# Game-Theoretic Models for Generative Learning

**PhD Thesis Proposal**


## Abstract

Machine learning has achieved great success in object recognition and classification problems based on the rapid development of 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 class labels given data samples, while generative learning tries to estimate the full data distribution and synthesizes new samples. It can control the generated samples by changing its class domain or modifing its visual appearance. 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 metrics that measure the similarity between distributions, which is a major issue in many generative learning problems. We then propose a conditional generative model to sovle the style transfer problem in image and speech processing. The disentangled representations of domain-specific style information and domain-invariant content information are modeled by autoencoders and domain classifiers. The encoders, decoders and classifiers form a distributinally robust game with competitive and collaborative agent coalitions. Finally, we plan to test our approach on real-world applications including conditional image synthesis and emotional speech conversion.


## 1 Introduction

In the last decade, machine learning has achieved great success with the rapid development of deep neural networks. Recent algorithms beat humans on ImageNet Challenge [79], the largest benchmark for image recognition. On the other side, people use computers to mimic the creative power of humans. A stream of papers [75][76][77][78][19] claimed their ability to generate lifelike pictures, text and music that can fool the human evaluators as well as machine classifiers. I identify three levels of artificial intelligence: memorization, recognition and creativity. Computers are approaching humans on the first two levels, and now going to the third.

Apart from object recognition and classification, people want to learn the mechanism of data generation and synthesize new samples with desired properties. Discriminative learning tries to infer knowledge from data, while generative learning attempts to learn the full distribution of data and generate new samples. My research focuses on the latter problem.

In probability theory, discriminative models make predictions by learning a conditional distribution $p(y \mid x)$, and generative models synthesize new data by drawing samples from $p(x)$. Both distributions are estimated from a limited set of observations, which could be noisy and imcomplete. The former problem is easier since $y$ is ususally in the low dimension space. Sometimes people only use a small portion of the data and ignore the other parts. For example in support vector machines (SVM) only points near the decision surface have influence on the classifier. The latter problem is harder because generative models need to estimate the full data distribution $p(x)$. Modeling the high-dimensional random variable $x$ is difficult. As the number of configurations can grow exponentially with the number of dimensions, there are not enough training examples for each dimension. Another

concern is the computation challenge; many algorithms involve operations that grow exponentially with the number of dimensions.

Generative models are widely used in unsupervised learning. A major limitation of the current learning algorithms is that they rely on large amounts of well-labelled data to achieve good accuracy. However, it is not available in many industrial applications. For this reason, people adapt a pretrained model on a source dataset to a similar target dataset. The model may perform pooly as it is specialized to the source domain. Domain adaptation strategies can solve this problem. Let $x_1 \in X_1$ be source domain data with associated labels $y_1$, $x_2 \in X_2$ be target domain data with unknown labels $f(x_2)$. A mapping from the source domain to the target domain can be established by the conditional generative model $p(x_2 \mid x_1)$. The classifier $f$ in the target domain is learnt by the category knowledge from the source domain.

Apart from traditional generative models like Gaussian mixture model (GMM) and Naive Bayes, neural network models exhibit increaing importance in modeling high dimension data. It is amazing if machines can generate artistic work in painting, music and sculpture. There are three kinds of dominant approaches in deep generative learning: generative adversarial networks (GAN), variational auto-encoders (VAE) and autoregressive models. In general, training deep generative models is hard and time consuming. The high dimensional training data and complex objective structures lead to many problems in optimization, such as instability, saturation, and mode collapse. Moreover, some models fail to provide enough diversity in the generated examples or just memorize the training set.

In this research, we plan to develop a new game-theoretic framework for generative learning. We first address the problem of data synthesis. The goal is to produce lifelike artificial examples that are indistinguishable from the training data. This involves the problem of computing the statistical distance between two datasets: the original real sample set and the generated fake sample set. Several distance metrics will be investigated to measure the similarity between distributions. As an important part of this research, we will compare Wasserstein distance [1] with the most prevalent information-based metrics such as Kullback-Leibler [80] and Jensen-Shannon divergence [81] in generative learning. We plan to develop a practical method to approximately calculate the distance between distributions. Both theoretical analysis and experimental simulations will be provided.

We then propose a conditional generative model to solve the unsupervised style transfer problem in image and speech processing. The objective is to learn a translation model between domains that can modify the domain-specific style features and preserve the domain-invariant content information. The disentangled representations of style and content are modeled by autoencoders and domain classifiers.

We formulate the problem as a distributionally robust game with payoff uncertainty. The encoders, decoders and classifiers form competitive and collaborative agent coalitions in this game. They optimize towards their self objectives as well as the common interst with other players. Agents with similar objectives form a group and work together against the others in order to optimize their expected payoffs. The optimization process is not deterministic since the payoff function of each player depends on the actions of other players. The distributionally robust Nash equilibrium is achieved by solving a minimax optimization problem, in which each agent tries to maximize its worst-case payoff. An iterative learning algorithm is introduced to solve the robust Nash equilibrium.

We will first verify the theoretical results through simulations, and then apply our approach on real datasets of images and speech. The agents are implemented with deep convolution neural networks to capture the spatial and temporal correlations in high dimension data. We plan to work on several tasks such as object detection, vehicle tracking, image synthesis and voice conversion. This research contributes to the areas of distributionally robust game, deep generative learning, stochastic optimization and time-series data analysis.

The following chapters first describes the already completed work: the theory part of distributionally robust games (Chapter 2), the investigation of distance metrics (Chapter 3), learning algorithms and applications on image synthesis (Chapter 4). After that the progress on the current work of style transfer is described in Chapter 5. Finally, the thesis outline and research timeline are given in Chapter 6.

# 2 Distributionally Robust Games

One fundamental problem in generative learning is to estimate the full data distribution given finite noisy observations. Suppose the training samples $x$ are drawn from a distribution $m$, and the generative model produces new data $\tilde{x}$ by sampling from an estimated distribution $\tilde{m}$. The objective is to minimize the statistical distance between $m$ and $\tilde{m}$ so that the synthesized data is similar to the real ones. Since the comparison is made on two statistical objects instead of two individual sample points, there is no deterministic objective to use. We formulate the problem as a distributionally robust game (DRG) with payoff uncertainty, and offer a corresponding distributionally robust equilibrium concept.

In this chapter, we first introduce the definition of distribution uncertainty set and formulate the distributionally robust game as a minimax optimization problem. Then, we define the distributionally robust equilibrium, analysis the complexity of the problem, and use triality theory to reduce model dimensionality via Lagrangian relaxation. Finally, we apply this model to solve an unsupervised generative learning problem.

## 2.1 Introduction

Games with payoff uncertainty refer to games where the outcome of a play is not known with certainty by the players. Such games are also called incomplete information games and can be formalized in different ways. Distribution-free models of incomplete information games, both with and without private information are examined in [2, 3]. There the players use a robust optimization approach to contend with the worst-case scenario payoff. The distribution free approach relaxes the well-known Bayesian game model approach by Harsanyi. The limitation of the distribution-free model is that the uncertainty set has to be carefully designed and in most cases such approach leads to too conservative and unrealistic scenarios. We propose a distribution-dependent model in which the uncertainty set contains all possible distributions within a bounded region.

Distributionally robust game (DRG) is a game with incomplete information. Instead of assuming an underlying mean-field or exactly known probability distribution, one acts with an uncertainty set, which could be distributions chosen by other players. The set of distributions should be chosen to fit for the applications at hand. In robust best-response problems, the uncertain sets are represented by deterministic models. The opponent players have a bounded capability to change the uncertain parameters, and therefore affects the objective function the decision maker seeks to optimize. Each player has his own robust best response optimization problem to solve. Thus, the standard best response problem of agent $j$: $\inf_{a_j \in \mathcal{A}_j} l_j(a_j, a_{-j}, \omega)$ becomes the minimax robust best response:

$$\inf_{a_j \in \mathcal{A}_j} \sup_{\omega \in \Omega} l_j(a_j, a_{-j}, \omega) \tag{1}$$

where $a_j$ is the action to choose and $l$ is the objective functional evaluated at uncertain state $\omega$. This approach to uncertainty has a long history in optimization, control and games [4, 5, 6]. A credible alternative to this set-based uncertainty is to use a stochastic model, in which the uncertain state $\omega$ is a random variable with distribution $m$. If we assume the generating mean-field distribution $m$ is known, it becomes the standard stochastic optimal control paradigm. If $m$ is not known and the only known is a set of distributions lie in the neighborhood of $m$: $\tilde{m} \in B_\rho(m)$, the resulting best response to mean-field formulation is the so-called distributionally robust best response:

$$\inf_{a_j \in \mathcal{A}_j} \sup_{\tilde{m} \in B_\rho(m)} \mathbb{E}_{\omega \sim \tilde{m}} l_j(a_j, a_{-j}, \omega) \tag{2}$$

### 2.1.1 Distribution Uncertainty Set

Let $(\Omega, \mathcal{F}, m)$ be a probability space. Here $m$ is a probability measure defined on $(\Omega, \mathcal{F})$. The distribution $m$ of the state $\omega$ is used to capture the probability of the different scenarios and of the corresponding performance function obtained under each of such scenarios for fixed action profile. We assume that the exact distribution of the state is not available in general. Therefore we propose an uncertainty/constraint set among all possible distributions with a divergence bounded by above by a scalar $\rho$. When $f-$divergence [7, 8, 9] is applied, such a constraint takes the form

$$B_\rho(m) = \{\tilde{m} \mid \int_\Omega d\tilde{m} = \tilde{m}(\Omega) = 1,\ D_f(\tilde{m} \parallel m) \leq \rho\}, \tag{3}$$

When the Wasserstein distance [1] is applied, the uncertainty set contains probability distributions within a Wasserstein ball.

$$B_\rho(m) = \{\tilde{m} \mid \int_\Omega d\tilde{m} = \tilde{m}(\Omega) = 1,\ W(m, \tilde{m}) \leq \rho\}, \tag{4}$$

**$f-$Divergence** Let $m$ and $\tilde{m}$ be two probability measures over a space $\Omega$ such that $\tilde{m}$ is absolutely continuous with respect to $m$. Then, for a convex function $f$, the $f-$divergence of $\tilde{m}$ from $m$ is defined as

$$D_f(\tilde{m} \parallel m) \equiv \int_\Omega f\left(\frac{d\tilde{m}}{dm}\right) dm - f(1),$$

where $\frac{d\tilde{m}}{dm}$ is the Radon-Nikodym derivative of the measure $\tilde{m}$ with respect to the measure $m$.

Recall that for a proper convex function $f$, the Legendre-Fenchel duality holds: $(f^*)^* = f$ where

$$f^*(\xi) = \sup_x[\langle x, \xi\rangle - f(x)] = -\inf_x[f(x) - \langle x, \xi\rangle]. \tag{5}$$

### 2.1.2 Related Work

Strategic learning has proven to be a powerful approach in stochastic games. In particular, its algorithmic nature is well suited to accommodate parallel and distributed information exchange and processing as well as hardware realizability. However, almost all existing learning approaches work well only for specific classes of games such as concave-convex zero-sum games, convex potential games, and some S-modular game problems with unimodal objective functions. For more general classes of games, convergence of strategic learning dynamics is still an open issue. In addition, learning algorithms for finding fixed points or equilibria for general classes of games still present several challenges. For polymatrix games with finite action spaces, there have been great progress including Cournot adjustment, Brown-von Neumann-Nash dynamics, reinforcement learning, combined learning (see [10] and the references therein). For continuous action spaces, however, only few handful works are available. Evolutionary dynamics and revision protocols based actions has been proposed [11, 12, 13]. The exploration of continuous action takes too much time if the dynamics is based on individual action or measurable subsets [14, 15, 16]. Moreover, the convergence time of these existing strategic learning algorithms (when a convergence to a point or a limit cycle occurs) are unacceptably high even for potential games and it often requires strong assumptions such as bounded densities. The above mentioned prior works do not consider robust games setting. In [2, 3] a robust game framework is presented. The authors defined a distributed-free approach by considering the worst-case performance. However the choice of the uncertainty set remains an important part of the robust game modelling [17]. In this chapter we are interested in learning in distributionally robust games under $f$-divergence [7, 8, 9, 18].

## 2.2 Problem Formulation

In distributionally robust games, each agent $j$ adjusts his action $a_j \in \mathcal{A}_j$ to optimize the worst-case payoff functional $\mathbb{E}_{\tilde{m}} l_j(a, \omega)$ subject to the constraint that the divergence $D_f(\tilde{m} \parallel m) \leq \rho$. This means that the worst-case performance is obtained under the assumption that a virtual player (nature) acts as a defender who modifies the distribution $m$ into $\tilde{m}$ with an effort capacity that should not exceed $\rho > 0$.

Throughout the paper we assume that the following conditions hold. The measure $\tilde{m}$ is continuous with respect to $m$; it is not a given profile, and it could be deformed or falsified by the defender. The payoff function $l_j(\cdot, \omega)$ is proper and upper semi-continuous for $m-$almost all $\omega \in \Omega$. Either the domain $\mathcal{A}_j$ is a non-empty compact set or $\mathbb{E}_{\omega\sim\tilde{m}} l_j(a, \omega)$ is coercive.

**Definition 1** (Distributionally Robust Game)**.** *The robust game $\mathcal{G}(m)$ involves*

- *The set of agents $\mathcal{J} = \{1, 2, \ldots, n\},\ n \geq 2$*
- *The action profile of each agent $j$, $\mathcal{A}_j$, $j \in \mathcal{J}$*
- *The uncertainty set of probability distributions defined on $m$, $\Omega$ and $\rho > 0$: $B_\rho(m)$*
- *The expected payoff function of agent $j$: $\mathbb{E}_{\omega\sim\tilde{m}} l_j(a, \omega)$, where $\tilde{m}$ is an alternative distribution of $m$ within some bounded distance*

Then the robust stochastic optimization of agent $j$ given the uncertain set and the action of other agents is formulated as a minimax problem

$$(P_j): \ \inf_{a_j \in \mathcal{A}_j} \ \sup_{\tilde{m} \in B_\rho(m)} \ \mathbb{E}_{\omega \sim \tilde{m}} l_j(a, \omega). \tag{6}$$

With the above game in mind, we introduce the following solution concept for game $\mathcal{G}(m)$.

**Definition 2** (Distributionally Robust Nash Equilibrium)**.** *Let $a_j^*$ be the configuration of agent $j$ and $a_{-j}^* := (a_k^*)_{k \neq j}$. A strategy profile $a^* = (a_1^*, \ldots, a_n^*)$ satisfying*

$$\sup_{\tilde{m} \in B_\rho(m)} \mathbb{E}_{\tilde{m}} l_j(a^*, \omega) \leq \sup_{\tilde{m} \in B_\rho(m)} \mathbb{E}_{\tilde{m}} l_j(a_j, a_{-j}^*, \omega)$$

*for every $a_j \in \mathcal{A}_j$ and every agent $j \in \mathcal{J}$, is said to be the* distributionally robust pure Nash equilibrium *of game $\mathcal{G}(m)$.*

**Existence of distributionally robust Nash equilibria** In other words, reaching the robust Nash equilibrium means all agents achieve the optimal payoff in their worst-case scenario. As in classical game theory, sufficient condition for existence of robust equilibrium can be obtained from the standard fixed-point theory: if $\mathcal{A}_j$ are nonempty compact convex sets and $l_j$ are continuous functions such that for any fixed $a_{-j}$, the function $a_j \mapsto l_j(a, \omega)$ is quasi-convex for each $j$, then there exists at least one distributionally robust *pure* Nash equilibrium. This result can be easily extended to the coupled-action constraint case for generalized robust Nash equilibria.

## 2.3 Minimax Robust Game

### 2.3.1 Infinite Dimension

Assume that $a \mapsto \mathbb{E}_{\tilde{m}} l_j(a, \omega)$ is continuous for $m-$almost all $\omega$. Then, the functional $F_j: \ \tilde{m} \mapsto \inf_{a_j} \mathbb{E}_{\tilde{m}} l_j(a, \omega)$ is Gateaux differentiable with derivative

$$F_{j,m}(\hat{m}) = \inf_{a_j \in \mathcal{A}_j^*(m)} \mathbb{E}_{\hat{m}} l_j(a, \omega),$$

where $\mathcal{A}_j^*(m) = \arg\min_{a_j} \mathbb{E}_m l_j(a, \omega)$ is the best-response under $m$. This derivative in the space of square integrable measurable functions under $m$ is of infinite dimensions, which does not facilitate the computation of the robust optimal strategy $a_j^*, \tilde{m}^*$. Below we propose an equivalent problem that reduces considerably the curse of dimensionality of the problem.

### 2.3.2 Triality Theory

We here streamline the basic idea of triality theory. For this purpose, consider uncoupled domains $\mathcal{A}_j, \ j \in \mathcal{J}$. For a general real-valued loss function $l_2$ on $\mathcal{A}_1 \times \mathcal{A}_2$, one has

$$\sup_{a_2 \in \mathcal{A}_2} \inf_{a_1 \in \mathcal{A}_1} \ l_2(a_1, a_2) \leq \inf_{a_1 \in \mathcal{A}_1} \sup_{a_2 \in \mathcal{A}_2} \ l_2(a_1, a_2)$$

and the difference is called the well-known duality gap.

$$\min_{a_1 \in \mathcal{A}_1} \max_{a_2 \in \mathcal{A}_2} \ l_2(a_1, a_2) - \max_{a_2 \in \mathcal{A}_2} \min_{a_1 \in \mathcal{A}_1} \ l_2(a_1, a_2)$$

It is widely known in duality theory from Sion's minimax Theorem [5] (which is an extension of von Neumann minimax Theorem) there is an equality, for example for convex-concave function, and the value is achieved by a saddle point in the case of non-empty convex compact domain. Similarly, for a general loss function $l_3$: $(a_1, a_2, a_3) \mapsto l_3(a_1, a_2, a_3)$, one has

$$\begin{aligned}
\inf_{a_3 \in \mathcal{A}_3} \sup_{a_2 \in \mathcal{A}_2} \inf_{a_1 \in \mathcal{A}_1} \ l_3(a_1, a_2, a_3) &\leq \inf_{a_1 \in \mathcal{A}_1, a_3 \in \mathcal{A}_3} \sup_{a_2 \in \mathcal{A}_2} \ l_3(a_1, a_2, a_3), \\
\sup_{a_3 \in \mathcal{A}_3} \inf_{a_2 \in \mathcal{A}_2} \sup_{a_1 \in \mathcal{A}_1} \ l_3(a_1, a_2, a_3) &\geq \sup_{a_1 \in \mathcal{A}_1, a_3 \in \mathcal{A}_3} \inf_{a_2 \in \mathcal{A}_2} \ l_3(a_1, a_2, a_3).
\end{aligned}$$

**Proposition 1** (Triality)**.** *Let $(a_1, a_2, a_3) \mapsto l_3(a_1, a_2, a_3) \in \mathbb{R}$ be a loss function $l_3$ defined on $\prod_{i=1}^3 \mathcal{A}_i$. Then, the following inequalities hold:*

$$\begin{array}{ll}
& \sup_{a_2 \in \mathcal{A}_2} \inf_{a_1 \in \mathcal{A}_1, a_3 \in \mathcal{A}_3} l_3(a_1, a_2, a_3) \\
\leq & \inf_{a_3 \in \mathcal{A}_3} \sup_{a_2 \in \mathcal{A}_2} \inf_{a_1 \in \mathcal{A}_1} l_3(a_1, a_2, a_3) \\
\leq & \inf_{a_1 \in \mathcal{A}_1, a_3 \in \mathcal{A}_3} \sup_{a_2 \in \mathcal{A}_2} l_3(a_1, a_2, a_3),
\end{array} \tag{7}$$

*and similarly*

$$\begin{array}{ll} & \sup_{a_1\in\mathcal{A}_1,a_3\in\mathcal{A}_3}\inf_{a_2\in\mathcal{A}_2} l_3(a_1,a_2,a_3) \\ \leq & \sup_{a_3\in\mathcal{A}_3}\inf_{a_2\in\mathcal{A}_2}\sup_{a_1\in\mathcal{A}_1} l_3(a_1,a_2,a_3) \\ \leq & \inf_{a_2\in\mathcal{A}_2}\sup_{a_1\in\mathcal{A}_1,a_3\in\mathcal{A}_3} l_3(a_1,a_2,a_3). \end{array} \tag{8}$$

*Proof.* First we shall prove the $\sup\inf$ inequality. Define

$$g(a_2,a_3)=\inf_{a_1\in\mathcal{A}_1} l_3(a_1,a_2,a_3).$$

Thus, for all $a_2,a_3$ one has $g(a_2,a_3)\leq l_3(a_1,a_2,a_3)$. It follows that, for any $a_1,a_3$,

$$\sup_{a_2\in\mathcal{A}_2} g(a_2,a_3)\leq \sup_{a_2\in\mathcal{A}_2} l_3(a_1,a_2,a_3).$$

Using the definition of $g$, one obtains

$$\sup_{a_2\in\mathcal{A}_2}\inf_{a_1\in\mathcal{A}_1} l_3(a_1,a_2,a_3)\leq \sup_{a_2\in\mathcal{A}_2} l_3(a_1,a_2,a_3),\ \forall a_1,a_3.$$

Taking the infinimum in $a_1$ yields:

$$\sup_{a_2\in\mathcal{A}_2}\inf_{a_1\in\mathcal{A}_1} l_3(a_1,a_2,a_3)\leq \inf_{a_1\in\mathcal{A}_1}\sup_{a_2\in\mathcal{A}_2} l_3(a_1,a_2,a_3),\ \forall a_3 \tag{9}$$

Now, for variable in $a_3$ we use two operations:

1. Taking the infininum in inequality (9) in $a_3$ yields

$$\begin{aligned}\inf_{a_3\in\mathcal{A}_3}\sup_{a_2\in\mathcal{A}_2}\inf_{a_1\in\mathcal{A}_1} l_3(a_1,a_2,a_3) &\leq \inf_{a_3\in\mathcal{A}_3}\inf_{a_1\in\mathcal{A}_1}\sup_{a_2\in\mathcal{A}_2} l_3(a_1,a_2,a_3)\\ &= \inf_{(a_1,a_3)\in\mathcal{A}_1\times\mathcal{A}_3}\sup_{a_2\in\mathcal{A}_2} l_3(a_1,a_2,a_3),\end{aligned} \tag{10}$$

   which proves the second part of the inequalities (7). The first part of the inequalities (7) follows immediately from (9).

2. Taking the supremum in inequality (9) in $a_3$ yields

$$\sup_{(a_2,a_3)\in\mathcal{A}_2\times\mathcal{A}_3}\inf_{a_1\in\mathcal{A}_1} l_3(a_1,a_2,a_3)\leq \sup_{a_3\in\mathcal{A}_3}\inf_{a_1\in\mathcal{A}_1}\sup_{a_2\in\mathcal{A}_2} l_3(a_1,a_2,a_3),$$

   which proves the first part of the inequalities (8). The second part of the inequalities (8) follows immediately from (9).

This completes the proof. □

We use the above inequalities in the Lagrangean relaxation of the minimax robust game.

### 2.3.3 Dimension Reduction

In order to reduce the curse of dimensionality of the problem, we use the triality theory and convert the original infinite dimensional problem $(P_j)$ into a finite dimensional problem.

**Proposition 2.** *The (distributionally) robust best-response problem of player $j$ is equivalent to the finite dimensional stochastic optimization problem when $\mathcal{A}_j$ are of finite dimensions:*

$$(P_j^*)\begin{cases}\inf_{a_j\in\mathcal{A}_j,\lambda\geq 0,\mu\in\mathbb{R}} l_j^*(a,\lambda,\mu,m)\\ l_j^*(a,\lambda,\mu,m)=\lambda(\rho+f(1))+\mu+\int\lambda f^*(\frac{l_j-\mu}{\lambda})dm=\mathbb{E}_m h_i,\end{cases} \tag{11}$$

*where $h_i$ is the integrand cost $\lambda(\rho+f(1))+\mu+\lambda f^*(\frac{l_i-\mu}{\lambda})$.*

*Proof.* The (distributionally) robust best-response problem of agent $j$ is equivalent to

$$\inf_{a_j}\sup_{L\in L_\rho(m)}\mathbb{E}_m[l_j L] \tag{12}$$

where $L(\omega) = \frac{d\tilde{m}}{dm}(\omega)$ is the likelihood, and set $L_\rho(m)$ is

$$L_\rho(m) = \{L \mid \int_\omega f(L(\omega))dm - f(1) \leq \rho, \ \int_\omega L(\omega)dm = 1\}.$$

We introduce the Lagrangian as

$$\begin{array}{ll} \tilde{l}_j(a, L, \lambda, \mu) & = \int_\omega l_j(a, \omega)L(\omega)dm + \lambda(\rho + f(1) - \int_\omega f(L(\omega))dm) \\ & +\mu(1 - \int_\omega L(\omega)dm), \end{array}$$

where $\lambda \geq 0$ and $\mu \in \mathbb{R}$. The problem solved by Player $j$ becomes

$$(\tilde{P}_j^*): \ \inf_{a_j} \sup_{L \in L_\rho(m)} \inf_{\lambda \geq 0, \mu \in \mathbb{R}} \tilde{l}_j(a, L, \lambda, \mu). \tag{13}$$

A full understanding of problem $(\tilde{P}_j^*)$ requires a triality theory (not a duality theory) whose main principles were streamlined in Section 2.3.2. The underlying idea is that one can use a transformation of the last two terms to derive a finite dimensional optimization problem. The Lagrangian $\tilde{l}_j$ of agent $j$ is clearly concave in $L$ and convex in $\lambda, \mu$, and is semi-continuous jointly. By the triality theory above, $\tilde{l}_j : (a, L, \lambda, \mu) \mapsto \tilde{l}_j(a, L, \lambda, \mu)$ satisfies the $\sup\inf$ inequality and has the following inequality:

$$\inf_{a_j} \sup_{L \in L_\rho(m)} \inf_{\lambda \geq 0, \mu \in \mathbb{R}} \tilde{l}_j(a, L, \lambda, \mu) \leq \inf_{a_j} \inf_{\lambda \geq 0, \mu \in \mathbb{R}} \sup_{L \in L_\rho(m)} \tilde{l}_j(a, L, \lambda, \mu).$$

In this case there is no gap in the second part of the optimization and the following equality holds:

$$\inf_{a_j} \sup_{L \in L_\rho(m)} \inf_{\lambda \geq 0, \mu \in \mathbb{R}} \tilde{l}_j(a, L, \lambda, \mu) = \inf_{a_j} \inf_{\lambda \geq 0, \mu \in \mathbb{R}} \sup_{L \in L_\rho(m)} \tilde{l}_j(a, L, \lambda, \mu).$$

The latter problem can be rewritten as

$$(\tilde{P}_j^*): \ \inf_{a_j \in \mathcal{A}_j, \lambda \geq 0, \mu \in \mathbb{R}} [\sup_{L \in L_\rho(m)} \tilde{l}_j(a, L, \lambda, \mu)]. \tag{14}$$

The Lagrangian function takes the form as

$$\tilde{l}_j = \lambda(\rho + f(1)) + \mu + \int \{L[l_j - \mu] - \lambda f(L)\}dm \tag{15}$$

It follows that

$$\sup_{L \in L_\rho(m)} \tilde{l}_j(a, L, \lambda, \mu) = \lambda(\rho + f(1)) + \mu + \sup_L \int \{L[l_j - \mu] - \lambda f(L)\}dm \tag{16}$$

Introducing the Fenchel-Legendre transform on $L$ and exchanging $\sup$ and $\int$, one gets

$$\sup_{L \in L_\rho(m)} \tilde{l}_j(a, L, \lambda, \mu) = \lambda(\rho + f(1)) + \mu + \int \lambda f^*(\frac{l_j - \mu}{\lambda})dm. \tag{17}$$

Since $\mathcal{A}_j \times \mathbb{R}_+ \times \mathbb{R}$ is a subset of a finite dimensional vector space, it follows that the robust best-response problem of agent $j$ is equivalent to the finite dimensional stochastic optimization problem:

$$(P_j^*) \begin{cases} \inf_{a_j \in \mathcal{A}_j, \lambda \geq 0, \mu \in \mathbb{R}} l_j^*(a, \lambda, \mu, m) \\ l_j^*(a, \lambda, \mu, m) = \mathbb{E}_m h_j. \end{cases} \tag{18}$$

where $h_j$ is the integrand cost $\lambda(\rho + f(1)) + \mu + \lambda f^*(\frac{l_j - \mu}{\lambda})$. □

We have converted the infinite dimensional problem $(P_j)$ into a finite dimensional problem $(P_j^*)$. The above calculations culminate in the following result:

**Proposition 3.** *If $a, \lambda^*(a), \mu^*(a)$ is a solution of $(P_j^*)$ then the optimal likelihood $L^*$ is such that $\int_\omega L^* dm = 1, \ f'(L^*) = \frac{l_j - \mu^*}{\lambda^*}$. This means that $a_j$ and $d\tilde{m}^* = L^* dm$ provide a solution of the original problem $(P_j)$.*

*Proof.* Let $\lambda^*(a), \mu^*(a)$ be solution to $(P_j^*)$ associated with the profile $a$, then the optimal likelihood $L^*$ is obtained by differentiating $f^*$ or by inverting the equation $f'(L^*) = \frac{l_j - \mu^*}{\lambda^*}$. As $\tilde{m}$ is a probability measure, using the definition of $L^*$ one gets

$$d\tilde{m}^*(\omega) = L^* dm(\omega).$$

It follows that $a_j^*, L^*$ solves the original problem $(P_j)$. □

### 2.4 Case Study: Unsupervised Generative Learning

In unsupervised learning, people want to learn from the distribution of training data without labeled information. Denote $m$ as the real data distribution, and $\tilde{m}$ as the distribution of synthesized outputs. Learning generative model means minimizing the statistical distance between these two distributions $\inf_m D_f(m \parallel \tilde{m})$. However, $m$ is not a given profile, so it needs to be estimated from the training samples. We use Legendre-Fenchel duality to transform it into a minimax optimization problem

$$\begin{aligned}\inf_m D_f(m \parallel \tilde{m}) &= \inf_m \int_\Omega f^{**}\left(\frac{dm}{d\tilde{m}}\right) d\tilde{m} - f(1) \\ &= \inf_m \int_\Omega [\sup_\xi \langle \xi, \left(\frac{dm}{d\tilde{m}}\right)\rangle - f^*(\xi)\,] d\tilde{m} - f(1) \\ &= \inf_m \int_\Omega \sup_\xi \langle \xi, dm\rangle - f^*(\xi)\, d\tilde{m} - f(1) \\ &\geq \inf_m \sup_\xi \int_\Omega \mathbb{E}_m \xi(\omega) - \mathbb{E}_{\tilde{m}} f^*(\xi(\omega)) - f(1)\end{aligned} \tag{19}$$

The optimal structure for $\xi$ is $\xi^*(\omega) = f'(\frac{dm}{d\tilde{m}})(\omega)$. We assume that the optimal $\xi^*$ can be parametrized by and represented as $\xi^*(\omega) = h_f(V(a_1, \omega))$ for some functions $h_f$ and $V$, where $V(a_1, \omega) : A_1 \times \Omega \to \mathbb{R}$, and $h_f : \mathbb{R} \to Dom(f^*)$, $Dom(f^*) = \{x \mid f^*(x) \in \mathbb{R}\}$. The measure $\tilde{m}$ is re-parametrized with $\tilde{m}_{a_2}$. Then,

$$\inf_m D_f(m \parallel \tilde{m}) = \inf_{a_1} \sup_{a_2} \mathbb{E}_{m_{a_1}} h_f(V) - \mathbb{E}_{\tilde{m}_{a_2}} f^*(h_f(V)) - f(1) \tag{20}$$

Therefore, the optimization problem $\inf_m D_f(m \parallel \tilde{m})$ is formulated as a zero-sum game with two adversarial players. The defender acts as a discriminator to find the maximum gap between $m$ and $\tilde{m}$, while the attacker acts as a generator that tries to synthesize indistinguishable samples to fool the discriminator. Since the payoff function is not deterministic and dependent on the other side, each agent optimizes its worst-case performance. It forms a distributionally robust game and the corresponding optimization problem is as follow:

$$(P) \begin{cases} \inf\limits_{a_1 \in A_1} \sup\limits_{a_2 \in A_2} L(a_1, a_2), \\ L(a_1, a_2) = \mathbb{E}_{m_{a_1}} h_f(V(a_1, \omega)) + \mathbb{E}_{\tilde{m}_{a_2}}(-f^*)(h_f(V(a_1, \omega))) - f(1) \end{cases} \tag{21}$$

where $A_1$ denotes the decision space of the attacker and $A_2$ is the decision space of the defender.

**Example 1.** *Let $f(x) = x \log x - (1 + x) \log(1 + x)$. Then the derivative of the function $f$ is $f'(x) = 1 + \log x - 1 - \log(1 + x) = \log(1 - \frac{1}{1+x})$ and $f''(x) = \frac{1}{x(1+x)} > 0$. As $f$ is a convex function, we can apply the Legendre transform*

$$f^*(\xi) = \sup_x [x\xi - f(x)] = -\log(1 - e^\xi)$$

*The parameterized objective function is now*

$$L(a_1, a_2) = \mathbb{E}_{m_{a_1}} h_f(V) + \mathbb{E}_{\tilde{m}_{a_2}} \log(1 - e^{h_f(V)}) - f(1)$$

*If $h_f = \log$, then*

$$L(a_1, a_2) = \mathbb{E}_{m_{a_1}} \log(V) + \mathbb{E}_{\tilde{m}_{a_2}} \log(1 - V) - f(1).$$

In order to provide training data with enough diversity, a particular sampling process was considered in [19]. Real data is randomly sampled from the training set $\omega \sim m$; fake data is first sampled from a noise distribution $z$ then fed into the generative model $G(z) \sim \tilde{m}$. The discriminator takes in both real and synthetic examples, and outputs two probabilities $V = D(\omega)$ and $D(G(z))$ for $a_1, a_2$ respectively. In this game, two opponent agents optimize on different objective functions. It is a zero-sum game with no cooperation. The attacker $G(z)$ tries to fool the defender $D$ by minimizing $\log(1 - D(G(z)))$, which means incorrectly predicting a synthetic example as real. The defender optimizes the discriminator by maximizing the sum $\log(D(\omega)) + \log(1 - D(G(z)))$, which means correctly predicting both the real and fake examples. The generative model is trained by solving the Nash equilibrium in a distributional robust game. The formulation is robust in the sense that the attacker tries to sneak into the real data distribution while facing the supervision of the defender.

Based on the ideas above, we formulate the generative learning problem into the DRG framework. As depicted in Figure 1, there are two groups of players in this game. The attackers train the generative model $G_{\theta_a}(z)$ to produce fake samples $\tilde{x}_i$ that are similar to the real ones, where $z$ is a low dimension random variable feed to the generator and $\theta_a$ is the model parameter. The defenders explore the neighborhood of $m$ and slightly change the real data to produce hard samples $x_i'$ which have the maximum distance from the model distribution. The attackers again refine the model to fit for those hard samples. The loss function is defined by the discrepancy $D(\tilde{m}, m')$, where $m'$ is the hard sample distribution chosen by the defender. Since the real distribution $m$ is unknown and we only have an observation dataset $\{x_1, \cdots, x_N\} \subset \mathbb{R}^d$, the optimization is performed by iteratively updating the generative model as well as the hard sample distribution $m'$, which is an approximation of $m$ within some bounded uncertain set $B_\rho(m)$.

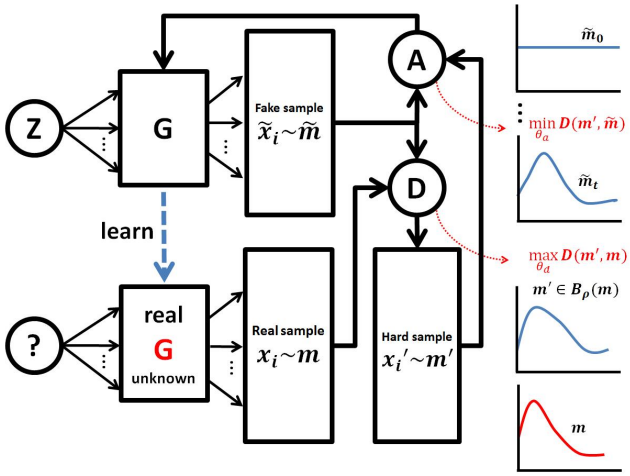



Figure 1: Distributionally robust game (DRG) framework for generative learning

As displayed in the right column of Figure 16, initially, the fake samples are drawn from an arbitrary distribution $\tilde{m}_0$, e.g., a uniform distribution. In each iteration, the attackers refine it closer to the hard sample distribution $m'$, while the defenders keep looking for the worst-case approximation $m'$ within the uncertain set $B_\rho(m)$. Once the algorithm converges, i.e. $D(\tilde{m}, m') \leq \epsilon$, we can ensure that the discrepancy is bounded by a small value if $D(\cdot, \cdot)$ satisfies triangle inequality:

$$D(\tilde{m}, m) \leq D(\tilde{m}, m') + D(m', m) \leq \epsilon + \rho \tag{22}$$

If $m'$ is exactly the worst distribution in $B_\rho(m)$, we can say that $D(\tilde{m}, m) \leq |\rho - \epsilon|$ (see Figure 2). Therefore, the learning task is completed and the fake samples drawn from $\tilde{m}$ will be indistinguishable from the real ones.

Next chapter will discuss the properties of Wasserstein distance as a metric for $D(\cdot, \cdot)$ and compare it with the popular used KL divergence.

# 3 Wasserstein Metric

## 3.1 Introduction

The uncertain set involves a term to measure the similarity between two distributions. There are various ways to define the divergence between two sets. A straightforward way is to sum up the point-wise loss on those two sets, such as $L^p$ distances used in ridge regression ($L^2$-norm) and Lasso ($L^1$-norm), KL divergence and its symmetric alternative Jensen-Shannon (JS) divergence. These loss functions are decomposable and widely adopted in both discriminative and generative tasks. Since the evaluation can be conducted on individual parts, they provide convenience for incremental learning and it's easier to develop efficient algorithms. However, they do not take into account the interactions of the individual points within a set. Non-decomposable losses such as F-measure, total variation and Wasserstein distance capture the entire structure of data and provide better topologies for optimization, at the cost of additional computational burden in loss evaluation.

### 3.1.1 Optimal Transportation Problem

The optimal transport cost measures the least energy required to move all the mass in the initial distribution $f_0$ to match the target distribution $f_1$.

$$C(f_0, f_1) = \inf_{\pi \in \Pi(f_0, f_1)} \int_{\mathcal{X} \times \mathcal{Y}} c(x, y) d\pi(x, y) \tag{23}$$

where $c(x, y)$ is the ground cost for moving one unit of mass from $x$ to $y$, and $\Pi(f_0, f_1)$ denotes the set of all measures on $\mathcal{X} \times \mathcal{Y}$ with marginal distributions $f_0, f_1$, i.e., the collection of all possible transport plans [1].

Wasserstein distance is a specific kind of optimal transport cost in which $c(x, y)$ is a distance function. The $p^{th}$ Wasserstein distance ($p \geq 1$) is defined on a completely separable metric space $(\mathcal{X}, d)$:

$$W_p(f_0, f_1) := (\inf_{\pi \in \Pi(f_0, f_1)} \int_{\mathcal{X} \times \mathcal{X}} d(x, y)^p d\pi(x, y))^{\frac{1}{p}} \tag{24}$$

Specifically, when $p = 1$, $W_1$ is called the Kantorovich-Rubinstein distance or Earth-Mover (EM) distance. It has duality as a supremum over all 1-Lipschitz functions $\psi$:

$$W_1(f_0, f_1) = \sup_{\|\psi\|_{Lip} \leq 1} \mathbb{E}_{f_0}[\psi(x)] - \mathbb{E}_{f_1}[\psi(x)] \tag{25}$$

Moreover, if $f_0, f_1$ are densities defined on $\mathbb{R}$, $F_0, F_1$ denotes their cumulative distribution functions (CDF). In this one dimensional case $f_0, f_1$ have inverse CDFs $F_0^{-1}$ and $F_1^{-1}$. Then the Wasserstein distance can be computed by the integration of their CDF difference.

$$W_1(f_0, f_1) = \int_{\mathbb{R}} |F_0(x) - F_1(x)| dx \tag{26}$$

## 3.2 From KL divergence to Wasserstein Metric

Although KL divergence and its generalized version $f-$divergence are very popular in generative learning literatures, using Wasserstein metric $D(m, \tilde{m}) = W_p(m, \tilde{m})$ has at least three advantages. First, Wasserstein metric is a true distance: the properties of positivity, symmetry and triangular inequality are fulfilled. Thanks to triangular inequality (Figure 2), the maximum discrepancy $D(m, \tilde{m})$ is bounded by $\rho + \epsilon$ when the optimizer approaches the distributionally robust equilibrium.

Second, the Wasserstein space has a finer topology where the loss changes smoothly with respect to the model parameters, thus effective gradients are always available during optimization. The information based divergence like KL doesn't recognize the spatial relationship between random variables. $D_{KL}(m, \tilde{m}) = \int m(x) \log(\frac{m(x)}{\tilde{m}(x)}) dx$ is invariant of reversible transformations on $x = (x_1, x_2, \cdots)^T$ because $m(x)dx$ removes the dimensional information. This property is illustrated in Figure 3. Therefore defining the uncertain set in Wasserstein space is more reasonable than using KL. It ensures the hard samples drawn from the Wasserstein neighborhood $B_\rho(m)$ will not deviate too far from the real distribution.

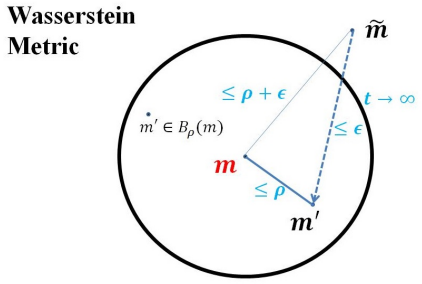



Figure 2: Wasserstein metric as a true distance

Third, f-divergence $D_f(m \,||\, \tilde{m}) = \int_\Omega f(\frac{dm}{d\tilde{m}}) d\tilde{m}$ requires the model distribution $\tilde{m}$ to be positive everywhere, which is not possible in many cases. But adding a widespread noise term to enforce this constraint will lead to unwanted blur in generated samples [20]. The Wasserstein metric does not impose such constraint, thus can produce sharp images.

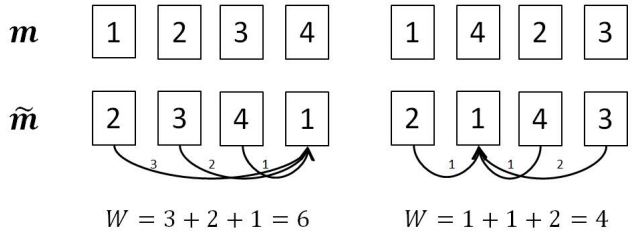



Figure 3: An example to show Wasserstein metric can recognize permutation changes, while KL divergence outputs the same value.

Thanks to these properties, optimizing with Wasserstein loss can continuously improve the model, while the discontinuity behavior of KL divergence will deteriorate the gradients and makes the training process unstable.

Consider the learning process as changing the model from one distribution $\tilde{m}_0$ to another $\tilde{m}_t$, then each path represents a specific kind of interpolation between them. A good learning algorithm should seek for the optimal path. Next part shows that Wasserstein metric prefers displacement interpolation than linear interpolation, which has better property in generative learning.

### 3.3 Dynamic Optimal Transport

Optimizing the geodesic path between two distributions gives the optimal transport cost as well as displacement interpolations. Compare with simple linear interpolations, displacement interpolations can keep modes (e.g., the Gaussian peak) and reflect translational motions between two objects. Figure 4 shows that under Wasserstein metric, displacement interpolation (up) has lower cost than linear interpolation (down), while in KL-divergence the contrary is the case.

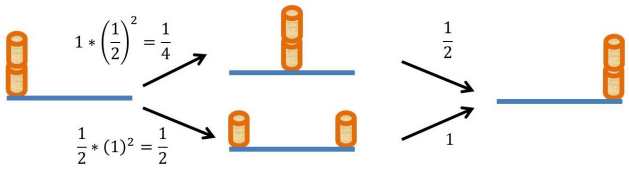



Figure 4: Displacement interpolation has lower cost in Wasserstein metric.

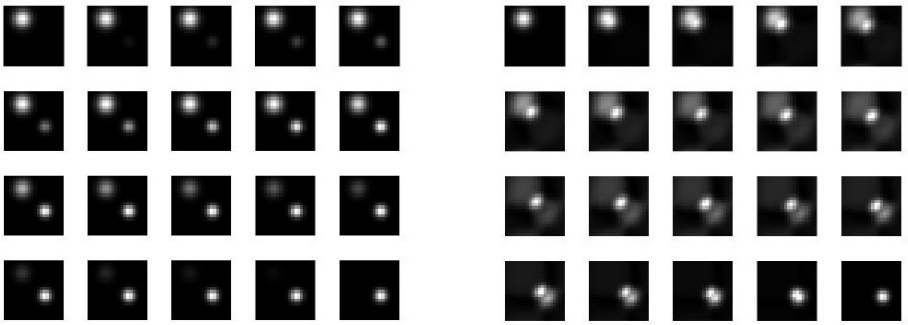
### 3.4 Case Study: A Toy Example

This example shows learning the optimal transportation between two density distributions. The initial and target distributions are given by two Gaussians (Figure 5).

$$\begin{aligned} m_0 &= \mathcal{N}(\mu = (0.2, 0.3), \sigma = 0.1) \\ m_1 &= \mathcal{N}(\mu = (0.6, 0.7), \sigma = 0.07) \end{aligned} \tag{27}$$

Figure 5: Initial and target distribution $m_0, m_1$.

Initially, the intermediate states are set as linear interpolations, where a unimodal distribution is changed to bimodal. This is not desirable because it doesn't keep the mode. (think of an object being split into two parts while moving and then put together, as the lower part of Figure 4). It has been proved in [32] that optimizing with Wasserstein loss can preserve variance of the intermediate distributions, thus the object moves from one position to another without changing its shape. Figure 6 shows that after 1000 iterations, the Gaussian peak is preserved in the intermediate distributions, and the loss decreases to the optimal transport cost (Figure 7).



Linear interpolation $(iter = 0)$ Displacement interpolation $(iter = 1000)$

Figure 6: Learning the optimal transportation between two normal distributions.

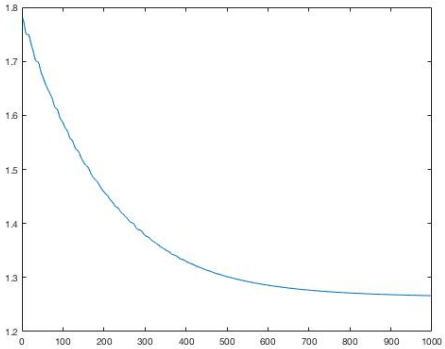



Figure 7: Transportation cost during optimization.

# 4 Learning Algorithms

In this section we provide learning procedure to solve the distributionally robust Nash equilibria, and develop practical implementation algorithms to train deep generative models.

## 4.1 Distributionally Robust Optimization

In DRG, the attacker and defender work against each other to find the robust Nash equilibrium by solving a minimax optimization problem

$$(P_j): \ \inf_{a_j \in \mathcal{A}_j} \ \sup_{m' \in B_\rho(m)} \ \mathbb{E}_{m'} l_j(a, \omega')$$

Since $B_\rho(m)$ is a subset of Lebesgue space (the set of integrable measurable functions under $m$), the original problem $(P_j)$ has infinite dimensions, which does not facilitate the computation of robust optimal strategies. It has been proved in section 2.3.3 that $(P_j)$ can be reduced to a finite dimensional stochastic optimization problem when $\omega' \mapsto l_j(a, \omega')$ is upper semi-continuous and $(\Omega, d)$ is a Polish space. We introduce a Lagrangian function for constraint (4),

$$\tilde{l_j}(a, \lambda, m, m') = \int_\omega^{'} l_j(a, \omega') dm' + \lambda(\rho - W(m, m')) \tag{28}$$

the original problem $(P_j)$ becomes

$$(\tilde{P_j}): \ \inf_{a_j \in \mathcal{A}_j, \lambda \geq 0} \sup_{m' \in B_\rho(m)} \tilde{l_j}(a, \lambda, m, m') \tag{29}$$

In robust game $\mathcal{G}(m)$, the defenders search for the worst hard sample distribution $m'$ in the Wasserstein neighborhood of $m$ to maximize its loss against the model $\tilde{m}$. According to the definition of Wasserstein metric with ground distance $d(\cdot, \cdot)$,

$$\begin{aligned} \sup_{m'} \tilde{l_j} &= \lambda\rho + \sup_{m'} \int_{\omega'} [l_j(a, \omega')] \, dm' - \lambda W(m, m') \\ &= \lambda\rho + \int_\omega \sup_{\omega'} [l_j(a, \omega') - \lambda d(\omega, \omega')] \, dm \end{aligned} \tag{30}$$

Define the integrand cost as

$$h_j(a, \lambda, \omega) = \lambda\rho + \sup_{\omega'} [l_j(a, \omega') - \lambda d(\omega, \omega')], \tag{31}$$

then $(\tilde{P_j})$ becomes a finite dimension problem on $\mathcal{A}_j \times \mathbb{R}_+ \times \Omega$ if $\mathcal{A}_j$ and $\Omega$ have finite dimensions

$$(\tilde{P_j^*}) \ \inf_{a_j \in \mathcal{A}_j, \lambda \geq 0} \mathbb{E}_m h_j(a, \lambda, \omega) \tag{32}$$

Since $m$ is an unknown distribution observed by the noisy unsupervised dataset $x_1, \ldots, x_N$, it is challenging to compute the expected payoff $\mathbb{E}_{m'} l_j(a, \omega')$, $\mathbb{E}_m h_j(a, \lambda, \omega)$ and their partial derivatives. We need a stochastic learning algorithm to estimate the empirical gradients for the Wasserstein metric.

For a single player, the stochastic state $\omega_j$ leads to error

$$\varepsilon_j = \nabla_{a,\lambda} h_j(a, \lambda, \omega_j) - \nabla_{a,\lambda} \mathbb{E}_m h_j(a, \lambda, \omega)$$

The variance of $\varepsilon_j$ is high and not vanishing. To handle this, we introduce a swarm of players $\omega_j \sim m$, $j \in \mathcal{J}$, then the error term becomes

$$\varepsilon = \frac{1}{|\mathcal{J}|} \sum_j \ \nabla_{a,\lambda} h_j(a, \lambda, \omega_j) - \nabla_{a,\lambda} \mathbb{E}_m h_j(a, \lambda, \omega)$$

It has zero mean and standard deviation as

$$\sqrt{\mathbb{E}[\varepsilon^2]} = \frac{1}{|\mathcal{J}|} \sqrt{var[\nabla_{a,\lambda} h_j(a, \lambda, \cdot)]}$$

For realized $\omega \leftarrow \{x_1, \ldots, x_N\}$, the expected payoff for $N$ players is $\frac{1}{N} \sum_{j=1}^N h_j(a, \lambda, \omega_j)$, and the optimal strategy is

$$(a^*, \lambda^*) \in \arg\min_{a,\lambda} \sum_{j=1}^N h_j(a, \lambda, \omega_j)$$

This provides an accurate robust equilibrium payoff when $N$ is very large.

### 4.2 Numerical Investigation

To illustrate the stochastic learning algorithm we consider specific robust games with finite number of players. Each player acts as if he is facing a group of opponents whose randomized control actions are limited to a Wasserstein ball, and tries to optimize the worst case payoff. The random variable $\omega$ is distributed over $m$ and we assume it has finite $p$ moments. We choose $|\mathcal{J}| = 2$, $p = 2$, $d(\omega, \omega') = \|\omega - \omega'\|_2^2$ and a convex payoff function $l_j(a, \omega')$ defined on $\mathbb{R}^2 \times \mathbb{R}^2$

$$l_j(a, \omega') = \|\omega' - a\|_2^2 = (\omega_1' - a_1)^2 + (\omega_2' - a_2)^2 \tag{33}$$

The optimal defender state $\omega'^*$ is computed through the Moreau-Yosida regularization, and the attacker's action pushes it closer to the destination $\omega$ as shown in Figure 8.

$$\begin{aligned} \sup_{m'} \tilde{l_j} &= \lambda\rho + \int_{\omega\in\Omega} \phi_j(a, \lambda, \omega) dm \\ \phi_j(a, \lambda, \omega) &= \sup_{\omega'\in\mathbb{R}^2} [l_j(a, \omega') - \lambda d(\omega, \omega')] \\ &= \sup_{\omega'\in\mathbb{R}^2} (\|\omega' - a\|_2^2 - \lambda\|\omega' - \omega\|_2^2) \end{aligned} \tag{34}$$

$$\omega'^* = \omega + \frac{\omega - a}{\lambda - 1}, \quad (\lambda > 1) \tag{35}$$

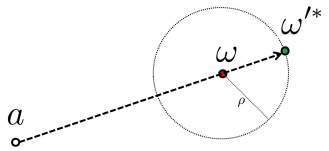



Figure 8: Action pushes the particle toward $\omega$ (which is unknown) given $\omega'^*$

Then $d(\omega, \omega'^*) = \|\frac{\omega - a}{\lambda - 1}\|_2^2$, leads to the worst-case loss

$$l_j(a, \omega'^*) = \|\omega'^* - a\|_2^2 = \frac{\lambda^2}{(\lambda - 1)^2}\|\omega - a\|_2^2 \tag{36}$$

The Moreau-Yosida regularization on $m'$ realized at $\omega'^*$ is

$$\begin{aligned} \phi_j(a, \lambda, \omega) &= l_j(a, \omega'^*) - \lambda d(\omega, \omega'^*) \\ &= \frac{\lambda}{\lambda - 1}\|\omega - a\|_2^2 \end{aligned} \tag{37}$$

The integrand cost function $h_j = \lambda\rho^2 + \frac{\lambda}{\lambda - 1}\|\omega - a\|_2^2$. Thus, problem $(\tilde{P}_j^*)$ becomes

$$\inf_{a,\lambda} \mathbb{E}_m h_j = \inf_{a,\lambda} \int_\omega \lambda\rho^2 + \frac{\lambda}{\lambda - 1}\|\omega - a\|_2^2 \, dm, \quad (\lambda > 1) \tag{38}$$

Given $N$ observations, the stochastic robust loss is

$$\begin{aligned} l_N^* &= \frac{1}{N}\sum_{j=1}^N h_j(a, \lambda, \omega_j) \\ &= \lambda\rho^2 + \frac{\lambda}{N(\lambda - 1)}\sum_{j=1}^N \|\omega_j - a\|_2^2 \end{aligned}$$

We set $\rho = 1$ and $m$ is a dirac distribution where $\omega_j \equiv 1$. Figure 9 plots the trajectories of strategies during learning.

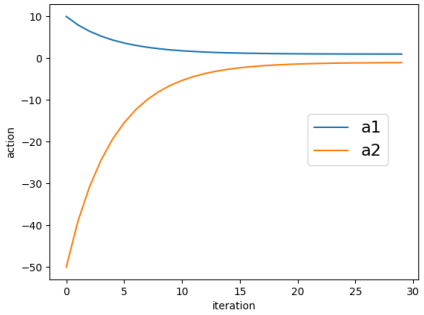



Figure 9: The optimal strategies converges to $(a_1^*, a_2^*) = (1, -1)$

### 4.3 Train a Deep Generative Model

Image generative models such as VAE [21], GAN [22], WGAN [23] have shown great success in recent years. VAE trains a encoder network and a decoder network by minimizing the reconstruction loss, i.e., the negative log-likelihood with a regularizer. It tends to produce blurring images due to the additional noise terms in their model. GAN trains a generator network and a discriminator network by solving a minimax problem based on the KL-divergence. The model is unstable (Figure 13) due to the discontinuity of the information-based loss functions, and the generator is vulnerable to saturation as the discriminator getting better. [24, 25] gives some empirical solutions to these problems, e.g., keeping balance in training generator and discriminator networks, designing a customized network structure. WGAN [23] defines a GAN model by an efficient approximation (equation 25) of the Earth Mover distance. During training, it simply crops all weights of the discriminator network to maintain the Lipschitz constraint.

In our framework, generative learning is formulated as a distributionally robust game with two competitive groups of players, whose actions are defined on the parameter space $\theta = (\theta_a, \theta_d) \in \Theta$. In stochastic settings, $\omega$, $\omega'$ and $\tilde{\omega}$ are instantiated to sample vectors $\{x_1, x_2, \ldots\}$, $\{x'_1, x'_2, \ldots\}$ and $\{\tilde{x}_1, \tilde{x}_2, \ldots\}$. The attacker produces indistinguishable artificial samples $\tilde{x}_i = G_{\theta_a}(z)$ to minimize the discrepancy $\inf_\theta D(\tilde{m}, m')$, where $\tilde{x}_i \sim \tilde{m}$. Meanwhile, the defender produce adversarial samples $x'_i = G_{\theta_d}(x_i)$, which are substitutes of the real ones, to maximize the loss $\sup_{m' \in B_\rho(m)} D(\tilde{m}, m')$, $x_i \sim m$, where $x'_i \sim m'$.

With Moreau-Yosida regularization, the defenders work on the following maximization problem to generate the optimal adversarial samples in Wasserstein ball $B_\rho(m)$,

$$\theta_d^* \in \arg\max_{\theta_d} \; l(\theta_a, \omega') - \lambda d(\omega, \omega')$$

and the attackers work on the minimization problem to find the best generative parameters $\theta_a^*$

$$\theta_a^* \in \arg\min_{\theta_a, \lambda} \; \lambda\rho + l(\tilde{x}, x'^*) - \lambda d(x, x'^*)$$

Given enough observations $\{x_1, x_2, \ldots\}$ from the unknown real distribution $m$, a similar distribution $\tilde{m}$ can be learned by solving the distributionally robust Nash equilibrium. New samples generated from $\tilde{x}_i \sim \tilde{m}$ should be indistinguishable from the real ones. The DRG algorithm is summarized in Algorithm 1.

### 4.4 Case Study: Unsupervised Learning for Clustering

We apply our algorithm to learn a three-dimensional data distribution for the Fisher's Iris dataset [26]. This dataset contains 150 samples, each having five attributes: petal length, petal width, sepal length, sepal width and class. We remove the class label and using PCA to extract three most prominent features. The observed data samples are plotted in Figure 10, each color represents a class of flower. The generative model is set as the affine transformations of three unit balls $z_1, z_2, z_3$. Initially, the attacker parameters $W_1, W_2, W_3$ are set as identity matrices and $b_1 = b_2 = b_3 = 0$, so all fake

**Algorithm 1** DRG with Wasserstein metric

**Input:** real data $(x_i)_{i=1}^N$, batch size $n$, initial attacker parameters $\theta_{a0}$, Lagrangian multiplier $\lambda_0$, initial defender parameters $\theta_{d0}$, number of defender updates per attacker loop $n_d$, Wasserstein ball radius $\rho$, learning rate $\eta_a, \eta_d$, low-dimension random noise $z \sim \zeta$
**Output:** $\theta_a$, $\theta_d$, $\lambda$
**while** $\theta_a$ has not converged **do**
  **for** $t = 1, 2, \ldots, n_d$ **do**
    Sample $(x_i)_{i=1}^n \sim m$ from real dataset
    Sample $(\tilde{x}_i)_{i=1}^n \sim \tilde{m}$ from generator $G_{\theta_a}(z)$
    $y_i \leftarrow E_{\theta_d}(x_i), \quad \tilde{y}_i \leftarrow E_{\theta_d}(\tilde{x}_i)$
    Modify to adversarial samples $y_i' \leftarrow G_{\theta_d}(y_i)$
    $g_d \leftarrow \nabla_{\theta_d} \, l(\tilde{y}_1^n, {y'}_1^n) - \lambda d(y_1^n, {y'}_1^n)$
    $\theta_d \leftarrow \theta_d + \eta_d RMSProp(g_d)$
  **end for**
  Sample $(x_i)_{i=1}^n \sim m$ from real dataset
  Sample $(\tilde{x}_i)_{i=1}^n \sim \tilde{m}$ from generator $G_{\theta_a}(z)$
  $y_i \leftarrow E_{\theta_d}(x_i), \quad \tilde{y}_i \leftarrow E_{\theta_d}(\tilde{x}_i)$
  Modify to adversarial samples $y_i' \leftarrow G_{\theta_d}(y_i)$
  $g_{a,\lambda} \leftarrow \nabla_{\theta_a,\lambda} \, \lambda\rho + l(\tilde{y}_1^n, {y'}_1^n) - \lambda d(y_1^n, {y'}_1^n)$
  $\theta_a \leftarrow \theta_a - \eta_a RMSProp(g_{a,\lambda})$
  $\lambda \leftarrow \lambda - \eta_a RMSProp(g_{a,\lambda})$
**end while**

samples are located in a unit ball shown in Figure 10.

$$\tilde{x} = G_{\theta_a}(z) \in \{W_1 z_1 + b_1, W_2 z_2 + b_2, W_3 z_3 + b_3\}$$
$$x' = G_{\theta_d}(x) = W_0 x + b_0$$

Hard samples are generated from the real ones by another affine transformation with parameters $W_0, b_0$. Since the dimension is low, we directly work with the raw data vectors. We set $n_d = 20, \rho = 0.1, \lambda_0 = 10, \eta_a = 0.1, \eta_d = 0.01$, and the training cost for attackers and defenders are displayed in Figure 11. In each defender loop, the hard samples are refined to maximize the Wasserstein loss, and then the attackers minimize the worst-case cost. After 150 iterations, the algorithm converged at the distributionally robust Nash equilibrium. The generated samples (black dots in Figure 10) successfully covered the region of real samples, which demonstrated the effectiveness of our algorithm.

Next section will show the application of DRG on image synthesis, in which the attackers and defenders are realized with deep neural networks.

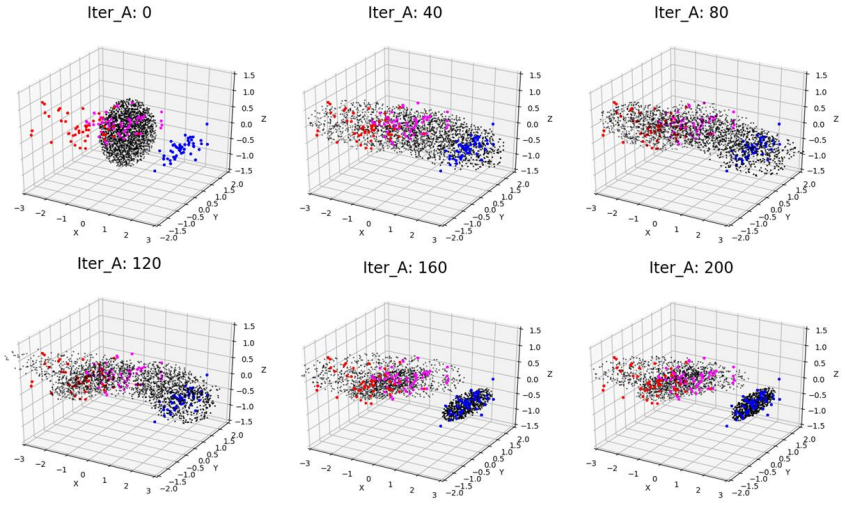



Figure 10: Generative learning for Iris Flower dataset

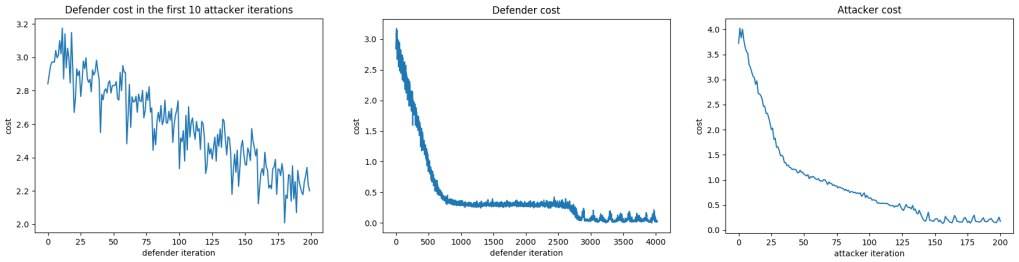



Figure 11: Attacker and defender cost during training

### 4.5 Case Study: Generative Modeling for Image Synthesis

**Dataset** We apply our DRG algorithm on the CelebA [27] dataset to generate artificial human faces. The training set has 202K cropped face images with size $64 \times 64$, therefore each real sample $x_i \sim m$ has 12288 dimensions. The artificial samples are generated from low-dimensional noise vectors $z \sim \zeta$, where $\zeta$ is a random normal distribution.

**Network Structure** In this paper, the generative network $x = G_{\theta_a}(z)$ follows the DCGAN [28] architecture. We design $y' = G_{\theta_d}(y)$ as a single layer neural network to perform modification. For the encoding network $y = E_{\theta_d}(x)$, we use one CNN-ReLU layer followed by 3 CNN-BatchNorm layers and a fully connected layer to produce code vectors. Both networks have about 5 million training parameters.

**Loss Functions** In DRG algorithm, the Wasserstein distance $l(\tilde{x}^n, x'^n)$ is implemented by Sinkhorn-Knopp's algorithm [29], and the ground cost $d(x^n, x'^n) = \frac{1}{n}\sum_{i=1}^{n} \|x_i - x'_i\|_2^2$. Instead of directly computing the L2-norm on raw data vectors, Algorithm 1 uses an encoder network $y = E_{\theta_d}(x)$ to learn useful features.

**Hyperparameters** The encoder maps the original data into a 100-dimension feature space, which matches the dimension of the random noise $z$. In all experiments, the cost based on Wasserstein metric is normalized to $[0, 1]$, where the supremum indicates the cost between images that are all black and all white. The hyperparameters listed in Algorithm 1 are chosen by validation and listed in table 1; others are set as the default values in their references. For training we choose the RMSProp optimizer [30] because it doesn't involve a momentum term. Empirically, we found momentum-related optimizers may deteriorate the training. The reason is, in robust games the payoff function is dynamic and changes every time the other players take actions. Since the structure of the objective surface is not stationary, it's meaningless to follow the velocity of the previous optimization steps.

Table 1: Hyper parameters

| parameters | $n$ | $\rho$ | $\lambda_0$ | $n_d$ | $\eta_a, \eta_d$ | $\theta_{a0}, \theta_{d0}, \eta$ |
|---|---|---|---|---|---|---|
| values | 64 | 0.1 | 10 | 1 | 0.00005 | random normal |

**Evaluation** Experimental results are demonstrated in Figure 12, in which the last line shows the most similar samples in the real dataset. The training curve for DRG is plotted in Figure 14. It means the Wasserstein loss is highly related to the sample quality. By optimizing the worst-case loss function, the DRG model converges very quickly to the real data distribution and successfully produce sharp and meaningful images. In experiments we found that the original GAN generator [28] suffers from unpredictable quality deterioration at iteration $5.3K$, $7.8K$, $10.2K$ (Figure 13), etc, while our algorithm keeps improving the sample quality. This problem is caused by the discontinuity of the KL-divergence.

The evaluation of generative models is itself a research topic. [31] figured out that different evaluation metrics favor different models. For example, a high log-likelihood doesn't mean good visual quality, and vice versa. Therefore, the metric used in training and evaluation should fit for the specific

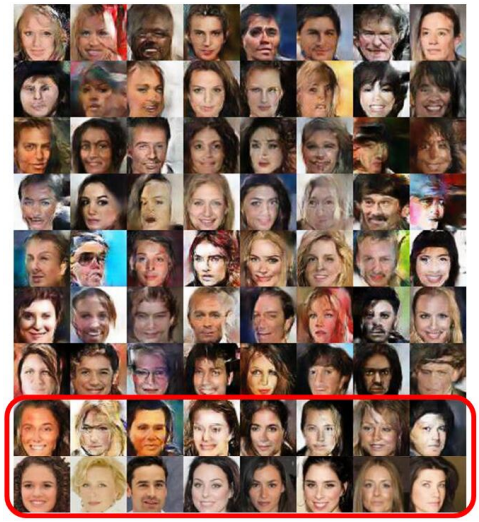


Figure 12: DRG results on CelebA, attacker iteration = 300K

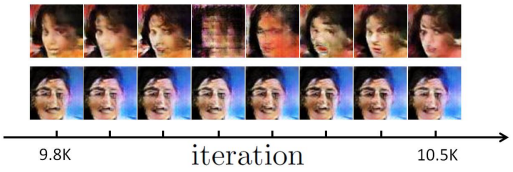



Figure 13: Stability of the generated models. Upper: DCGAN, Bottom: DRG

application. In our case, the learned fake data distribution should be as close as to the real one. So we measure the discrepancy between these distributions using Wasserstein metric. We compare our algorithm with DCGAN [28] and WGAN [23], and report the quantitative results in Table 2.

The computation complexity per attacker iteration is linear $O(n)$ with respect to the batch size. We use a Titan Xp to train the model and plot the computation time in Figure 15. When $n = 64$, it takes 0.2 seconds for an attacker update. Our algorithm has smaller constant factor than WGAN.

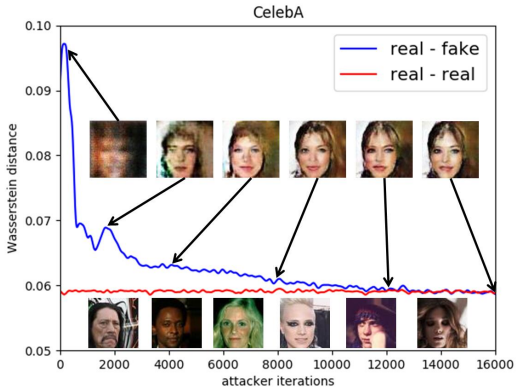



Figure 14: Training curve for DRG algorithm with Wasserstein metric. The loss goes down as generated samples getting better, and converges to the Wasserstein distance between two real data sets. The curves are smoothed for visualization purpose.

Table 2: Performance evaluation

| $W(m, \tilde{m})$ ($\times 10^{-5}$) | 1K samples | 10K samples |
|---|---|---|
| real - real | 12.9 | 1.74 |
| real - DRG | **22.6** | **15.9** |
| real - DCGAN | 37.3 | 16.4 |
| real - WGAN | 31.0 | 17.2 |

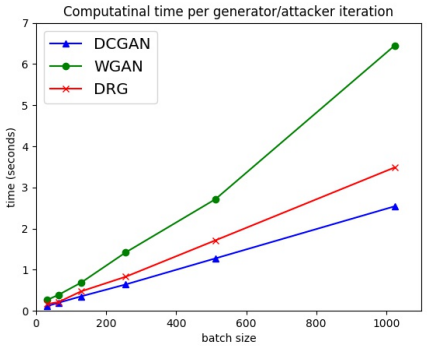



Figure 15: Computation time with respect to batch size.

# 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 $p(y \mid x)$, while generative models synthesize new data with desired labels by drawing samples from estimated distribution $p(x \mid y)$. From a perspective of probabilistic modelling, style transfer learns two conditional distributions $p(x_1 \mid x_2)$ and $p(x_2 \mid x_1)$. 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 marginal distributions $p(x_1), p(x_2)$.

To solve this problem, additional constraints are required. Some researchers proposed to keep a particular part of the data unchanged, e.g., pixel intensity, gradient or object boundaries [2][3]; others suggested to preserve some properties of the data, such as semantic features or class labels [4].

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.

Some approaches [11][12][13] assume there exists a transformation $\mathcal{T}$ such that the source and target data can be matched in a new representation $p(\mathcal{T}(x_1)) = p(\mathcal{T}(x_2))$. The types of transformation

includes projections, affine transform, and non-linear mapping defined by neural networks. The objective is to minimize the gap between two transformed distributions. Several metrics used to compare distributions have been discussed in chapter 3. One requirement is that $p(x_1)$ and $p(x_2)$ share a common support, otherwise the optimization process is instable due to vanishing gradient issues.

Anothter idea is to directly assume the mapping $\mathcal{T} : x_2 = \mathcal{T}(x_1)$ to be the optimal transport between $p(x_1)$ and $p(x_2)$. The optimal solution $\mathcal{T}^*$ minimizes the global transportation cost (Wasserstein distance) between the source and target distributions. As a by-product, minimizing the transporation cost gives the alignment of corresponding samples as well as their labels. It can be used to solve the domain adaptation problem [14] by finding the joint distribution optimal transport between $p(x_1, y_1)$ and $p(x_2, f(x_2))$. $f$ is the classifier in the target domain, which can be infered from the aligned data-label pairs.

While many approaches have been proposed, learning the mapping between different domains is still an open problem. Our work is motivated to support the development of more robust and comprehensible generative models for style transfer.

### 5.3 Related Work

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 $p(x \mid z)$. 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 $p(x \mid z)$ 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

$$p(x) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}) \tag{39}$$

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 nonphoto-realistic 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 ($f_0$), 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 $p(x_j \mid x_i)$ 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 $p(x_j \mid x_i)$. 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 16 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 $j$.

$$\begin{aligned} x'_{i \leftarrow j} &= G_i(c_j, s_i) = G_i(E_j^c(x_j), s_i) \\ x'_{j \leftarrow i} &= G_j(c_i, s_j) = G_j(E_i^c(x_i), s_j) \end{aligned} \tag{40}$$

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 $p(x_j \mid x_i)$, 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.

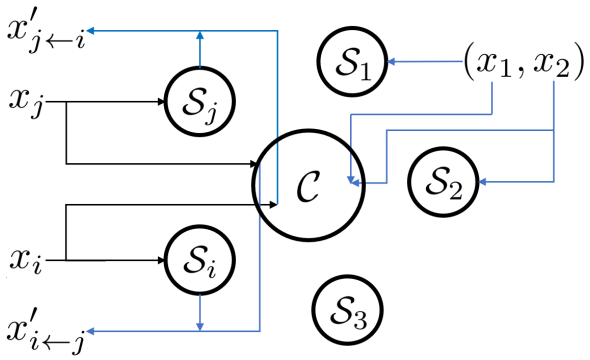



Figure 16: Autoencoder model with partially shared latent space. Sample $x_i$ is encoded into a domain-specific space $\mathcal{S}_i$ and a shared content space $\mathcal{C}$. Corresponding datapoints $(x_1, x_2)$ are encoded in the same content code. Converted sample $x'_{j \leftarrow i}$ is generated 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 3.

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 3: 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.

$$L_{rec}^{x_i} = \mathbb{E}_{x_i}(\|x_i - x'_i\|_1), \quad x'_i = G_i(E_i^c(x_i), E_i^s(x_i)) \tag{41}$$

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 $s_2 \to x'_{2\leftarrow 1} \to s'_{2\leftarrow 1}$.

$$\begin{aligned} L_{cyc}^{c_1} &= \mathbb{E}_{c_1,s_2}(\|c_1 - c'_{2\leftarrow 1}\|_1), \quad c'_{2\leftarrow 1} = E_2^c(x'_{2\leftarrow 1}) \\ L_{cyc}^{s_2} &= \mathbb{E}_{c_1,s_2}(\|s_2 - s'_{2\leftarrow 1}\|_1), \quad s'_{2\leftarrow 1} = E_2^s(x'_{2\leftarrow 1}) \end{aligned} \tag{42}$$

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 $p(x_j), p(x'_{i\leftarrow j})$.

$$L_{GAN}^{x_i} = \mathbb{E}_{c_j,s_i}[\log(1 - D_i(x'_{i\leftarrow j}))] + \mathbb{E}_{x_i}[\log D_i(x_i)] \tag{43}$$

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

$$\begin{aligned} &\min_{E_1^c,E_1^s,E_2^c,E_2^s,G_1,G_2} \max_{D_1,D_2} L(E_1^c, E_1^s, E_2^c, E_2^s, G_1, G_2, D_1, D_2) \\ &= \lambda_s(L_{cyc}^{s_1} + L_{cyc}^{s_2}) + \lambda_c(L_{cyc}^{c_1} + L_{cyc}^{c_2}) + \lambda_x(L_{rec}^{x_1} + L_{rec}^{x_2}) + \lambda_g(L_{GAN}^{x_1} + L_{GAN}^{x_2}) \end{aligned} \tag{44}$$

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

### 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 $E_i^c, G_i, D_i$ 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 17 shows the multi-domain transformation model.

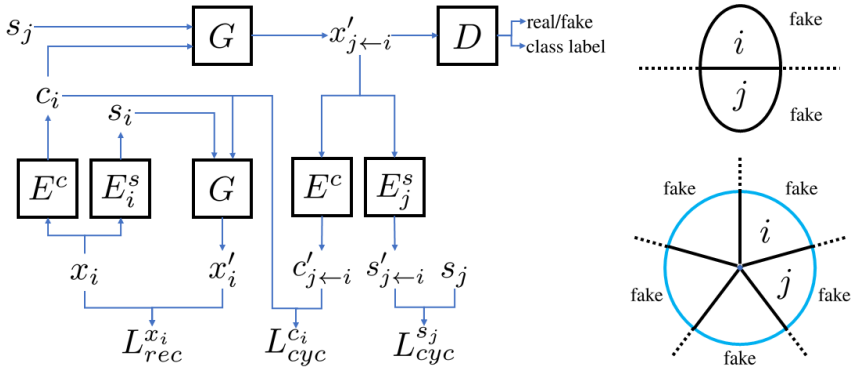



Figure 17: Learn multi-domain transformation. Left: conversion model with shared content encoder, decoder and classifier. Top-right: two domains. Bottom-right: multiple domains (n=5).

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

Image style transfer, or image-to-image translation is a hot topic in computer vision. The objective is to transfer the visual style of an image while keep its semantic content. Similar tasks includes texture synthesis, artistic style transfer, photorealistic image style transfer, etc.

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.

Instead of example-based style transfer, we will focus on leanring the translation model between two image distributions. There are a broad range of researches on image synthesis and representation learning. We will expore the state-of-the-art neural network architecture and implement our model to learn the disentangled representations of image style and content. The proposed method will be tested on a typical image style transfer task.

### 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.

Existing VC approaches cannot be applied directly because they change speaker identity by assuming pronunciation and intonation to be a part of the speaker-independent information. Since the speaker's emotion is mainly conveyed by prosodic aspects, some studies have focused on modelling prosodic

features such as pitch, tempo, and volume [66][67]. In [68], a rule-based emotional voice conversion system was proposed. It modifies prosody-related acoustic features of neutral speech to generate different types of emotions. A speech analysis-synthesis tool STRAIGHT [52] was used to extract fundamental frequency ($F_0$) and power envelope from raw audio. These features were parameterized and modified based on Fujisaki model [69] and target prediction model [70]. The converted features were then fed back into STRAIGHT to re-synthesize speech waveforms with desired emotions. However, this method requires temporal aligned parallel data that is difficult to obtain in real applications; and the accurate time alignment needs manual segmentation of the speech signal at phoneme level, which is very time consuming.

To address these issues, we propose a nonparallel training method. Instead of learning example based one-to-one mapping between paired emotional utterances $(x_1, x_2)$, we learn the conversion model between two emotion domains $(\mathbb{X}_1, \mathbb{X}_2)$.

Inspired by the disentangled representation learning in image style transfer [31][9], we assume that each speech signal $x_i \in \mathbb{X}_i$ can be decomposed into a content code $c \in \mathcal{C}$ that represents emotion-invariant information and a style code $s_i \in \mathcal{S}_i$ that represents emotion-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 content code of the source speech and recombine it with style code of the target emotion. A generative adversarial network (GAN) [15] is added to improve the quality of converted speech. Our approach is nonparallel, text-independent, and does not rely on any manual operation.

For implementation, we use deep neural networks to learn the latent representations of speech. Specifically, the encoders and decoders are implemented with one-dimensional CNNs to capture the temporal dependences. The GAN discriminators are implemented with two-dimensional CNNs to capture the spectra-temporal patterns. All networks are equipped with gated linear units (GLU) [73] as activation functions.

We plan to test our approach on IEMOCAP [71] to learn the conversion models for four emotions: angry, happy, neutral, sad. IEMOCAP is a nonparallel dataset widely used in emotional speech recognition and analysis. It contains scripted and improvised dialogs in five sessions; each has labeled emotional sentences pronounced by two professional English speakers. The emotions in scripted dialogs have strong correlation with the lingual content. Since our task is to change emotion but keep the speaker identity and linguistic content, we only use the improvised dialogs of the same speaker.

There are three metrics for performance evaluation: emotion correctness, voice quality and the ability to retain speaker identity. For subjective evaluation, we will conduct listening tests on Amazon MTurk to evaluate the converted speech. Each example is evaluated by a group of random listeners. They will be asked to manually classify the emotion, and give 1-to-5 opinion scores on voice quality and the similarity with the original speaker. The classification result and mean opinion score (MOS) are two major measurements. For objective evaluation, we plan to use the state-of-the-art speech emotion classifier [72] to check the emotion category of generated speech. It indicates a success if our model can increase the proportion of the target emotions and reduce the original emotions. To our knowledge, this is the first work for nonparallel emotion conversion. If there's time, we will develop multidomain emotion conversion models for unseen speakers.

# 6 Dissertation Outline and Research Timeline

## 6.1 Dissertation Outline

Below is the proposed outline of the planned dissertation.

1. Introduction
   - 1.1. Generative Learning Background
   - 1.2. Game Theory Background
   - 1.3. Motivation
   - 1.4. Summary of Contributions
2. Distributionally Robust Games
   - 2.1. Introduction
   - 2.2. Related Work
   - 2.3. Problem Formulation
   - 2.4. Minimax Robust Game
     - 2.4.1. Infinite Dimension
     - 2.4.2. Triality Theory
     - 2.4.3. Dimension Reduction
   - 2.5. Case Study: Unsupervised Generative Learning
3. Wassertein Metric
   - 3.1. Introduction
   - 3.2. Optimal Transportation Problem
   - 3.3. From KL divergence to Wasserstein Metric
   - 3.4. Displacement Interpolation
   - 3.5. Case Study: A Toy Example Simulation
4. Learning Generative Models
   - 4.1. Distributionally Robust Optimization
   - 4.2. Numerical Investigation
   - 4.3. Train a Deep Generative Model
   - 4.4. Case Study: Unsupervised Learning for Clustering
   - 4.5. Case Study: Generative Modeling for Image Synthesis
5. Image Style Transfer
   - 5.1. Introduction
   - 5.2. Motivation
   - 5.3. Related Work
   - 5.4. Method
     - 5.4.1. Two-domain Transformation
     - 5.4.2. Multi-domain Transformation
   - 5.5. Experiments
6. Emotional Voice Conversion
   - 6.1. Introduction
   - 6.2. Related Work
     - 6.2.1. Emotion-related Features
     - 6.2.2. Nonparallel Training Approaches
     - 6.2.3. Disentangled Representation Learning
   - 6.3. Method
     - 6.3.1. Assumptions
     - 6.3.2. Model
     - 6.3.3. Loss functions

### 6.2 Research Timeline

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

## 7 Conclusion

This proposal contains my plan for graduating with a PhD in the early Fall of 2019, and described how game theory models can be used in generative learning. While this thesis focus on image synthesis and speech conversion, it is a general model and can apply to other kinds of data by replacing the data representation network. Its contributions can help shed a light to a much unexplored field, and help create a path towards fully automatic data generation and manipulation.

# 8 List of Publications

## 8.1 Thesis Related Publications

The following publications are directly related to my dissertation.

1. Jian Gao, Deep Chakraborty, Hamidou Tembine, and Olaitan Olaleye, “Nonparallel emotional speech conversion,” CoRR, vol. abs/1904.00848. (submitted to ICASSP 2019)
2. Jian Gao and Hamidou Tembine, Distributionally Robust Games for Deep Generative Learning, July 2018. DOI: 10.13140/RG.2.2.15305.44644
3. 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
4. Jian Gao and Hamidou Tembine, Distributionally Robust Games: Wasserstein Metric, International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, July 2018
5. 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)*
6. 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)*
7. 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

## 8.2 Other Publications

The papers below were published during my PhD, but will not be included in the thesis, as they diverge too much from the main topic of the research.

1. 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
2. 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*)
3. 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
4. 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)
5. 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

# References


[1] Cédric Villani. *Optimal Transport: Old and New*. Grundlehren der mathematischen Wissenschaften. Springer, 2009 edition, September 2008.

[2] Michele Aghassi and Dimitris Bertsimas. Robust game theory. *Mathematical Programming*, 107(1):231–273, Jun 2006.

[3] Dimitris Bertsimas, David B. Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, 2011.

[4] H.E. Scarf. *A Min-max Solution of an Inventory Problem*. P (Rand Corporation). Rand Corporation, 1957.

[5] Maurice Sion. On general minimax theorems. *Pacific J. Math.*, 8(1):171–176, 1958.

[6] Allen L. Soyster. Technical note - convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations Research*, 21(5):1154–1157, 1973.

[7] Imre Csiszár. Eine informationstheoretische ungleichung und ihre anwendung auf den beweis der ergodizitat von markoffschen ketten. *Magyar. Tud. Akad. Mat. Kutató Int. Közl*, 8:85–108, 1963.

[8] Tetsuzo Morimoto. Markov processes and the h-theorem. *Journal of the Physical Society of Japan*, 18(3):328–331, 1963.

[9] S. M. Ali and S. D. Silvey. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society. Series B*, 28(1):131–142, 1966.

[10] Hamidou Tembine. *Distributed Strategic Learning for Wireless Engineers*. CRC Press, Inc., Boca Raton, FL, USA, 2012.

[11] Evolutionary dynamics over continuous action spaces for population games that arise from symmetric two-player games. *Journal of Economic Theory*, 148(2):743 – 777, 2013.

[12] Lacra Pavel. *Game Theory for Control of Optical Networks*. Static & Dynamic Game Theory: Foundations & Applications. 1 edition.

[13] Gradient dynamics in population games: Some basic results. *Journal of Mathematical Economics*, 46(5):691 – 707, 2010. Mathematical Economics: Special Issue in honour of Andreu Mas-Colell, Part 1.

[14] B. D. Nichols. A comparison of action selection methods for implicit policy method reinforcement learning in continuous action-space. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3785–3792, July 2016.

[15] Stochastic fictitious play with continuous action sets. *Journal of Economic Theory*, 152:179 – 213, 2014.

[16] Pairwise comparison dynamics for games with continuous strategy space. *Journal of Economic Theory*, 153:344 – 375, 2014.

[17] Dario Bauso, Hamidou Tembine, and Tamer Başar. Robust mean field games. *Dynamic Games and Applications*, 6(3):277–303, Sep 2016.

[18] I. Csiszár. Information-type measures of difference of probability distributions and indirect observations. *Studia Scientiarum Mathematicarum Hungarica*, 2:299–318, 1967.

[19] Ian Goodfellow and et al. Generative adversarial nets. In *NIPS*. 2014.

[20] Yuhuai Wu, Yuri Burda, Ruslan Salakhutdinov, and Roger B. Grosse. On the quantitative analysis of decoder-based generative models. *CoRR*, abs/1611.04273, 2016.

[21] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

[22] 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.

[23] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *CoRR*, abs/1701.07875, 2017.

[24] Martín Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *CoRR*, abs/1701.04862, 2017.

[25] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2226–2234, 2016.

[26] M. Lichman. UCI machine learning repository, 2013.

[27] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[28] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.

[29] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific J. Math.*, 21(2):343–348, 1967.

[30] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

[31] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *CoRR*, abs/1511.01844, 2015.