[MLX+16] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, and Zhen Wang. Multi-class generative adversarial networks with the l2 loss function. *arXiv preprint arXiv:1611.04076*, 2016.

[MMIK18] Takeru Miyato, Shin-ichi Maeda, Shin Ishii, and Masanori Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *PAMI*, 2018.

[MMR09] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *NeurIPS*, 2009.

[MNG16] L. Mescheder, S. Nowozin, and A. Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In *arXiv*, 2016.

[MNG17] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In *ICML*, 2017.

[MO14] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[MR16] A. Mnih and D. J. Rezende. Variational inference for monte carlo objectives. In *ICML*, 2016.

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*, 2013.

[MSJ+15] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[MSM93] Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 1993.

[Mül97] Alfred Müller. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 1997.

[MYB16] Y. Miao, L. Yu, and Phil Blunsomi. Neural variational inference for text processing. In *ICML*, 2016.

[Naw28] B. Nawahi. Acoustic production in liquid filled ceramic environments. *J. Chem. Phys.*, 108:9893–9904, 1928.

[NH99] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in Graphical Models. Cambridge, MA: MIT Press: pp. 355-368*, 1999.

[NK17] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *NeurIPS*, 2017.

[NWC+11] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop*, 2011.

[NYFK19a] Yoshihiro Nagano, Shoichiro Yamaguchi, Yasuhiro Fujita, and Masanori Koyama. A differentiable gaussian-like distribution on hyperbolic space for gradient-based learning. *arXiv preprint arXiv:1902.02992*, 2019.

[NYFK19b] Yoshihiro Nagano, Shoichiro Yamaguchi, Yasuhiro Fujita, and Masanori Koyama. A wrapped normal distribution on hyperbolic space for gradient-based learning. In *ICML*, 2019.

[OKK16] A. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural network. In *ICML*, 2016.

[Ovi19] Ivan Ovinnikov. Poincar\'e wasserstein autoencoder. *arXiv preprint arXiv:1901.01427*, 2019.

[PCK18] Yookoon Park, Jaemin Cho, and Gunhee Kim. A hierarchical latent structure for variational conversation modeling. In *NAACL*, 2018.

[PCLW18] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *AAAI*, 2018.

[PGH+16] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. In *NeurIPS*, 2016.

[PGH+17] Y. Pu, Z. Gan, R. Henao, C. Li, and L. Carin. Stein variational autoencoder. In *arXiv*, 2017.

[PRWZ02] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[PTKY11] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 2011.

[PTS+20] Sujoy Paul, Yi-Hsuan Tsai, Samuel Schulter, Amit K. Roy-Chowdhury, and Manmohan Chandraker. Domain adaptive semantic segmentation using weak labels. In *European Conference on Computer Vision (ECCV)*, 2020.

[PvdWRÁ16] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016.

[PWH+17] Yuchen Pu, Weiyao Wang, Ricardo Henao, Liqun Chen, Zhe Gan, Chunyuan Li, and Lawrence Carin. Adversarial symmetric variational autoencoder. In *NIPS*, 2017.

[PY+10] Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 2010.

[PYS+16] Y. Pu, X. Yuan, A. Stevens, C. Li, and L. Carin. A deep generative deconvolutional image model. *Artificial Intelligence and Statistics (AISTATS)*, 2016.

[PYY+17] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C. Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *CVPR*, 2017.

[R+66] R Tyrrell Rockafellar et al. Extension of fenchel'duality theorem for convex functions. *Duke mathematical journal*, 1966.

[RAY+16] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.

[RBH+15] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko. Semi-supervised learning with ladder networks. In *NIPS*, 2015.

[RDGF16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *CVPR*, 2016.

[RDS+14] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F. Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 2014.

[RDV+11] Salah Rifai, Yann N Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. The manifold tangent classifier. In *NeurIPS*, 2011.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 2015.

[RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NeurIPS*, 2015.

[RMC15] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[RMC16] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

[RMea15] D.J. Rezende, S. Mohamed, and et al. Variational inference with normalizing flows. In *ICML*, 2015.

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

[RNR+17] Konstantinos Rematas, Chuong H Nguyen, Tobias Ritschel, Mario Fritz, and Tinne Tuytelaars. Novel views of objects from a single image. *PAMI*, 2017.

[ROPV19] Ali Razavi, Aäron van den Oord, Ben Poole, and Oriol Vinyals. Preventing posterior collapse with delta-vaes. In *ICLR*, 2019.

[RTB16] R. Ranganath, D. Tran, and D. M. Blei. Hierarchical variational models. In *ICML*, 2016.

[RTCM15] R. Ranganath, L. Tang, L. Charlin, and D. M.Blei. Deep exponential families. In *AISTATS*, 2015.

[RVRK16] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016.

[San05] F. Sanford. Oxidized ferrous materials. *Journal of Junk*, 5(4):324–345, 2005.

[Sar11] Rik Sarkar. Low distortion delaunay embedding of trees in hyperbolic plane. In *International Symposium on Graph Drawing*, pages 355–366. Springer, 2011.

[SBNE18] Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. In *ICLR*, 2018.

[SDBR15] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR workshop*, 2015.

[SDVG18] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *IJCV*, 2018.

[SFS16] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016.

[SGZ+16] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NeurIPS*, 2016.

[SH70] M. Smith and I. A. Hall. *Handbook of Interstellar Travel*. Dover, New York, 7th edition, 1970.

[SH09] R. Salakhutdinov and G. E. Hinton. Deep boltzmann machines. In *AISTATS*, 2009.

[SHH16] Jakub Sochor, Adam Herout, and Jiri Havel. Boxcars: 3d boxes as cnn input for improved fine-grained vehicle recognition. In *CVPR*, 2016.

[SHK+14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 2014.

[SKCK17] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *ICLR*, 2017.

[SKFD10] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, 2010.

[SKP15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

[SLBJ17] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style transfer from non-parallel text by cross-alignment. In *NeurIPS*, 2017.

[SLD17] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *PAMI*, 2017.

[SLJ+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[SLZ+17] Kihyuk Sohn, Sifei Liu, Guangyu Zhong, Xiang Yu, Ming-Hsuan Yang, and Manmohan Chandraker. Unsupervised domain adaptation for face recognition in unlabeled videos. In *ICCV*, 2017.

[Spr15] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. In *ICLR*, 2015.

[SPT+17] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017.

[SRM+16] C.K. Sonderby, T. Raiko, L. Maaloe, S.K. Sonderby, and O. Winther. Ladder variational autoencoders. *NIPS*, 2016.

[SS16] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV Workshop*, 2016.

[SSND18] Xiaoyu Shen, Hui Su, Shuzi Niu, and Vera Demberg. Improving variational encoder-decoders in dialogue generation. In *AAAI*, 2018.

[SSSI11] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *IJCNN*, 2011.

[SSSS16] Ozan Sener, Hyun Oh Song, Ashutosh Saxena, and Silvio Savarese. Learning transferrable representations for unsupervised domain adaptation. In *NeurIPS*, 2016.

[STS+20] Jong-Chyi Su, Yi-Hsuan Tsai, Kihyuk Sohn, Buyu Liu, Subhransu Maji, and Manmohan Chandraker. Active adversarial domain adaptation. In *WACV*, 2020.

[SUH17] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *ICML*, 2017.

[SUHS17] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Adversarial dropout regularization. *arXiv preprint arXiv:1711.01575*, 2017.

[SVL14] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

[SWUH18] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018.

[SYS+20] Huajie Shao, Shuochao Yao, Dachun Sun, Aston Zhang, Shengzhong Liu, Dongxin Liu, Jun Wang, and Tarek Abdelzaher. Controlvae: Controllable variational autoencoder. In *ICML*, 2020.

[SZ15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[SZM+19] Wenxian Shi, Hao Zhou, Ning Miao, Shenjian Zhao, and Lei Li. Fixing gaussian mixture vaes for interpretable text generation. *arXiv preprint arXiv:1906.06719*, 2019.

[TB99] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[TB15] L. Theis and M. Bethge. Generative image modeling using spatial lstms. In *NIPS*, 2015.

[TBG18] Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea. Poincar\'e glove: Hyperbolic word embeddings. *arXiv preprint arXiv:1810.06546*, 2018.

[TCD+19] Chenyang Tao, Liqun Chen, Shuyang Dai, Junya Chen, Ke Bai, Dong Wang, Jianfeng Feng, Wenlian Lu, Georgiy V Bobashev, and Lawrence Carin. On fenchel mini-max learning. In *NeurIPS*, pages 10427–10439, 2019.

[TDB16] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Multi-view 3d models from single images with a convolutional network. In *ECCV*, 2016.

[TE11] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR*, 2011.

[TH12] T. Tieleman and G. Hinton. Lecture 6.5 rmsprop. *Technical report*, 2012.

[THDS15] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, 2015.

[THS+18] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018.

[THSD17] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017.

[THZ+14] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[TOB16] L. Thei, A. Oord, and M. Bethge. A note on the evaluation of generative models. In *ICLR*, 2016.

[TPW16] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200*, 2016.

[Tre01] H.L. Van Trees. Detection, estimation, and modulation theory. In *Wiley*, 2001.

[TŠ13] Radim Tyleček and Radim Šára. Spatial pattern templates for recognition of objects with regular structure. In *GCPR*, 2013.

[TSSC19] Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, and Manmohan Chandraker. Domain adaptation for structured output via discriminative patch representations. In *ICCV*, 2019.

[TSY+18] Luan Tran, Kihyuk Sohn, Xiang Yu, Xiaoming Liu, and Manmohan Chandraker. Joint pixel and feature-level domain adaptation in the wild. *arXiv preprint arXiv:1803.00068*, 2018.

[TSZY15] Ben Tan, Yangqiu Song, Erheng Zhong, and Qiang Yang. Transitive transfer learning. In *SIGKDD*, 2015.

[TT13] E.G. Tabak and C.V. Turner. A family of nonparametric density estimation algorithms. *Comm. Pure & Appl. Math.*, 2013.

[TW18] Jakub M Tomczak and Max Welling. Vae with a vampprior. In *AISTATS*, 2018.

[TYL17] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, 2017.

[Ung08] Abraham Albert Ungar. A gyrovector space approach to hyperbolic geometry. *Synthesis Lectures on Mathematics and Statistics*, 1(1):1–194, 2008.

[UVL16] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[VDM14] Laurens Van Der Maaten. Accelerating t-sne using tree-based algorithms. *JMLR*, 2014.

[VLL+10] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *JMLR*, 2010.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[VTBE15] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015.

[WBM+10] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical report, California Institute of Technology, 2010.

[WBSG16] Yuhuai Wu, Yuri Burda, Ruslan Salakhutdinov, and Roger Grosse. On the quantitative analysis of decoder-based generative models. *arXiv preprint arXiv:1611.04273*, 2016.

[WBW+11] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset, 2011.

[WFB17] David Warde-Farley and Yoshua Bengio. Improving generative adversarial networks with denoising feature matching. In *ICLR*, 2017.

[WGX+19] Wenlin Wang, Zhe Gan, Hongteng Xu, Ruiyi Zhang, Guoyin Wang, Dinghan Shen, Changyou Chen, and Lawrence Carin. Topic-guided variational autoencoders for text generation. *arXiv preprint arXiv:1903.07137*, 2019.

[WL16] D. Wang and Q. Liu. Learning to samples: With application to amoritized mle for generalized adversarial learning. In *arXiv:1611.01722v2*, 2016.

[WSL+14] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, 2014.

[WSL17] Liwei Wang, Alexander Schwing, and Svetlana Lazebnik. Diverse and accurate image description using a variational auto-encoder with an additive gaussian encoding space. In *NeurIPS*, 2017.

[WTG+19] Wenlin Wang, Chenyang Tao, Zhe Gan, Guoyin Wang, Liqun Chen, Xinyuan Zhang, Ruiyi Zhang, Qian Yang, Ricardo Henao, and Lawrence Carin. Improving textual network learning with variational homophilic embeddings. *arXiv preprint arXiv:1909.13456*, 2019.

[XBK+15] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

[XD18] Jiacheng Xu and Greg Durrett. Spherical latent spaces for stable variational autoencoders. In *EMNLP*, 2018.

[XWW+16] Lingxi Xie, Jingdong Wang, Zhen Wei, Meng Wang, and Qi Tian. Disturblabel: Regularizing cnn on the loss layer. In *CVPR*, 2016.

[XZCC18] Shaoan Xie, Zibin Zheng, Liang Chen, and Chuan Chen. Learning semantic representations for unsupervised domain adaptation. In *ICML*, 2018.

[XZH+17] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. *arXiv preprint arXiv:1711.10485*, 2017.

[YCBL14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NeurIPS*, 2014.

[YG14] Aron Yu and Kristen Grauman. Fine-grained visual comparisons with local learning. In *CVPR*, 2014.

[YHSBK17] Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. Improved variational autoencoders for text modeling using dilated convolutions. In *ICML*, 2017.

[YJ16] Jianfei Yu and Jing Jiang. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In *EMNLP*, 2016.

[YLCLT15] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*, 2015.

[YRYL15] Jimei Yang, Scott E Reed, Ming-Hsuan Yang, and Honglak Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In *NeurIPS*, 2015.

[YVDY18] Michelle Yuan, Benjamin Van Durme, and Jordan L Ying. Multilingual anchoring: Interactive topic modeling and alignment across languages. In *NeurIPS*, 2018.

[YYS+17] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Towards large-pose face frontalization in the wild. In *ICCV*, 2017.

[YYSL16] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. Attribute2image: Conditional image generation from visual attributes. In *ECCV*, 2016.

[YZTG17] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *CVPR*, 2017.

[YZWY17] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, 2017.

[ZFY+10] Erheng Zhong, Wei Fan, Qiang Yang, Olivier Verscheure, and Jiangtao Ren. Cross validation framework to choose amongst models and datasets for transfer learning. In *ECML-KDD*, 2010.

[ZGF+17] Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation. In *ICML*, 2017.

[ZGL+17] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. *arXiv preprint arXiv:1702.08811*, 2017.

[ZHDC12] M. Zhou, L. Hannah, D. Dunson, and L. Carin. Beta-negative binomial process and Poisson factor analysis. In *AISTATS*, 2012.

[ZHPA13] James Y Zou, Daniel J Hsu, David C Parkes, and Ryan P Adams. Contrastive learning using spectral methods. In *NeurIPS*, 2013.

[Zhu05] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.

[ZKSE14] J.-Y. Zhu, P. Krahenbuhl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *ECCV*, 2014.

[ZKSE16] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *ECCV*, 2016.

[ZKZ+17] Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M Rush, and Yann LeCun. Adversarially regularized autoencoders. *arXiv preprint arXiv:1706.04223*, 2017.

[ZML16] J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. *ICLR*, 2016.

[ZML17] J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. In *ICLR*, 2017.

[ZPIE17] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

[ZR18] Yftah Ziser and Roi Reichart. Pivot based language modeling for improved neural domain adaptation. In *ACL*, 2018.

[ZR19] Zachary M Ziegler and Alexander M Rush. Latent normalizing flows for discrete sequences. In *ICML*, 2019.

[ZSE18] Shengjia Zhao, Jiaming Song, and Stefano Ermon. Infovae: Information maximizing variational autoencoders. In *AAAI*, 2018.

[ZTS+16] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *ECCV*, 2016.

[ZXL+17] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017.

[ZXS+16] Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. Variational neural machine translation. In *EMNLP*, 2016.

[ZZE17] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *ACL*, 2017.

[ZZW+17] Han Zhao, Shanghang Zhang, Guanhang Wu, João P Costeira, José MF Moura, and Geoffrey J Gordon. Multiple source domain adaptation with adversarial training of neural networks. *arXiv preprint arXiv:1705.09684*, 2017.

[ZZW+18] Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. Adversarial multiple source domain adaptation. In *NIPS*, 2018.

# Biography

Shuyang Dai is currently a Ph.D. candidate at Duke University, Electrical and Computer Engineering department. In 2016, he received his Bachelor's degree in University of California, Los Angeles. Starting from August 2016, Shuyang was admitted to Duke University to study towards the Ph.D. degree, advised by Professor Lawrence Carin. His research interests lie in the deep generative model and unsupervised domain adaptation with applications in computer vision and natural language processing. During his Ph.D. study, Shuyang has published on top tier conferences, including NeurIPS, ICML, AAAI, NAACL, BMVC, ACCV, and AISTATS.