Game-Theoretic Models for Generative Learning

PhD Thesis Proposal

Abstract

Machine learning has achieved great success in classification tasks thanks to the powerful deep neural networks. However, there's still a big gap between computer and human intelligence. Discriminative learning attempts to infer knowledge from data by modeling the conditional distribution of outputs given inputs, while generative learning tries to estimate the entire distribution of data and produces new samples. My thesis will focus on the latter problem and investigate generative learning from a game-theoretic perspective. We first formulate the problem as a distributionally robust game with payoff uncertainty, and then develop a robust optimization algorithm to solve the Nash equilibrium. Meanwhile, we will study the distance metric to measure similarity between distributions, which is vital to build the objective functions. Last, we plan to test our approach in simulations and apply it to train several generative models for images and audio.

1 Introduction

In the last decade, machine learning approaches have achieved great success with the rapid development of deep neural networks. I identify three levels of artificial intelligence: memorization, recognition and generation. Computers are approaching humans on the first two levels, and now going to the third.

Apart from recognition and classification tasks, people hope to learn the entire distribution of data and generate new samples. In mathematics, it means to learn a function describing the structure of unlabeled data, for example, density estimation is a direct application. It requires inferring the real data distribution given a set of observations. There's no straightforward way to evaluate the accuracy of generative models, because the goal is to produce lifelike artificial examples and the evaluation is subjective in most cases.

In generative learning, new samples produced by the learned model should be indistinguishable from the original data and have enough diversity. Generative models are powerful tools for many tasks such as image and audio generation, video prediction, style transfer, voice conversion, and semi-supervised learning.

If we work on low-dimension data, or just want some statistics instead of producing new samples, there's no need to model the data distribution. A bunch of unsupervised learning algorithms work well, e.g., k-means, Gaussian mixture model, EM, PCA, etc. But for complex tasks such as generating images, audios and videos, neural networks demonstrate more power. There are three popular mainstream methods in deep generative learning, generative adversarial network (GAN), variational auto-encoders (VAE) and autoregressive models (e.g., WaveNet, PixelRNN).

In general, training deep generative models is hard and time consuming. For example, it requires 8 GPUs training 6 days to learn the WaveNet autoencoder. The high dimensional training data and complex objective structures lead to many problems in optimization, such as algorithm instability, saturation, and mode collapse. Moreover, the model should have strong generalization power to produce diverse artificial examples instead of just memorizing the training set.

In this research work, we plan to develop a new game-theoretic framework for generative learning. We formulate the problem as a distributionally robust game with payoff uncertainty, and develop a learning algorithm to solve the robust Nash equilibrium. In this game there are several groups of players that are competitive, noncooperative and have different objectives. Each player works in a continuous action space to optimize its expected worst-case performance. The players are implemented with neural network models to perform prediction, classification or data generation tasks. Agents with similar objectives form a group and work together against the others in order to optimize their expected payoffs. The distributionally robust Nash equilibrium is achieved by solving a minimax optimization problem. We consider using stochastic optimization techniques to deal with large-scale learning tasks.

Iterative optimization algorithms travel to the equilibria by minimizing a loss function, which is in our case a distance metric defined to measure the similarity between two distributions. As an important part of this research, We will study the pros and cons of several kinds of distance metrics, and compare Wasserstein distance with the most prevalent information-based metrics such as Kullback-Leibler and Jensen-Shannon divergence. We plan to develop a practical method to approximately calculate the distance between distributions and give theoretical analysis and numerical evaluations.

We will first verify the theoretical results through simulations, and then apply our approach on real datasets for image and audio generation. We plan to work on several tasks such as object detection, vehicle tracking, image generation and audio synthesis. This research contributes to the areas of distributionally robust game, deep generative learning, stochastic optimization and time-series data analysis.

2 Distributionally Robust Games

2.1 Introduction

2.1.1 Distribution Uncertainty Set

2.1.2 Related Work

2.2 Problem Formulation

2.2.1 From unsupervised learning to Generative Model

2.2.2 Game Theoretic Framework for Learning

2.2.3 Definition

2.2.4 The Existence of Distributionally Robust Nash Equilibria

2.3 Minimax Robust Game

2.3.1 From Duality to Triality Theory

2.3.2 Dimension Reduction

2.3.3 Evaluation

2.4 Case Study: Learning a Generative Adversarial Model

3 Wasserstein Metric

All headings should be lower case (except for first word and proper nouns), flush left, and bold.

First-level headings should be in 12-point type.

3.1 Introduction

3.1.1 Optimal Transportation Problem

3.1.2 Definition

Second-level headings should be in 10-point type.

3.2 From KL divergence to Wasserstein Metric

Third-level headings should be in 10-point type.

3.3 Other Metrics: L1, L2, Maximum Mean Discrepancy

3.4 Dynamic Optimal Transport

There is also a `\paragraph` command available, which sets the heading in bold, flush left, and inline with the text, with the heading followed by 1 em of space.

3.5 Case Study: A Toy Example

4 Learning Algorithms

These instructions apply to everyone.

4.1 Bregman Learning under f-divergence

The `natbib` package will be loaded for you by default. Citations may be author/year or numeric, as long as you maintain internal consistency. As to the format of the references themselves, any style is acceptable as long as it is used consistently.

The documentation for `natbib` may be found at

```
http://mirrors.ctan.org/macros/latex/contrib/natbib/natnotes.pdf
```

Of note is the command `\citet`, which produces citations appropriate for use in inline text. For example,

```
\citet{hasselmo} investigated\dots
```

produces

> Hasselmo, et al. (1995) investigated…

If you wish to load the `natbib` package with options, you may add the following before loading the `nips_2018` package:

```
\PassOptionsToPackage{options}{natbib}
```

If `natbib` clashes with another package you load, you can add the optional argument `nonatbib` when loading the style file:

```
\usepackage[nonatbib]{nips_2018}
```

4.2 Distributionally Robust Optimization

Footnotes should be used sparingly. If you do require a footnote, indicate footnotes with a number[1] in the text. Place the footnotes at the bottom of the page on which they appear. Precede the footnote with a horizontal rule of 2 inches (12 picas).

Note that footnotes are properly typeset *after* punctuation marks.[2]

4.3 Train a Deep Generative Model

All artwork must be neat, clean, and legible. Lines should be dark enough for purposes of reproduction. The figure number and caption always appear after the figure. Place one line space before the figure

[1] Sample of the first footnote.
[2] As in this example.

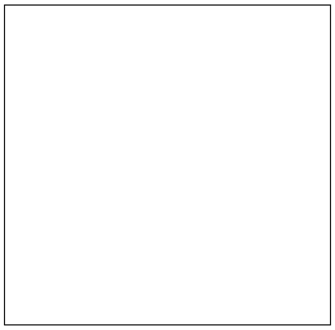

Figure 1: Sample figure caption.

Table 1: Sample table title

Part		
Name	Description	Size (μm)
Dendrite	Input terminal	$\sim$100
Axon	Output terminal	$\sim$10
Soma	Cell body	up to 10^6

caption and one line space after the figure. The figure caption should be lower case (except for first word and proper nouns); figures are numbered consecutively.

You may use color figures. However, it is best for the figure captions and the paper body to be legible if the paper is printed in either black/white or in color.

4.4 Case Study: Unsupervised Learning for Clustering

All tables must be centered, neat, clean and legible. The table number and title always appear before the table. See Table 1.

Place one line space before the table title, one line space after the table title, and one line space after the table. The table title must be lower case (except for first word and proper nouns); tables are numbered consecutively.

Note that publication-quality tables *do not contain vertical rules.* We strongly suggest the use of the `booktabs` package, which allows for typesetting high-quality, professional tables:

`https://www.ctan.org/pkg/booktabs`

This package was used to typeset Table 1.

4.5 Case Study: Generative Modeling for Image Synthesis

5 Style Transfer as a Minimax Game

5.1 Introduction

Style transfer originally means rendering an image in different styles. This meaning can be extended to other kinds of data like music and speech. More generally, it refers to mapping data from one domain to anther while keeping its semantic (underlying) content / the domain-invariant knowledge. For example, transfer photographs to artistic paintings, convert one person's voice to another, or translate music to imitate different instruments. Another case is transfer learning. It adapts a pre-trained model in source domain to classify samplpes in target domain where labeled data is limited.

Let $x_1 \in \mathbb{X}_1$ be samples in the source domain and $x_2 \in \mathbb{X}_2$ be samples in the target domain. The goal of style transfer is to learn a mapping function $\mathcal{T} : \mathbb{X}_1 \to \mathbb{X}_2$ such that the generated output $x'_{2\leftarrow 1} = \mathcal{T}(x_1)$ is indistinguishable from the real samples drawn from the target domain. The optimal mapping $\mathcal{T}^*$ transforms x_1 to $x'_{2\leftarrow 1}$ such that $x'_{2\leftarrow 1} \overset{d}{=} x_2$. Semantic content should be preserved during the transformation.

We propose a game-theoretic approach to learn the mapping. The domain-invariant content information and domain-specific style information are decomposed by disentangled representation learning. For high dimensional data like image and speech waveform, we employ autoencoders to independently model the high-level semantic content and the low-level style information. The learning problem is formulated as a distributionally robust game with cooperative agents and payoff uncertainty. In this game, several groups of players run with different objectives. The intergroup competition and intragroup collaboration enable the players to learn from each other and optimize their worst-case performance.

This work has a wide range of applications. In visual and performing arts, it's inspiring to automatically generate artificial paintings with user-specified style or play synthetic music with desired timbre and musical instrument. In informatics, it's useful to transform speaker identity by modifying his voice to sound like another person. It is also possible to learn and mimic animal's vocalization and study the feedback on the artificially generated sound. For case study, we apply our approach in two scenarios: image style transfer and emotional voice conversion.

5.2 Motivation

In machine learning, discriminative models predict labels from data by learning a conditional distribution $p(y|x)$, while generative models synthesize new data with desired labels by drawing samples from estimated distribution $p(x|y)$. From a perspective of probabilistic modeling, style transfer learns two conditions $p(x_1|x_2)$ and $p(x_2|x_1)$. When paired data is available, it is easy to infer from the joint distribution $p(x_1, x_2)$. For nonparallel data, the problem is ill-posed because the joint solution is not unique given two marginals $p(x_1), p(x_2)$.

Additional conditions are required to handle this problem. Some suggested to keep some parts of the data unchanged, such as pixel intensity, gradient or object boundary [2][3]; others proposed to preserve certain properties of the samples after style transfer, for example semantic features and class labels [4].

Gatys et al. [7] introduced an algorithm to separate and recombine the content and style of natural images. They claimed that a Convolutional Neural Network (CNN) is the ideal representation to factorize semantic content and artistic style. High-level features are extracted by higher layers of the network, while low-level features are captured by the correlations between filter responses in various layers. However, the style is learnt from a single image, which limits the ability to capture the general theme of the target domain.

Zhu et al. [10] proposed a very straightforward constraint called cycle-consistency. It assumes that if a sample is translated from source domain to target domain and then translated back, it should be unchanged. Choi et al. [11] generalized it to perform multiple-domain translation using a single generative model. However, domain transfer is not a one-to-one mapping, but many-to-many. In some cases, the cycle-consistency constraint is too strong to provide enough diversity in the translated outputs.

Based on a similar idea, Liu et al. [8] developed the UNIT framework by making a fully shared latent space assumption, in which corresponding images across domains can be mapped to a same latent code in shared-latent space. This assumption implies the cycle-consistency constraint. Xun et al. [9] extended it to a partially shared latent space assumption, where each example is generated from a shared content code and a domain-specific style code. Images are translated across domains by replacing the style code.

Other approaches [11][12][13] assume there exist a transformation $\mathcal{T}$ so that the source and target distributions can be matched by the transformed representations, $p(\mathcal{T}(x_1)) = p(\mathcal{T}(x_2))$. It requires minimizing the distance between distributions, which has been discussed in chapter 3. It is equivalent to searching for a transporation plan $\mathcal{T}$ such that $p(\mathcal{T}(x_1)) = p(x_2)$. A particular solution is the optimal plan that minimizes the transportation cost, i.e., the Wasserstein distance between the source and target data points. As a by-product, minimizing the transporation cost gives an alignment of the corresponding samples as well as their labels. Based on this idea, Damodaran et al. [14] proposed to minimize the Wasserstein distance between joint distributions $p(x_1, y_1)$ and $p(x_2, f(x_2))$, where f is the classifier in the target domain. Category knowlege is transferred between domains by matching the data-label pairs.

5.3 Related Work

Learning generative models for style transfer is an open problem. There are several topics closely related to this work, but with different definitions.

Deep Generative Modeling The original objective of this topic is to generate new samples from scratch by learning complicated data distributions in an unsupervised way. At test time, it takes random noise as input and outputs realistic samples. In some cases, it can also take in conditional information to produce user-specified output. There are three main frameworks of deep generative modeling: generative adversarial networks (GANs) [15], variational auto-encoders (VAEs) [16], and auto-regressive models [17].

GANs build the generative model on the top of a discriminative network to force the output to be indistinguishable from the real samples. This model works pretty well for generating images with impressive visual quality [18] and high resolution [19]. Variations under this framework include conditional GAN [20] that generate samples conditioned on class labels, LAPGAN [21] that generates images in a coarse-to-fine fashion, WGAN-GP [22] that enables stable training of GANs without hyperparameter tuning.

VAEs use an encoder-decoder framework to model data in a latent space and optimize the reconstruction loss plus a regularizer. The generative process has two steps of sampling: first draw latent variables from $p(z)$ and then draw datapoints from the conditional distribution $p(x|z)$. At test time, the encoder part is discarded and the decoder takes random noise as input to generate new samples. However, the reconstructed samples are blurry. This is because the VAE decoder assumes $p(x|z)$ to be an isotropic Gaussian, which leads to the use of L2 loss. To remedy this, VAE-GAN [23] suggests learning the loss through a GAN discriminator.

Auto-regressive model is quite different from the above two. It aims at modeling time-varying processes by assuming that the value of a time series depends on its previous values and a stochastic term. For a sequential data sample $x = (x_1, x_2, \ldots, x_T)$, the joint distribution $p(x)$ is factorised as a product of conditional distributions

$$p(x) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}) \tag{1}$$

This idea is quite straightforward for modeling audio sequence [24], but it also works for images. In PixelRNN [25], each image is written as a sequence, in which pixels are taken row by row from the image. The two-dimensional spatial autocorrelation of pixels is modeled by one-dimensional temporal correlations. Since the generation process is sequential, it requires a lot of GPU memory and computation time (200K updates over 32 GPUs) even after some modifications [26].

Image Style Transfer There are two types of style transfer problems: example-based style transfer where the style comes from one image, and domain-based style transfer where the style is learnt from a collection of images in a specific domain. The former problem originates from nonphotorealistic rendering (NPR) [27] in computer graphic, and has the similar meaning of realistic image manipulation. The goal is to edit image in a user-specified way and keep it as realistic as possible. Practical issues include texture synthesis and transfer [28], photo manipulation of shape and color [29], photorealistic image stylization [30], etc. In general, the output should be similar to the input in high-level structures and varies in low-level details such as color and texture.

Recently, Gatys et al. [31] claimed the image content and style information are separable in Convolutional Neural Network representations. They introduced a method [32] to separate and recombine content and style of natural images by matching feature correlations (Gram matrix) in different convolutional layers. However, their synthesis process is slow (an hour for a 512*512 image). Moreover, the style from a single image is ambiguous and may not capture the general theme of an entire domain of images.

The second problem, also named as image-to-image translation, learns a mapping to transfer images from one domain to another. For example, super-resolution [39] maps low-dimentional images to high-dimention, colorization [40] maps gray images to color; other cases include day to night, dog to cat, yong to old, summer to winter, photographs to paintings, aerial photos to maps [30,34,35,36,37,38,41]. The mapping can be learnt in a supervised or unsupervised manner. In supervised settings [33,42,43], corresponding image pairs across domains are available for training. In unsupervised settings [2,4,8,9,10], there's no paired data and the training set only contains independent set of images for each domain. Our work is under the unsupervised settings because it is more applicable, and the training data is almost free and unlimited.

Domain Adaptation Most recognition algorithms are developed and evaluated on the same data distributions, e.g, the public datasets ImageNet, MS-COCO, CIFAR-10, MNIST. In real applications, people often confront performance degradation when apply a classifier trained on a source domain to a target domain.

In unsupervised domain adaptation, source domain has labeled data $\mathcal{D}_s = \{x_i^s, y_i^s\}_{i=1}^{n_s}$ while target domain contains data without labels $\mathcal{D}_t = \{x_i^t\}_{i=1}^{n_t}$. The goal is to learn a classifier $f : x_i^t \mapsto y_i^t$ for the unseen target samples by exploring the knowledge learnt from the source domain. Domain adaptation algorithms attempts to transfer knowledge across domains by solving the domain shift problem, i.e., the data-label distributions $p(x^s, y^s)$ and $p(x^t, y^t)$ are different.

There are many approaches to address this issue. One is to extract transferable features that are invariant across domains [45,46], or learn representative hash codes [47] to find a common latent space where the classifier can be used without considering the data's origin. Another trend is to learn the transformation between domains [48] to align the source and target datapoints through barycentric mapping, and train a classifier on the transferred source data. Courty [49] and Damodaran[14] proposed to look for a transformation that matches the data-label joint distributions $p(x^s, y^s)$ in source domain to its equivalent version $p(x^t, y^t)$ in target domain. The predictive function f is learnt by minimizing the optimal transport loss between the distributions $p(x^s, y^s)$ and $p(x^t, f(x^t))$. As a by-product, minimizing the optimal transport cost is equivalent to mapping a source domain sample to a target domain sample with similar semantic content, and this is the domain transfer problem.

Voice Conversion Voice conversion (VC) aims to change a speaker's voice to make it sounds like spoken by another person. It is a special case of voice transformation (VT), whose goal is to modify human speech without changing its content. VC transforms speacker identity by replacing speaker-dependent components of the signal while maintaining the linguistic information. Speech quality and speaker similarity are two important factors to evaluate a VC system. There are a bunch of VC applications, such as movie dubbing, personalized TTS (Text To Speach) systems, speaker accent or emotion transformation, speaking-aid devices, call quality enhancement, etc.

Most VC frameworks involve three steps: feature extraction, feature conversion, waveform generation. In speech analysis, waveform signals are encoded into feature representations that are easy to control and modify. Spectral envelope, mel-cepstrum, fundamental frequency ($f0$), formant frequencies and bandwidths are the most widely used features to represent speech in short-time segments. To capture contextual information across frames, implicit methods such as hidden Markov models (HMMs), Long Short-Term Memory (LSTM) and recurrent neural networks (RNNs) [63] were developed.

The main work in VC is to transform the source feature sequences to target feature sequences that capture the speaker identity. Most traditional VC systems perform frame-by-frame mapping under the assumption that speech segments are independent from each other. Some recent models such as HMM and RNN incorporate speech dynamics implicitly. There are four typical approaches to learn the mapping function: codebook mapping (e.g., Vector quantization (VQ) [56]), mixed linear mappings (e.g., Gaussian mixture model (GMM) [57]), neural network mapping (e.g., RBM, DNN, RNN [55]), and exemplar-based mapping (e.g., non-negative matrix factorization (NMF) [58]). Beyond these, an autoregressive neural network model called WaveNet [24] was proposed. It can directly learn the mapping based on raw audio and generate speech waveforms conditioning on the speaker identity.

There are various assumptions in speech analysis and waveform generation. Source-filter models assume speech to be generated by excitation signals passing through a vocal tract, and encode speech waveforms as acoustic features that represent sound source and vocal tract independently. However, the original phase information will lose under this assumption. At conversion time, the converted target features are passed through a vocoder based on the source filter model to reconstruct the waveform. Quality degradation may happen due to the inaccurate assumption. Iterative phase reconstruction algorithm Griffin-Lim [59] was adopted to aleviate this issue. Harmonic plus noise models (HNM) [54] assume speech to be a combination of a noise component and a harmonic component, i.e., sinusoidal waves with frequencies relevant to pitch. Speech is parameterized by the fundamental frequency f_0 and a spectrum which consists of a lower band of harmonic and a higher band of noise. Other assumptions include stationary speech signal, frame-by-frame mapping, time-invariant linear filter, etc. Recently, Tamamori et. al [53] proposed a speaker-dependent WaveNet vocoder that does not require explicit modeling of excitation signals and those assumptions.

In terms of conversion conditions, VC can be categorized into parallel and non-parallel, text-dependent and text-independent systems [50]. In parallel systems, the training corpus consists of paired recodings from the source and target spearkers with same liguistic contents. The shared acoustic features can be used to train the mapping model. To get parallel feature sequences of equal length, a time-alignment step must be included to remove the temporal differences in the recordings, for example, the dynamic time warping (DTW) [56] algorithm. Phoneme transcriptions are also useful for time alignment. Non-parallel system does not require sentences with the same linguistic contents. It is much more useful and practical because non-parallel speech data is easier to collect and therefore can get larger training sets. There are several ways to learn the mapping without paired data: (1) use unit selection [60] to choose matched linguistic feature pairs; (2) build pseudo parallel sentences on extra automatic speech recognition (ASR) modules [61]; (3) extract speaker-independent features in shared latent space [62]; (4) use unpaired image-to-image translation approaches [8][9][33].

Parallel, text-dependent systems are supposed to have better performancce. However, parallel utterance pairs are difficult to get. Most parallel VC systems require time alignment to extract parallel source-target features. The misalignment in automatic time alignment algorithms often leads to degradation in speech quality, while manual correction is arduous. Recently, the winner of VC Challenge 2018 [51] showed their algorithm can achieve similar results in both parallel and non-parallel settings. It first uses a lot of external speech data with phonetic transcriptions to train a speaker-independent content-posterior-feature extractor, followed by a speaker-dependent LSTM-RNN to predict fundamental frequency f_0 and STRAIGHT spectral features [52], and then reconstruct the waveforms with a speaker-dependent WaveNet vocoder [53]. Moreover, Kaneko et.al [64] and Fang et. al [54] claimed their nonparallel, text-independent VC algorithms based on CycleGAN [10] perform comparable to or better than the state-of-the-art parallel approaches.

5.4 Method

We propose a general approach for style transfer. Let $x_i \in \mathbb{X}_i$ be a datapoint sampled from domain i. The goal is to learn a conversion model $p(x_j|x_i)$ that maps x_i to domain j ($j \neq i$) by changing its style and preseving the original content. For unpaired training data, we have marginal distributions $p(x_i), p(x_j)$ instead of the joint $p(x_i, x_j)$, so it requires additional constraints to determine $p(x_j|x_i)$. Inspired by disentangled representation learning in [32], we assume that each example $x_i \in \mathbb{X}_i$ can be decomposed into a content code $c \in \mathcal{C}$ that encodes domain-invariant information and a style code $s_i \in \mathcal{S}_i$ that encodes domain-dependent information. $\mathcal{C}$ is shared across domains and contains the information we want to preserve. $\mathcal{S}_i$ is domain-specific and contains the information we want to change. In conversion stage, we extract the content code of x_i and recombine it with a style code randomly sampled from $\mathcal{S}_j$. A generative adversarial network (GAN) [15] is added to ensure that the converted samples are indistinguishable from the real ones.

Figure 2 shows the autoencoder model of style transfer with a partially shared latent space. Any pair of corresponding datapoints (x_i, x_j) is assumed to have a shared latent code $c \in \mathcal{C}$ and domain-specific style codes $s_i \in \mathcal{S}_i$, $s_j \in \mathcal{S}_j$. The generative model of domain i is an autoencoder that consists of a deterministic decoder $x_i = G_i(c_i, s_i)$ and two encoders $c_i = E_i^c(x_i)$, $s_i = E_i^s(x_i)$. The encoder $E_i = (E_i^c, E_i^s)$ and decoder G_i are inverse operations such that $E_i = G_i^{-1}$. To generate converted sample $x'_{j\leftarrow i}$, we just extract and recombine the content code of x_i with the style code of domain j.

$$\begin{aligned} x'_{i\leftarrow j} &= G_i(c_j, s_i) = G_i(E_j^c(x_j), s_i) \\ x'_{j\leftarrow i} &= G_j(c_i, s_j) = G_j(E_i^c(x_i), s_j) \end{aligned} \tag{2}$$

It should be noted that the style code s_i is not inferred from one example, but learnt from the entire target domain j, which is a major difference from [32]. This is because the style extracted from a single example is ambiguous and may not capture the general characteristics of the target domain. To restrict the mapping $p(x_j|x_i)$, we use an constriant that is slightly different from the cycle consistency [10]. It assumes that an example converted to another domain and converted back should be unchanged, i.e., $x''_{i\leftarrow j\leftarrow i} = x_i$. Instead, we apply a semi-cycle consistency in the latent space by assuming that only the latent codes remain unchanged $E_i^c(x'_{i\leftarrow j}) = c_i$ and $E_i^s(x'_{i\leftarrow j}) = s_i$. The relaxed constriant provides diversity to the generated samples.

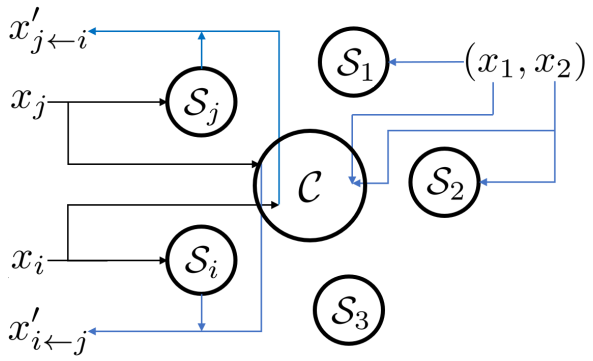

Figure 2: Autoencoder model with partially shared latent space. Sample x_i is encoded into a domain-specific space $\mathcal{S}_i$ and a shared content space $\mathcal{C}$. Corresponding datapoints (x_1, x_2) are encoded in the same content code. Converted sample $x'_{j\leftarrow i}$ is generated by recombining the original content code of x_i and the style code randomly sampled from $\mathcal{S}_j$.

We formulate the learning problem as a distributionally robust game. Each domain i has four agents E_i^c, E_i^s, G_i, D_i, in which D_i is a discriminator with two objectives. One is to distinguish between real samples and machine-generated samples, the other is to classify domain labels. On the other side, the generators G_i have two purposes: synthesize realistic samples and convert them into domain i. For example in voice conversion, the synthesized speech is evaluated on both the naturalness of its quality and the correctness of speaker's identity.

5.4.1 Two-domain transformation

There are $4n$ agents for n domains. For simplicity, we first investigate the conversion model between two domains. When $n = 2$, there are 8 agents: E_1^c, E_1^s, G_1, D_1 for domain $\mathbb{X}_1$ and E_2^c, E_2^s, G_2, D_2 for domain $\mathbb{X}_2$. Each agent has a different objective, and the utility function of one agent depends on the action of other agents. Collaboration and competition exist among them. So this is a distributionally robust game with cooperative agents and uncertain utility functions. Based on the analysis above, there are five modules need to learn:

① domain-invariant content encoder

② domain-specific style encoders

③ real/fake discriminator

④ domain classifier D_i

⑤ fake samples generator

The encoders and decoders form a group to synthesize converted samples in the target domain. Another group is the discriminator and classifier. They work on the opposite side to distinguish between real/fake samples and predict the domain label. The intergroup competition and intragroup collaboration are listed in table 2.

How to define the game?
When Nash equilibrium reaches, the autoencoder $(E_i^{c*}, E_i^{s*}, G_i^*)$ minimizes the reconstruction error $L_{rec}^x \to 0, L_{rec}^c \to 0, L_{rec}^s \to 0$. The GAN network (D_i^*, G_i^*) and adversarial loss $L_{GAN}^{x_i}$ converge at saddle points that minimize the distance between $p(x_i)$ and $p(x'_{j\leftarrow i})$. The classifier D_i^{cls*} correctly predicts the domain category of both real and fake samples $D_i^{cls*}(x_i) = i$, $D_i^{cls*}(x'_{i\leftarrow j}) = i$.

Table 2: Cooperative game

Intergroup competition	Learning module	Objective
E_1^s, E_2^s	②	$\min L_{cyc}^s$
D_1, D_2	④	$\min L_{cls}^x$
Intragroup collaboration	**Learning module**	**Objective**
E_1^c, E_1^s, G_1	$Autoencoder_1$	$\min L_{rec}^{x_1}$
E_2^c, E_2^s, G_2	$Autoencoder_2$	$\min L_{rec}^{x_2}$
G_1, G_2	⑤	$\min L_{GAN}^x$
D_1, D_2	③	$\max L_{GAN}^x$
E_1^c, E_2^c	①	$\min L_{cyc}^c$

The collaboration means, agents in different domains can help each other as they may have common interests. For instance, a sample can be used to update the real/fake discriminator even if its class label is missing. Except for the autoencoders E_i^c, E_i^s, G_i, other coalitions are cross-domain. (G_1, G_2) works on data synthesis, D_1, D_2 works on real/fake discrimination, E_1^c, E_2^c extracts high-level content information that we want to preserve.

We jointly train the encoders, decoders and GAN's discriminators with multiple objectives. To keep encoder and decoder as inverse operations, a reconstruction loss is applied in the direction $x_i \to (c_i, s_i) \to x'_i$, $(i, j \in 1, 2)$. Sample x_i should not be changed after encoding and decoding.

$$L_{rec}^{x_i} = \mathbb{E}_{x_i}(\|x_i - x'_i\|_1), \quad x'_i = G_i(E_i^c(x_i), E_i^s(x_i)) \tag{3}$$

In our model, the latent space is partially shared. Thus the cycle consistency constraint [10] is not preserved, i.e., $x''_{1\leftarrow 2\leftarrow 1} \neq x_1$. We apply a semi-cycle loss in the coding direction $c_1 \to x'_{2\leftarrow 1} \to c'_{2\leftarrow 1}$ and $s_2 \to x'_{2\leftarrow 1} \to s'_{2\leftarrow 1}$.

$$\begin{aligned} L_{cyc}^{c_1} &= \mathbb{E}_{c_1,s_2}(\|c_1 - c'_{2\leftarrow 1}\|_1), \quad c'_{2\leftarrow 1} = E_2^c(x'_{2\leftarrow 1}) \\ L_{cyc}^{s_2} &= \mathbb{E}_{c_1,s_2}(\|s_2 - s'_{2\leftarrow 1}\|_1), \quad s'_{2\leftarrow 1} = E_2^s(x'_{2\leftarrow 1}) \end{aligned} \tag{4}$$

Moreover, we add a GAN module to ensure the quality of generated samples. They should be indistinguishable from the real samples in the target domain $\mathbb{X}_i$. GAN loss is computed between x_j and $x'_{i\leftarrow j}$ to represent the distance between two distributions $p(x_j), p(x'_{i\leftarrow j})$.

$$L_{GAN}^{x_i} = \mathbb{E}_{c_j,s_i}[\log(1 - D_i(x'_{i\leftarrow j}))] + \mathbb{E}_{x_i}[\log D_i(x_i)] \tag{5}$$

The full loss is the weighted sum of $L_{recon}, L_{cycle}, L_{GAN}$.

$$\begin{aligned} &\min_{E_1^c, E_1^s, E_2^c, E_2^s, G_1, G_2} \max_{D_1, D_2} L(E_1^c, E_1^s, E_2^c, E_2^s, G_1, G_2, D_1, D_2) \\ &= \lambda_s(L_{cyc}^{s_1} + L_{cyc}^{s_2}) + \lambda_c(L_{cyc}^{c_1} + L_{cyc}^{c_2}) + \lambda_x(L_{rec}^{x_1} + L_{rec}^{x_2}) + \lambda_g(L_{GAN}^{x_1} + L_{GAN}^{x_2}) \end{aligned} \tag{6}$$

where $\lambda_s, \lambda_c, \lambda_x, \lambda_g$ control the weights of the components.

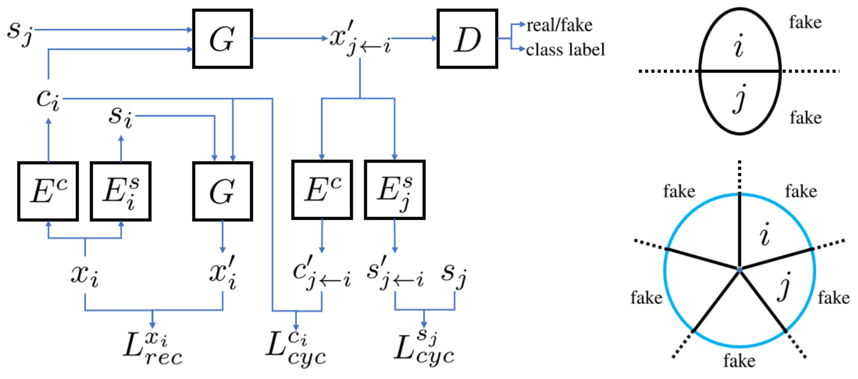

Figure 3: Learn multi-domain transformation. Left: conversion model with shared content encoder, decoder and classifier. Top-right: two domains. Bottom-right: multiple domains (n=5).

5.4.2 Multi-domain transformation

In multi-domain case, there are $4n$ agents in the game. To reduce complexity, we replace the domain-specific models E^c_i, G_i, D_i with a shared content encoder E^c, a shared decoder G, and a single multiclass classifier D^{cls}. Thus, only $n + 3$ agents left. (form coalition?) Figure 3 shows the multi-domain transformation model.

When $n = 2$, there are 3 kinds of data: real in $\mathbb{X}_1$, real in $\mathbb{X}_2$ and fake in $\mathbb{X}_{fake}$. Two real/fake discriminators D_1, D_2 are enough for classification. If this idea is extended to multiple domains, there will be n binary discriminators and 2^n outputs. Instead, we replace them with one binary real/fake discriminator D and one multiclass domain classifier D^{cls}. The two-step classification not only reduces complexity ($2n$ outputs) but also makes the most of the training data. An example in domain i is also useful to train the generative model for other domains, as it is the common interest of all agents $G_1, G_2, \ldots G_n$ to synthesize realistic data. Therefore, the multi-domain transformation model can be trained on different datasets with partially labeled data.

In the following sections, we will apply the proposed method on two real-world applications: image style transfer and emotinal speech conversion.

5.5 Case Study: Image Style Transfer

5.6 Case Study: Emotional Voice Conversion

Voice transformation (VT) is a technique to modify some properties of human speech while preserving its linguistic information. VT can be applied to change the speaker identity, i.e., voice conversion (VC) [50], or to transform the speaking style of a speaker, such as emotion and accent conversion [65]. In this section, we will focus on emotion voice transformation. The goal is to change emotion-related characteristics of a speech signal while preserving its linguistic content and speaker identity. Emotion conversion techniques can be applied to various tasks, such as hiding negative emotions for customer service agents, helping film dubbing, and creating more expressive voice messages on social media.

6 Dissertation Outline

The research will be split into the following four stages:

6.1 Introduction

6.2 Related Work

6.3 Distributionally Robust Games

In this part we introduce distributionally robust games and develop new filtering and learning architectures under this framework. The system may contain several competing neural networks: the attackers learn to generate synthetic samples that are supposed to have the same distribution as the original ones, while the defenders try to find counter-examples and create difficulties for the other side. Each player tries to perform better and beat the others, which forms a multi-agent zero-sum game with uncertain payoffs. The players use a robust optimization approach to contend with the worst-case scenario payoff. The attacker network is constructed based on the outcome of defender networks, and vice versa. The competing networks are trained together iteratively until achieving the distributional robust Nash equilibrium.

6.4 Wasserstein Metric

The loss function is designed to measure the similarity of two probability distributions. Unsupervised learning is conducted by minimizing the loss. We plan to study the properties of several widely used loss metrics:

- Compare L1, L2-loss, KL-divergence, f-divergence, and Wasserstein distance
- Study the time-dependent formulation of the optimal transportation cost
- Test the effect of translation and perturbation for a certain loss metric

6.5 Learning Algorithms for Robust Optimization

- Develop a specific learning algorithm to find robust Nash equilibria, which should be stable and efficient
- Compare with existing numerical optimization approaches in large-scale machine learning: SGD, Adam, Momentum, Ishikawa-Nesterov, Newton's method, conjugate gradient, natural gradient, etc.
- Compare with existing deep generative models: RBM, VAE, GAN, WGAN, etc.

6.6 Generative Modeling for Vehicle Tracking

6.7 Generative Modeling for Image Synthesis

6.8 Generative Modeling for Voice Conversion

6.9 Experiments on Large-scale Machine Learning datasets

- Maryland Traffic Surveillance Dataset (Vehicle tracking)
- Large-scale CelebFaces Attributes Dataset (CelebA, image synthesis)
- Large-scale Scene Understanding Challenge (LSUN, image synthesis)
- Interactive Emotional Dyadic Motion Capture (IEMOCAP, emotional voice conversion)

6.10 Discussion

6.11 Conclusion and Future Work

7 Research Plan

7.1 Research Progress

Literature review, planning

Theory part on distributional robust games, Bregman learning and convex optimization

Theoretical analysis and comparison for L2 distance, f-divergence and Wasserstein metric

Algorithm design, overall integration, simulations, specific implementations on real problems

Application part on large-scale machine learning: experiments, evaluation and revision

Documentation and Defence

7.2 Application on Image and Audio Synthesis

- Test on large-scale image dataset MNIST, CelebA and LSUN
- Literature review on emotional speech classification and audio synthesis
- Compare two sound representations in generative learning: waveform and spectrogram
- Design deep generative models for emotional speech generation
- Test on voice conversion or music style transfer if possible

7.3 Timeline

Fall 2018	Study on Generative Models for Voice Conversion
Spring 2019	Write Thesis
August 2019	Thesis Defense

8 Conclusion

Tackling the aforementioned problems would take us much closer to real intelligent systems, and defines three core pillars of Artificial Intelligence. However, there are many other problems which need to be solved and integrated to achieve a fully intelligent system, e.g. navigation, learning by imitation, cooperation, and many others.

9 List of Publications

9.1 Thesis Related Publications

Jian Gao and Hamidou Tembine, Distributionally Robust Games for Deep Generative Learning, July 2018. DOI: 10.13140/RG.2.2.15305.44644

Jian Gao, Yida Xu, Julian Barreiro-Gomez, Massa Ndong, Michalis Smyrnakis and Hamidou Tembine (September 5th 2018) Distributionally Robust Optimization. In Jan Valdman, Optimization Algorithms, IntechOpen. DOI: 10.5772/intechopen.76686. ISBN: 978-1-78923-677-4

Jian Gao and Hamidou Tembine, Distributionally Robust Games: Wasserstein Metric, International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, July 2018

Jian Gao and Hamidou Tembine, Bregman Learning for Generative Adversarial Networks, Chinese Control and Decision Conference (CCDC), Shenyang, China, June 2018 *(Best Paper Finalist Award)*

Jian Gao and Hamidou Tembine, Distributed Mean-Field-Type Filter for Vehicle Tracking, in American Control Conference (ACC), Seattle, USA, May 2017 *(Student Travel Award)*

Dario Bauso, Jian Gao and Hamidou Tembine, Distributionally Robust Games: f-Divergence and Learning, 11th EAI International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS), Venice, Italy, Dec 2017

9.2 Other Publications

J. Gao and H. Tembine, "Distributed Mean-Field-Type Filters for Traffic Networks," in IEEE Transactions on Intelligent Transportation Systems. doi: 10.1109/TITS.2018.2816811

J. Gao and H. Tembine, "Empathy and berge equilibria in the forwarding dilemma in relay-enabled networks," 2017 International Conference on Wireless Networks and Mobile Communications (WINCOM), Rabat, 2017, pp. 1-8. doi: 10.1109/WINCOM.2017.8238199 (*Best paper Award*)

J. Gao and H. Tembine, "Correlative mean-field filter for sequential and spatial data processing," IEEE EUROCON 2017 -17th International Conference on Smart Technologies, Ohrid, 2017, pp. 243-248. doi: 10.1109/EUROCON.2017.8011113

Fanhuai Shi, Jian Gao, Xixia Huang, An affine invariant approach for dense wide baseline image matching. International Journal of Distributed Sensor Networks (IJDSN) 12(12) (2016)

J. Gao and H. Tembine, "Distributed Mean-Field-Type Filters for Big Data Assimilation," 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Sydney, NSW, 2016, pp. 1446-1453. doi: 10.1109/HPCC-SmartCity-DSS.2016.0206

[1]

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.