Triple Generative Adversarial Networks

Chongxuan Li, Kun Xu, Jiashuo Liu, Jun Zhu, Member, IEEE, and Bo Zhang

Abstract—Generative adversarial networks (GANs) have shown promise in image generation and classification given limited supervision. Existing methods extend the unsupervised GAN framework to incorporate supervision heuristically. Specifically, a single discriminator plays two incompatible roles of identifying fake samples and predicting labels and it only estimates the data without considering the labels. The formulation intrinsically causes two problems: (1) the generator and the discriminator (i.e., the classifier) may not converge to the data distribution at the same time; and (2) the generator cannot control the semantics of the generated samples. In this paper, we present the triple generative adversarial network (Triple-GAN), which consists of three players—a generator, a classifier, and a discriminator. The generator and the classifier characterize the conditional distributions between images and labels, and the discriminator solely focuses on identifying fake image-label pairs. We design compatible objective functions to ensure that the distributions characterized by the generator and the classifier converge to the data distribution. We evaluate Triple-GAN in two challenging settings, namely, semi-supervised learning and the extreme low data regime. In both settings, Triple-GAN can achieve state-of-the-art classification results among deep generative models and generate meaningful samples in a specific class simultaneously.

Index Terms—Generative adversarial network, deep generative model, semi-supervised learning, extreme low data regime, conditional image generation.

1 INTRODUCTION

Deep neural networks (DNNs) have achieved remarkable progress in various fields, including computer vision [1], [3], [4], [5], speech recognition [6] and natural language processing [7], [8], [9]. These DNN methods are discriminative [10] in the sense that they learn a conditional distribution of an output (e.g., a semantic label) given an input data point (e.g., an image). Generally, discriminative networks are unable to generate new samples or infer the missing input values as they take inputs as pure conditions and do not capture their distribution. Moreover, discriminative networks (especially DNNs) typically rely on a large amount of high-quality labeled data [11], which can be rare and expensive in many important tasks, such as medical analysis and automatic driving [12].

In contrast, generative models in general [10] or the recent development of deep generative models (DGMs) [13], [14], [15], [16] provide a complementary solution. Such methods directly model the input data (as well as labels if presented) via a joint probability distribution, and thereby can generate unseen samples and perform inference on missing values. When labels are considered, such methods can perform classification as well [17]. Besides, by leveraging a deep network architecture, DGMs can learn representations from unlabeled data via (approximate) posterior inference [13] and can reduce the demand for labeled data under a semi-supervised formulation [17].

Notably, among various types of DGMs, generative adversarial networks (GANs) [14] have made significant progress in generating realistic images [18], [19], [20], [21], [22] and learning representations [23], [24], [25] in an unsupervised manner, which makes GANs popular in many vision applications [25], [26], [27], [28], [29]. The paradigm of the original GAN framework is a two-player game, where a generator takes a random noise as input and produces a fake data sample while a binary discriminator identifies whether a certain sample is true or fake. Theoretically, assuming that the discriminator has infinite power, the generator can capture the underlying data distribution when converges.

Though promising and appealing, the representations and models learned by purely unsupervised learning methods like GANs are inappropriate in some important downstream tasks [30], [31]. In this paper, we are interested in two highly recognized but not yet well solved tasks. The first task is classification, which is one of the basic problems in machine learning. The representations learned by unsupervised DGMs (including GANs) can be less discriminative than those learned by supervised DNNs [32]. Therefore, the predictions made based on the representations are not sufficiently accurate. The second task is to learn latent factors of clear physical meaning and generate data according to given semantics. This is of general interest [33], [34], [35] because synthesizing samples in a controllable way is desirable both in research and applications. However, unsupervised learning of such representations is challenging and highly depends on the inductive biases of the model [31]. The conclusion also holds for GANs.

To solve the classification and the conditional generation tasks, recent advances [36], [37], [38], [39], [40] attempt to incorporate supervision into GANs. Because labels can be rare in some real-world scenarios, we investigate whether we can solve the two tasks if only a small number of labels are accessible in this paper. We consider two closely related yet challenging settings. The first one is semi-supervised learning [37], [38], [39], where a large amount of unlabeled data is available and can be explored to boost the classification performance (as well as the conditional generation performance in DGMs [17]). The second one is the more
we emphasize that how to leverage unlabeled data
in the extreme low data regime, where no
unlabeled data is available.

Several methods [37], [38], [39], [40] are proposed to solve
the classification task by using a categorical
discriminator under the same two-player formulation as the original GAN.

Though obtaining a substantial improvement over DNNs in
classification, these methods lack a rigorous analysis of the
equilibrium of the game, and the generator and the classifier
are not guaranteed to converge to the data distribution.

Indeed, in these methods, a single discriminator has to play
two incompatible roles: identifying fake samples and
predicting labels simultaneously. Assume that the generator
converges to the data distribution, meaning that it can
produce a fake sample that looks real. On one hand, as
a necessary condition of the generator’s convergence, the
discriminator should identify it as a fake sample with
non-zero probability (See [14] for the proof). On the other
hand, as a classifier converging to the data distribution,
the discriminator should always treat it as a real sample
and predict the correct class confidently. It conflicts as the
discriminator has two incompatible convergence points,
which indicates that the generator and the classifier (i.e., the
discriminator) may not converge to the data distribution
at the same time. The empirical results in Improved-GAN [39]
well support our analysis. In fact, Improved-GAN proposes
two alternative training objective functions that work well for
either classification or image generation in semi-supervised
learning, but not both (See details in Sec. 2). Therefore,
we argue that optimizing such heuristic objective functions
under the two-player formulation may lead to a sub-optimal
classifier and hence the classification task is still open.

As for the second class-conditional generation task, a
conditional generator, which maps a semantic label to a
data point, can be trained in a “discriminative” manner
straightforwardly given fully labeled data [36]. Existing
methods [18], [24], [40], [41], including DADA [40] in the
extreme low data regime, are in this paradigm. However,
we emphasize that how to leverage unlabeled data in the
conditional generation task is highly nontrivial due to the
presence of missing labels in semi-supervised learning. In
fact, to our best knowledge, none of the existing GANs [37],
[38], [39] can learn disentangled representations with full
use of partially labeled data. Again, we believe that the
problem is intrinsically caused by the two-player formulation
in these methods. Specifically, the discriminators in these
methods estimate a single data instead of a data-label pair
and the label information is totally ignored. Therefore, the
generators will not receive any learning signal regarding the
label information from the discriminators and hence cannot
control the semantics of the generated samples, which is not
satisfactory.

In this paper, we propose the triple generative adversarial
network (Triple-GAN) framework for both classification and
class-conditional image generation with limited supervision.

Specifically, we introduce two conditional networks—a
classifier and a generator to generate fake labels given real
data and fake data given real labels, which will perform
the classification and class-conditional generation tasks
respectively. To jointly justify the quality of the samples from
the conditional networks, we define a single discriminator
network which has the sole role of distinguishing whether
a data-label pair is from the real labeled dataset or not. The
resulting model is called Triple-GAN because we consider
three networks as well as three joint distributions, i.e. the
ture data-label distribution and the distributions defined
by the two conditional networks. Directly motivated by
the desirable equilibrium, we carefully design compatible
objective functions, including adversarial losses and cross-
entropy losses, for the three players. The objective functions
are optimized by alternative stochastic gradient descent [14].

Theoretically, we prove that the unique equilibrium of
Triple-GAN is that the distributions defined by both the
generator and the classifier converge to the data distribution
under a nonparametric assumption [14]. This equilibrium
implies that the classifier of Triple-GAN can converge to
a better solution than existing work [18], [24], [36], [41]
and therefore advances the state-of-the-art results in the
classification task. Besides, the discriminator can access the
label information of the unlabeled data from the classifier
and then force the generator to generate correct image-label
pairs, which solves the class-conditional generation task.

Empirically, we evaluate Triple-GAN on the widely
adopted MNIST [42], SVHN [43], CIFAR10 [44] and Tiny
ImageNet [45] datasets in semi-supervised learning and the
extreme low data regime. Triple-GAN can simultaneously
learn a good classifier and a good conditional generator,
which agrees with our motivation and theoretical results.

Overall, our main contributions are:

1) We analyze the problems in existing GANs with
limited supervision [37], [38], [39], [40] and propose a
game-theoretical Triple-GAN framework consisting of
three players to solve them;

2) We provide a rigorous analysis of the equilibrium of
Triple-GAN under a nonparametric assumption [14]
and prove that Triple-GAN leads to a unique desir-
able equilibrium;

3) We show that Triple-GAN can advance the state-
of-the-art classification results of DGMs on various
datasets substantially in both semi-supervised learn-
ing and the extreme low data regime;

4) We demonstrate that Triple-GAN is the first GAN
based method that can disentangle category from
style features given partially labeled data.

The rest of the paper is organized as follows. Section 2
presents the background. Section 3 presents the Triple-GAN
framework for classification and class-conditional generation
in both semi-supervised learning and the extreme low data
regime. Section 4 surveys the related work. Section 5
presents the experiments. Finally, Section 6 concludes.

2 Notation and Background

We first introduce some notations used in this paper. We
 denote the data as \( x \) and the label as \( y \), following the
true data distribution \( p(x, y) \). We denote the true marginal
distributions of \( x \) and \( y \) as \( p(x) \) and \( p(y) \), respectively. We

1. Note that we have a set of fully labeled data (though of small size)
in the extreme low data regime.

2. Available at https://tiny-imagenet.herokuapp.com/
denote the prior distribution over the latent variables as \( p_z(z) \). We denote the marginal distribution of \( x \) defined by the generator as \( p_g(x) \) and the conditional distribution defined by the classifier as \( p_c(y|x) \).

Below, we revisit the original GAN framework and its important variants with supervision.

**GAN.** The goal of the original GAN [14] is to learn \( p_g(x) \) that approximates \( p(x) \) in purely unsupervised learning. In GAN, a generator \( G \) takes a random vector \( z \) as input and produces a fake sample \( G(z) \). A discriminator \( D \) identifies whether a certain data point \( x \) is sampled from \( p(x) \) or \( p_g(x) \) while \( G \) tries to fool \( D \) by producing realistic samples. Formally, the learning problem of GAN is formulated as a two-player minimax game:

\[
\min_G \max_D \mathbb{E}_{p(x)}[\log(D(x))] + \mathbb{E}_{p_c(z)}[\log(1 - D(G(z)))].
\]

(1)

Theoretically, such a two-player game is sufficient to describe the process for unsupervised learning, or more precisely unconditional image generation. Indeed, under a nonparametric assumption [14], the optimal discriminator is \( D^*_G(x) = p(x)/(p_g(x) + p(x)) \) for a given generator. The optimal discriminator provides a tight estimate of the Jensen-Shannon divergence between \( p(x) \) and \( p_g(x) \), and the equilibrium of GAN is achieved if and only if \( p_g(x) = p(x) \), which is desirable in terms of image generation.

**CatGAN.** CatGAN [37] extends the original GAN to semi-supervised learning by employing a categorical discriminator, which also serves as a classifier. The goal of CatGAN is to learn \( p_g(x) \) that approximates \( p(x) \) and \( p_c(y|x) \) that approximates \( p(y|x) \). The learning problem of CatGAN is formulated as:

\[
\min_G \max_D \mathbb{E}_{p(x)}[\log D(x)] + \mathbb{E}_{p_c(z)}[\log(1 - D(G(z)))].
\]

(2)

The first term is the traditional cross-entropy loss on the labeled data for classification. The second and the third terms are the generalized adversarial losses, which will be analyzed in detail. The objective function is the summation of the two types of losses and the relative weight is controlled by a hyperparameter \( \lambda \). According to the adversarial losses in Eqn. (2), the discriminator decreases the predictive entropy of a real sample while increases the predictive entropy of a fake sample to distinguish these two types of samples. Besides, decreasing the predictive entropy of the real data also provides a way to leverage the unlabeled data for semi-supervised classification [45]. In contrast, the generator tries to fool the discriminator by producing samples that have low predictive entropy. CatGAN also adopts a label balance regulariztion, which is omitted here for simplicity.

Although the adversarial losses and the cross-entropy loss are plausible for generation and classification, respectively, a heuristic combination of such losses lacks a rigorous analysis, in contrast to the original GAN [14], and leads to incompatible convergence points of the discriminator, as argued in Sec. [1].

**Improved-GAN.** Improved-GAN [39] shares the same goals and settings with CatGAN [37] but uses a slightly different categorical discriminator that has one extra class that corresponds to the fake samples. The learning problem for the discriminator is:

\[
\max_D \mathbb{E}_{p(x,y)}[\log p_c(y|x, y < K + 1)] + \mathbb{E}_{p_c(z)}[\log p_c(y = K + 1|G(z))] + \mathbb{E}_{p(x)}[\log(1 - p_c(y = K + 1|x))],
\]

(3)

where \( K \) is the number of real classes and the \((K + 1)\)-th class corresponds to the fake samples.

The generator does not directly fool the discriminator but has two alternative objective functions. The objective of feature matching enforces \( G \) to produce samples that have similar features (extracted by a neural network) to the real samples, which is formulated as:

\[
\min_G \|\mathbb{E}_{p_c(x)}[f(x)] - \mathbb{E}_{p_c(z)}[f(G(z))]\|^2_2,
\]

(4)

where \( f(x) \) is the activation of a certain layer in the discriminator given input \( x \) and \( \| \cdot \|_2 \) is the l2 norm. In practice, feature matching works well in classification but fails to generate indistinguishable samples [39]. The other objective of minibatch discrimination considers the difference of the features in a minibatch of data to avoid mode collapse [19] in image generation. It is good at image generation with full labels but cannot predict labels accurately. Improved-GAN shares the same drawbacks with CatGAN [37] and the empirical results of the two objective functions well support our analysis in Sec. [1], namely, that the generator and classifier in [39] may not converge to the data distribution at the same time. In our experiments, we directly compare with the feature matching baseline since we are interested in classification and conditional generation with limited supervision.

**DADA.** DADA [40] considers a more challenging extreme low data regime setting, where only a small number of labeled data is available. DADA also employs a two-player formulation and extends the discriminator to \( 2K \) classes, where the first \( K \) classes are real and the last \( K \) classes are fake. DADA proposes a two-phase training schedule. In the first joint training phase, the learning objective of the discriminator is:

\[
\max_D \mathbb{E}_{p(x,y)}[\log p_c(y|x, y < K + 1)] + \mathbb{E}_{p_c(z)(p(y)}[\log p_c(y = K + 1|G(z), y, K < y < 2K + 1)],
\]

(5)

and the learning objective of the generator is:

\[
\min_G -\mathbb{E}_{p(x,y)}[\log p_c(y - K|x, K < y < 2K + 1)] + \lambda \mathbb{E}_{p(x)}[\|f(x)|y] - \mathbb{E}_{p_c(z)p(y)}[f(G(z, y))]|y]\|^2_2,
\]

(6)

where the last term is a conditional variant of the feature matching loss [39], as presented in Eqn. (4) and \( \lambda \) is a hyperparameter. In the second phase, the generator is fixed as a data provider and the classifier is trained for classification based on both the real data and fake data. Similar to previous work [37, 39], DADA is heuristics without a guarantee that the players will converge to the data distribution.

**CGAN.** The conditional GAN (CGAN) [36] and its variants [18, 24, 40, 41] aim to learn \( p_g(x|y) \approx p(x|y) \) to solve the conditional generation task. CGAN is very similar to GAN but supplies extra label information for both
the generator and the discriminator. Formally, the learning problem of CGAN is formulated as:

\[
\min_G \max_D \mathbb{E}_{p(x,y)}[\log D(x,y)] + \mathbb{E}_{p_c(z)p(y)}[\log(1 - D(G(y,z), y))]. \tag{7}
\]

However, it can be seen that CGAN requires fully labeled data to train a conditional generator in a “discriminative” way and it is non-trivial to leverage unlabeled data by extending CGAN. Besides, existing semi-supervised methods also lack the ability of conditional generation.

3 Method

As discussed above, prior GANs with limited supervision are not problemless: (1) the generator and the classifier (i.e., the discriminator) may not converge to the data distribution at the same time; and (2) the generator cannot control the semantics of the generated samples. Therefore, we argue that the classification and conditional generation tasks given limited supervision are still largely open. To this end, we proposed a unified game-theoretical framework triple generative adversarial network (Triple-GAN), which intrinsically avoid the problems in existing methods.

In particular, we consider two closely related yet challenging learning settings in this paper. The first one is a semi-supervised learning, where we have a labeled dataset of small size and an unlabeled of large size. The second one is the extreme low data regime, where we only have a labeled dataset of small size. In both settings, we want to predict a label \( y \) for an input \( x \) (i.e., perform classification) as well as to generate a new sample \( x \) conditioned on a label \( y \) (i.e., generate samples in a given class).

Triple-GAN is based on the insight that the joint distribution can be factorized in two ways, namely, \( p(x,y) = p(x)p(y|x) \) and \( p(x,y) = p(y)p(x|y) \), and that the conditional distributions \( p(y|x) \) and \( p(x|y) \) are of interests for classification and class-conditional generation, respectively.

To jointly estimate these conditional distributions, which are characterized by a classifier network and a class-conditional generator network, we define a single discriminator network which has the sole role of distinguishing whether a sample is from the true data distribution or the models. Hence, we naturally extend GAN to Triple-GAN, a three-player game to characterize the process of classification and class-conditional generation with limited supervision, as detailed below.

3.1 A game with three players

Triple-GAN consists of three components: (1) a classifier \( C \) that (approximately) characterizes the conditional distribution \( p_c(y|x) \approx p(y|x) \); (2) a class-conditional generator \( G \) that (approximately) characterizes the conditional distribution in the other direction \( p_g(x|y) \approx p(x|y) \); and (3) a discriminator \( D \) that distinguishes whether a pair of data \((x,y)\) comes from the true distribution \( p(x,y) \). All the components are parameterized as neural networks. Our desired equilibrium is that the joint distributions defined by the generator and the classifier both converge to the true data distribution. To achieve it, we design a game with compatible objective functions for the three players as follows.

We make a mild assumption that the samples from both \( p(x) \) and \( p(y) \) can be easily obtained. In the game, after a sample \( x \) is drawn from \( p(x) \), \( C \) produces a fake label \( y \) given \( x \) following the conditional distribution \( p_c(y|x) \). Hence, the fake input-label pair is a sample from the joint distribution \( p_c(x,y) = p(x)p_c(y|x) \). Similarly, a fake input-label pair can be sampled from \( G \) by first drawing \( y \sim p(y) \) and then drawing \( x|y \sim p_g(x|y) \); hence from the joint distribution \( p_g(x, y) = p(y)p_g(x|y) \). For \( p_g(x|y) \), we assume that \( x \) is transformed by the latent style variables \( z \) given the label \( y \), namely, \( x = G(y, z), z \sim p_z(z) \), where \( p_z(z) \) is a simple normal distribution.

3. In semi-supervised learning, \( p(y) \) is assumed same to the distribution of labels on labeled data, which is uniform in our experiment.
distribution (e.g., uniform or standard normal). Then, the fake input-label pairs \((x, y)\) generated by both \(C\) and \(G\) are sent to the single discriminator \(D\). \(D\) can also access the input-label pairs from the true data distribution as positive samples. We refer the objective functions in the process as adversarial losses, which can be formulated as a minimax game:

\[
\min_{C,G} \max_D \mathbb{E}_{p(x,y)}[\log D(x,y)] + \alpha \mathbb{E}_{p_c(x,y)}[\log(1 - D(x,y))] + (1 - \alpha) \mathbb{E}_{p_g(x,y)}[\log(1 - D(G(y,z), y))],
\]

where \(\alpha \in (0, 1)\) is a constant that controls the relative importance of generation and classification and we focus on the balance case by fixing it as 1/2 throughout the paper.

The game defined in Eqn. (8) achieves its equilibrium if and only if \(p(x,y) = (1 - \alpha)p_g(x,y) + \alpha p_c(x,y)\) (See proof in Sec. 3.2). Intuitively, it means that the discriminator balances between the true data distribution and a mixture distribution defined by the generator and the classifier. Further, the equilibrium indicates that if one of \(C\) and \(G\) tends to the data distribution, the other will also go towards the data distribution, which avoids the incompatible convergence problem as in previous work [37], [39], [40].

However, unfortunately, the adversarial losses defined in Eqn. (8) cannot guarantee that \(p(x,y) = p_g(x,y) = p_c(x,y)\) is the unique equilibrium, which is unsatisfactory. A natural solution to the problem is to introduce an additional loss to encourage two of the three distributions to be the same, e.g., \(p_c(x,y) = p(x,y)\). Because the cross-entropy loss is proven effective for classification and efficient to calculate, we introduce the cross-entropy loss on the labeled data to \(C\) as follows:

\[
\mathcal{R}_C = \mathbb{E}_{p(x,y)}[-\log p_c(y|x)],
\]

which is equivalent to the KL-divergence between \(p_c(x,y)\) and \(p(x,y)\) for optimization. Consequently, the game with the cross-entropy loss \(\mathcal{R}_C\) is defined as:

\[
\min_{C,G} \max_D \mathbb{E}_{p(x,y)}[\log D(x,y)] + \alpha \mathbb{E}_{p_c(x,y)}[\log(1 - D(x,y))] + (1 - \alpha) \mathbb{E}_{p_g(x,y)}[\log(1 - D(G(y,z), y))] + \mathcal{R}_C.
\]

For simplicity, we denote the objective function in Eqn. (8) as \(U(C, G, D)\) and that in Eqn. (10) as \(\tilde{U}(C, G, D)\). It will be proven that the game defined by \(\tilde{U}(C, G, D)\) has a unique equilibrium that both \(p_c(x,y)\) and \(p_g(x,y)\) converge to \(p(x,y)\) in the following section.

### 3.2 Theoretical Analysis and Additional Regularization

We now provide a formal theoretical analysis of Triple-GAN under a nonparametric assumption [14] and further introduce a pseudo discriminative loss, which is a regularization that encourages \(p_c(x,y) = p_g(x,y)\) to boost the classification performance without changing the global equilibrium. For clarity of the main text, we only present the main results here and defer the proof details to Appendix A.

First, we would like to show that in the game defined by \(U(C, G, D)\), the optimal \(D\) given \(C\) and \(G\) balances between the true data distribution and a mixture distribution of \(p_c(x,y)\) and \(p_g(x,y)\), as summarized in Lemma 3.1.

#### Lemma 3.1

For any fixed \(C\) and \(G\), the optimal \(D\) of the game defined by the objective function \(U(C, G, D)\) is:

\[
D^*_C,G(x,y) = \frac{p(x,y)}{p(x,y) + \alpha p_c(x,y)},
\]

where \(p_\alpha(x,y) := (1 - \alpha)p_g(x,y) + \alpha p_c(x,y)\) is a mixture distribution for \(\alpha \in (0, 1)\).

Given \(D^*_C,G\), we can omit \(D\) and reformulate the minimax game with the objective function \(U(C, G, D)\) as:

\[
V(C,G) = \max_D U(C,G,D),
\]

whose minimum is summarized as in Lemma 3.2.

#### Lemma 3.2

The global minimum of \(V(C,G)\) is achieved if and only if \(p(x,y) = p_\alpha(x,y)\).

The lemma shows that the adversarial losses ensure that if one of \(p_c(x,y)\) and \(p_g(x,y)\) gets closer to the data distribution after convergence, the other one will also be closer to the data distribution.

We can further show that \(C\) and \(G\) can at least capture the marginal distributions of data, especially for \(p_g(x)\), even there may exist multiple global equilibria, as summarized in Corollary 3.2.1.

#### Corollary 3.2.1

Given \(p(x,y) = p_\alpha(x,y)\), the marginal distributions of \(p_g(x,y)\), \(p_c(x,y)\), and \(p(x,y)\) are the same, i.e., \(p(x) = p_g(x) = p_c(x) = p_\alpha(x)\). \(G\) and \(C\) do not compete as in the two-player based formulation and it is easy to verify that \(p(x,y) = p_c(x,y) = p_g(x,y)\) is a global equilibrium point. However, it may not be unique and we should minimize an additional objective to ensure the uniqueness. In fact, this is true for the objective function \(\tilde{U}(C,G,D)\) in problem (10), as stated below.

#### Theorem 3.3

The equilibrium of \(\tilde{U}(C,G,D)\) is achieved if and only if \(p(x,y) = p_\alpha(x,y)\).

The conclusion essentially motivates our design of Triple-GAN, as we can ensure that both \(p_g(x,y)\) and \(p_c(x,y)\) will converge to the true data distribution if the model has been trained to achieve the optimum.

We can further show another nice property of \(\tilde{U}\), which allows us to regularize our model for a more stable and better convergence in practice without changing the global equilibrium, as summarized below.

#### Corollary 3.3.1

Adding any divergence (e.g., the KL divergence) between any two of the joint distributions or the conditional distributions, or the marginal distributions, to \(\tilde{U}\) as an additional regularization to be minimized, will not change the global equilibrium of \(\tilde{U}\).

Because label information is extremely insufficient in the two settings considered in this paper, we propose a pseudo discriminative loss, which is in the form of a cross-entropy loss of \(C\) on the samples generated by \(G\). Formally, we have:

\[
\mathcal{R}_P = \mathbb{E}_{p_g(x,y)}[-\log p_c(y|x)].
\]

Intuitively, a good \(G\) can provide meaningful labeled data beyond the training set as extra side information for \(C\), which will boost the predictive performance (See Sec. 5 for the empirical evidence). In this sense, the generative process can
be viewed as a type of data augmentation, which has been proven effective with a large amount of labels in a concurrent work [46] but not in the two challenging settings in our paper. Theoretically, minimizing the pseudo discriminative loss with respect to the parameters in \( C \) is equivalent to minimizing the KL-divergence between \( p_\theta(x, y) \) and \( p_c(x, y) \) (See proof in Appendix A). Note that the KL-divergence cannot be optimized directly because its computation involves the unknown likelihood ratio \( p_\theta(x, y) / p_c(x, y) \). Therefore, the pseudo discriminative loss provides a simple way to regularize the classifier without changing the global equilibrium following Corollary [3.3.1]. The pseudo discriminative loss is weighted by a hyperparameter \( \alpha_P \) and added to the loss function of the classifier directly.

Given the above objective functions, we optimize the three networks alternatively by stochastic gradient descent (SGD) based methods. Let \( \theta_C, \theta_D \) and \( \theta_G \) denote the trainable parameters in \( C, D \) and \( G \), respectively. We need to compute the gradients with respect to all the parameters. The gradients with respect to \( \theta_D \) and \( \theta_G \) can be easily obtained. However, the computation of the gradients of \( \mathbb{E}_{p(x)}\mathbb{E}_{p_c(y|x)}[\log(1 - D(x, y))] \) with respect to \( \theta_c \) involves a summation over a discrete random variable \( y \), i.e. the class label. Note that directly applying the Monte Carlo method is not feasible because the feedback of the discriminator is not differentiable with respect to \( \theta_c \). Therefore, in the extreme low data regime, since the size of the data is rather small, we directly integrate out the class label and the gradients are as follows:

\[
\mathbb{E}_{p(x)} \left[ \nabla_{\theta_c} \sum_{y \in \mathcal{Y}} p_c(y|x) \log(1 - D(x, y)) \right],
\]

where \( \mathcal{Y} \) is the set of all possible categories. However, integrating out the class label is time-consuming for a large number of unlabeled data in semi-supervised learning. Therefore, we use a variant of the REINFORCE algorithm [47] to estimate the gradients of \( \theta_c \) as follows:

\[
\mathbb{E}_{p(x)} \mathbb{E}_{p_c(y|x)} \left[ \nabla_{\theta_c} \log p_c(y|x) \log(1 - D(x, y)) \right].
\]

In our experiment, inspired by the straight-through estimator [48], we find the best strategy is to use the most probable \( y \) instead of sampling a \( y \) according to \( p_c(y|x) \) to approximate the expectation in Eqn. (14). We refer the readers to [48] for a detailed discussion about the gradient estimator. The Algorithm 1 presents the whole training procedure.

### 3.3 Training Techniques in Semi-Supervised Learning

The above algorithm is a general framework that can be applied to both semi-supervised learning and the extreme low data regime. However, because the central idea of semi-supervised learning is to leverage the unlabeled data, we introduce two additional techniques to exploit the unlabeled data for better classification performance.

1. The techniques presented in this section are not applicable in the extreme low data regime, where we do not have any unlabeled data.
may lead to a biased solution theoretically but work well in practice.

The first empirical technique for semi-supervised learning is to augment the labeled data in the training of $D$. In particular, we generate pseudo labels through $C$ for some unlabeled data and use these pairs as positive samples for $D$ to accept. Via this trick, $D$ will access samples beyond the empirical distribution of the small-size labeled data and encourage $G$ to generate diverse samples. The cost of the trick is on introducing some bias to the target distribution of $D$, which is a mixture of $p_c(x, y)$ and $p(x, y)$ instead of the pure $p(x, y)$. However, this is acceptable as $C$ converges quickly and $p_c(x, y)$ and $p(x, y)$ are close (See results in Sec. 5).

Since properly leveraging the unlabeled data is key to success in semi-supervised learning, it is necessary to regularize $C$ heuristically as in many existing methods [30], [37], [49], [50] to make more accurate predictions. We consider two alternative losses on the unlabeled data. The first one is the entropy minimization loss [37] as follows:

$$R_{ut} = \mathbb{E}_{p(x)}\mathbb{E}_{p_c(y|x)}[-p_c(y|x)],$$

which encourages $C$ to make predictions confidently on the unlabeled data. The second one is the consistency loss [50], which penalizes the network if it predicts the same unlabeled data inconsistently given different noises like dropout masks. Formally, the consistency loss is defined as follows:

$$R_{ut} = \mathbb{E}_{p(x)}|p_c(y|x, \epsilon) - p_c(y|x, \epsilon')|,$$

where $|| \cdot ||^2$ is the square of the $l_2$-norm and $\epsilon$ and $\epsilon'$ denote two different random noises. Following [37], [50], we use the entropy minimization loss on the MNIST and SVHN datasets and the confidence loss on the CIFAR10 and Tiny ImageNet datasets (See details in Sec. 5). Note that adding such extra unlabeled data losses won’t hurt but help the convergence of $G$ in practice because the losses only affect $C$ directly and if $C$ can make more accurate predictions (i.e. $p_c(x, y)$ gets closer to $p(x, y)$), then $G$ will become better according to Lemma 3.2.

4 Related Work

Various approaches have been developed to learn directed DGMs, including variational autoencoders (VAEs) [13], [51], generative moment matching networks (GMMNs) [52], [53], generative adversarial nets (GANs) [14] and autoregressive models [15], [54]. These learning criteria are systematically compared in [55] and among them, GANs have proven effective on generating realistic samples [18], [19], [20], [21], [22] and becomes popular in vision applications [26], [27], [28], [29].

Recent work has introduced inference networks in GANs for representation learning. In ALI [23], [24], the inference network approximates the posterior distribution of latent variables given true data in an unsupervised manner. Triple-GAN also has an inference network (classifier) as in ALI but there exist two important differences in the global equilibria and the objective functions between them: (1) Triple-GAN matches both the distributions defined by the generator and classifier to true data distribution while ALI only ensures that the distributions defined by the generator and the inference network to be the same; (2) the discriminator will reject the samples from the classifier in Triple-GAN while the discriminator will accept the samples from the inference network in ALI, which leads to different update rules for the discriminator and inference network. These differences naturally arise because Triple-GAN is proposed to solve the problems of GANs with limited supervision as analyzed in Sec. 1. Indeed, ALI [24] uses the same approach as Improved-GAN [39] to deal with partially labeled data and hence it still suffers from the same problems (See a direct comparison in Table 1). Following ALI, InfoGAN [34] and Graphical-GAN [35] disentangle explainable factors from other latent features in a purely unsupervised manner. However, such methods highly depend on proper inductive biases of the model and the data [31]. Besides, the meaning of the disentangled factors cannot be predefined. In comparison, Triple-GAN can avoid the issues under the help of the given supervision.

Deep generative models can be naturally extended to handle partially labeled data. For instance, the conditional VAE [17] and its variants [30], [56] treat the missing labels as latent variables and infer them for unlabeled data. Apart from the work mentioned in Sec. 4, BadGAN [57] is another semi-supervised GAN-based approach, which is based on the same observation that the objective functions in existing semi-supervised GANs are incompatible. However, BadGAN focuses on the classification problem and trains the generator to attack the decision boundary of the classifier instead of to match the data distribution. Therefore, BadGAN achieves comparable classification results with Triple-GAN but leave the generation problem unsolved.

Some preliminary results were published in [58]. We extend the original version by presenting the framework in a more coherent and detailed way, extending the framework to the challenging extreme low data regime, and presenting new results on the Tiny ImageNet dataset.

5 Experiments

In this section, we first present the datasets and the experimental settings briefly. Then, we show the classification and conditional generation results in both semi-supervised learning and the extreme low data regime.

5.1 Datasets and Settings

We evaluate Triple-GAN on the widely adopted MNIST [42], SVHN [43], and CIFAR10 [44] and Tiny ImageNet datasets. MNIST consists of 50,000 training samples, 10,000 validation samples and 10,000 testing samples of handwritten digits of size $28 \times 28$. SVHN consists of 73,257 training samples and 26,032 testing samples and each is a colored image of size $32 \times 32$, containing a sequence of digits with various backgrounds. CIFAR10 consists of colored images distributed across 10 general classes—airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. There are 50,000 training samples and 10,000 testing samples of size $32 \times 32$ in CIFAR10. We split 5,000 training data of SVHN and CIFAR10 for validation if needed. On CIFAR10, we follow [50] to perform ZCA for the input of $C$ but still generate and discriminate the raw images using $G$ and $D$, respectively. The Tiny ImageNet dataset consists of natural images of size $64 \times 64$. We select
10 classes, including lion, snow mountain, coffee, teddy bear, penguin, cat, tower, butterfly, car and cabin, out of the 200 classes and downscale the images to $32 \times 32$ (the dataset will be released with the source code). Each class consists of 500 training samples and 50 testing samples with diverse visual appearance. We randomly split ten percent of the training samples as the validation set if required.

We implement our method based on Theano and here we briefly summarize our experimental settings. For a fair comparison, all the classification results of the baselines are from the corresponding papers or obtained based on the implementation released by the corresponding authors. The generator and classifier of Triple-GAN have comparable architectures to those of the baselines [37, 39, 40] (See details in Appendix E). Besides, we average the classification results of Triple-GAN by multiple runs with different random initialization and splits of the training data and report the mean error rates with the standard deviations following existing work [39].

The pseudo discriminative loss is not applied until the number of epochs reaches a threshold that the generator could generate meaningful data. We search the threshold in $\{200, 300\}$, $\alpha_p$ in $\{0.1, 0.03\}$ and the global learning rate in $\{0.0003, 0.001\}$ based on the validation performance on each dataset. All of the other hyperparameters including relative weights and parameters in Adam [60] are fixed according to [39] across all of the experiments. Further, in our experiments, we find that the training techniques for the original two-player GANs [39, 61] are sufficient to stabilize the optimization of Triple-GAN.

### 5.2 Semi-Supervised Learning

In semi-supervised learning, we will show Triple-GAN can make predictions accurately and control the semantics of the generated samples as a unified model.

#### 5.2.1 Classification

Firstly, we compare our method with a large body of approaches in the widely used settings on the MNIST, SVHN and CIFAR10 datasets given 100, 1,000 and 4,000 label\(^{\dagger}\) respectively. Table 1 summarizes the quantitative results. On all of the three datasets, Triple-GAN outperforms the strongest semi-supervised GANs (e.g., Improved-GAN) substantially and consistently, which demonstrate the benefit of compatible learning objective functions proposed in Triple-GAN. Further, Triple-GAN achieves state-of-the-art results on the more challenging SVHN and CIFAR10 datasets. Note that for a fair comparison with previous GANs, we do not leverage the extra unlabeled data on SVHN, while some baselines [30, 55] do.

Secondly, we evaluate our method with different numbers of labeled samples on the MNIST and Tiny ImageNet datasets for a systematical comparison with our main baseline, i.e., Improved-GAN [39], as shown in Table 2 and Table 3 respectively. On both datasets, Triple-GAN consistently outperforms Improved-GAN with a substantial margin, which again demonstrates the benefit of the proposed game-theoretical framework. In addition, we can see that Triple-GAN achieves a more significant improvement as the number of labeled data decreases, suggesting the effectiveness of the pseudo discriminative loss.

Thirdly, we investigate the reasons for the outstanding performance of Triple-GAN. We train a single semi-supervised classifier without $G$ and $D$ on SVHN as the baseline and get a result of more than 10% error rate, which shows that $G$ is important for semi-supervised learning even though the classifier can leverage unlabeled data directly. On CIFAR10, the baseline (a simple version of II model [50]) achieves 17.7% error rate. The smaller improvement is reasonable as CIFAR10 consists of complex natural images and hence $G$ is not as good as that on SVHN. In addition, we evaluate Triple-GAN without the pseudo discriminative loss on SVHN and it achieves about 7.8% error rate, which shows the advantages of the compatible objective functions (better than the 8.11% error rate of Improved-GAN) and the importance of the pseudo discriminative loss (worse than the complete Triple-GAN by 2%). Furthermore, Triple-GAN has a comparable convergence speed to Improved-GAN [39], as shown in Appendix D.

5. Our code is at https://github.com/zhenxuan00/triple-gan

6. We use these amounts of labels as default settings throughout the paper if not specified.
Finally, we systematically analyze the effect of the numbers of labeled and unlabeled data on the SVHN dataset in Fig. 2. As the number of labels increases, the performance of Triple-GAN is better and has a lower variance. However, if given a sufficient number of unlabeled data, i.e., more than 20,000, then the performance of Triple-GAN won’t get hurt too much by reducing the number of labels from 2,000 to 500. The results suggest the importance of unlabeled data to the final performance of the classifier in semi-supervised learning.

5.2.2 Generation

We demonstrate that Triple-GAN can learn good $G$ and $C$ simultaneously by generating samples in various ways with the exact models used in Sec. 5.2.1. For a fair comparison, the generative model and the number of labels are the same as the previous method [39].

In Fig. 3, we first compare the quality of the samples generated by Triple-GAN and Improved-GAN trained with feature matching [39], which works well for semi-supervised classification. We can see that Triple-GAN outperforms the baseline by generating fewer meaningless samples and clearer digits. Further, the baseline generates the same strange sample four times, labeled with the red rectangles in Fig. 3. The comparison on MNIST and CIFAR10 is presented in Appendix B. We also evaluate the samples on CIFAR10 quantitatively using the inception score metric following [39]. The value of Triple-GAN is $5.08 \pm 0.09$ while that of the Improved-GAN trained without minibatch discrimination [39] is $3.87 \pm 0.03$, which agrees with the visual quality results in Fig. 3. We then illustrate images generated by Triple-GAN in four specific classes on CIFAR10 in Fig. 4. See samples in more classes in Appendix C. In most cases, Triple-GAN is able to generate meaningful images with correct semantics.

Further, we show the ability of Triple-GAN to disentangle category and latent style features in Fig. 5. It can be seen that the latent factors in Triple-GAN encode meaningful physical factors, including the scale, intensity, orientation, color, and so on. Some GANs [18], [24], [41] can generate data class-conditionally given full labels, while Triple-GAN can do the same thing given much less label information.

Finally, we demonstrate the generalization capability of our Triple-GAN via class-conditional latent space interpolation. As presented in Fig. 6, Triple-GAN can transit smoothly from one sample to another with totally different visual factors without losing label semantics, which proves that Triple-GANs can learn meaningful latent spaces class-conditionally instead of overfitting to the training data, especially labeled data.

Overall, these classification and generation results in semi-supervised learning confirm that Triple-GAN avoid the incompatible convergence problems of $C$ and $G$ and can lead to a situation where both the generation and classification are good in semi-supervised learning.
(a) MNIST data  
(b) MNIST samples

(c) SVHN data  
(d) SVHN samples

(e) CIFAR10 data  
(f) CIFAR10 samples

(g) Tiny Imagenet data  
(h) Tiny Imagenet samples

Fig. 5. (a), (c), (e) and (h) are randomly selected labeled data of the MNIST, SVHN, CIFAR10 and Tiny Imagenet datasets, respectively. (b), (d), (f) and (i) are samples from Triple-GAN on the corresponding datasets, where each row shares the same label and each column shares the same latent variables.

5.3 The Extreme Low Data Regime

We evaluate Triple-GAN on a small subset of the SVHN, CIFAR10 and Tiny ImageNet datasets to show the ability of generation and classification in the extreme low data regime. We remove the losses on the unlabeled data as presented in Sec. 3.3 and keep all the networks the same as those in

TABLE 4

| Number of labels | 500  | 800  | 1,000 |
|------------------|------|------|-------|
| Improved-GAN     | 52.50| 32.71| 25.47 |
| DADA             | 50.27| 31.82| 24.17 |
| CNN baseline     | 48.54 (±2.60) | 30.02 (±2.17) | 22.72 (±2.39) |
| Triple-GAN (ours) | **45.49 (±1.24)** | **27.50 (±2.72)** | **21.44 (±1.95)** |

semi-supervised learning.

5.3.1 Classification

We compare Triple-GAN with two models systematically in the classification task. The main competitor is DADA [40], which is a conditional variant of Improved-GAN for the extreme low data regime setting. The CNN baseline denotes a classifier that shares the same architecture as the classifier
We now present samples generated by Triple-GAN in the worse because there is no unlabeled data available in the extreme low data regime. Nevertheless, Triple-GAN is still comparable to that of DADA [40].

Fig. 8 (b) presents the samples of Triple-GAN on the Tiny ImageNet dataset. With only 2,000 labeled data, Triple-GAN is able to generate meaningful samples given a specific class. Besides, the visual quality of the samples from Triple-GAN is better than that of DADA shown in Fig. 8 (a).

6 Conclusion

We present Triple-GAN, a unified game-theoretical framework for classification and class-conditional generation in both semi-supervised learning and the extreme low data regime. Triple-GAN involves three players—a generator, a discriminator and a classifier, whose objective functions consist of compatible adversarial losses and cross-entropy losses. With such objective functions, Triple-GAN addresses two main problems of the existing methods [37], [39], [40]. Specifically, Triple-GAN ensures that both the generator and the classifier can achieve their own optima (or equivalently both converge to the data distribution) respectively and enable the generator to sample data in a specific class. Our empirical results on the MNIST, SVHN, CIFAR10 and Tiny ImageNet datasets demonstrate that Triple-GAN can simultaneously achieve the state-of-the-art classification results among deep generative models and control the semantics of the generated samples, in two challenging settings with limited supervision.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (No. 2017YFA0700904), NSFC Projects (Nos. 61620106010, 61621136008), Beijing NSF Project (No. L172037), Beijing Academy of Artificial Intelligence (BAAI), Tiangong Institute for Intelligent Computing, the JP Morgan Faculty Research Program and the NVIDIA NVAIL Program with GPU/DGX Acceleration. C. Li was supported by the Chinese postdoctoral innovative talent support program and Shuimu Tsinghua Scholar Program.

### Table 5

| Datasets      | CIFAR10 | Tiny ImageNet | Tiny ImageNet |
|---------------|---------|---------------|---------------|
| Number of labels | 4,000   | 1,000         | 2,000         |
| DADA [40]     | 33.57 ±0.46 | 41.13 ±4.09   | 29.73 ±2.02   |
| CNN baseline  | 46.10 ±3.18 | 33.75 ±0.25   | 24.92 ±1.12   |
| Triple-GAN (ours) | 32.73 ±0.86 | 27.75 ±1.09   | 20.67 ±1.84   |
[53] G. K. Dziugaite, D. M. Roy, and Z. Ghahramani, “Training generative neural networks via maximum mean discrepancy optimization,” arXiv preprint arXiv:1505.03906, 2015.

[54] K. Gregor, I. Danihelka, A. Mnih, C. Blundell, and D. Wierstra, “Deep autoregressive networks,” arXiv preprint arXiv:1310.8499, 2013.

[55] L. Theis, A. van den Oord, and M. Bethge, “A note on the evaluation of generative models,” arXiv preprint arXiv:1511.01844, 2015.

[56] L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther, “Auxiliary deep generative models,” arXiv preprint arXiv:1602.05473, 2016.

[57] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. R. Salakhutdinov, “Good semi-supervised learning that requires a bad gan,” in Advances in neural information processing systems, 2017, pp. 6510–6520.

[58] L. Chongxuan, T. Xu, J. Zhu, and B. Zhang, “Triple generative adversarial nets,” in Advances in neural information processing systems, 2017, pp. 4088–4098.

[59] Theano Development Team, “Theano: A Python framework for fast computation of mathematical expressions,” arXiv e-prints, vol. abs/1605.02688, May 2016. [Online]. Available: http://arxiv.org/abs/1605.02688

[60] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[61] E. L. Denton, S. Chintala, and R. Fergus, “Deep generative image models using a Laplacian pyramid of adversarial networks,” in NIPS, 2015.

[62] T. Miyato, S.-i. Maeda, M. Koyama, K. Nakae, and S. Ishii, “Distributional smoothing with virtual adversarial training,” arXiv preprint arXiv:1507.00677, 2015.

Chongxuan Li received his BS degree from Institute for Interdisciplinary Information Sciences, Tsinghua University, China. He received his PhD degree in the Department of Computer Science and Technology at Tsinghua University, China, where he is currently a Postdoctor. His research interests are primarily on approximate Bayesian inference and deep generative models, with applications on unsupervised learning, semi-supervised learning and structured data.

Kun Xu received his BS degree from department of automation, Tsinghua University, China. He is currently working toward his PhD degree in the Department of Computer Science and Technology at Tsinghua University. His research interests are primarily on statistical machine learning, especially structured generative models for various learning tasks including unsupervised.

Bo Zhang graduated from the Department of Automatic Control, Tsinghua University, Beijing, China, in 1958. Currently, he is a Professor in the Department of Computer Science and Technology, Tsinghua University and a Fellow of Chinese Academy of Sciences, Beijing, China. His main interests are artificial intelligence, pattern recognition, neural networks, and intelligent control. He has published over 150 papers and four monographs in these fields.
APPENDIX A

DETAILED THEORETICAL ANALYSIS

Lemma 3.1. For any fixed C and G, the optimal discriminator D of the game defined by the objective function U(C, G, D) is

\[ D^*_C,G(x, y) = \frac{p(x, y)}{p(x, y) + \alpha p_c(x, y)}, \]

where \( p_c(x, y) := (1 - \alpha)p_g(x, y) + \alpha p_c(x, y) \) is a mixture distribution for \( \alpha \in (0, 1) \).

Proof. Given the classifier and generator, the objective function can be rewritten as

\[ U(C, G, D) = \iint p(x, y) \log D(x, y) dydx + (1 - \alpha) \iint p(y)p_z(z) \log (1 - D(G(z, y), y)) dydz + \alpha \iint p(x)p_c(y|x) \log (1 - D(x, y)) dydx = \iint p(x, y) \log D(x, y) dydx + \iint p_c(x, y) \log (1 - D(x, y)) dydx, \]

which achieves the maximum at \( \frac{p(x, y)}{p(x, y) + \alpha p_c(x, y)} \).

Lemma 3.2. The global minimum of V(C, G) is achieved if and only if \( p(c, y) = p_\alpha(x, y) \).

Proof. Given \( D^*_C,G \), we can reformulate the minimax game with value function U as:

\[ V(C, G) = \iint p(x, y) \log \frac{p(x, y)}{p(x, y) + \alpha p_c(x, y)} dydx + \iint p_c(x, y) \log \frac{p_c(x, y)}{p(x, y) + \alpha p_c(x, y)} dydx. \]

Following the proof in GAN, the V(C, G) can be rewritten as

\[ V(C, G) = -\log 4 + 2D_{JS}(p(x, y) || p_\alpha(x, y)), \]

where \( D_{JS} \) is the Jensen-Shannon divergence, which is always non-negative and the unique optimum is achieved if and only if \( p(x, y) = p_\alpha(x, y) = (1 - \alpha)p_g(x, y) + \alpha p_c(x, y) \).

Corollary 3.2.1. Given \( p(x, y) = p_\alpha(x, y) \), the marginal distributions of \( p(x, y) \), \( p_c(x, y) \) and \( p_g(x, y) \) are the same, i.e. \( p(x) = p_g(x) = p_c(x) \) and \( p(y) = p_g(y) = p_c(y) \).

Proof. Remember that \( p_g(x, y) = p(y)p_g(x|y) \) and \( p_c(x, y) = p(x)p_c(y|x) \). Take integral with respect to \( x \) on both sides of \( p(x, y) = p_\alpha(x, y) \) to get

\[ \int p(x, y) dx = (1 - \alpha) \int p_g(y) dy + \alpha \int p_c(y) dy, \]

which indicates that

\[ p(x) = (1 - \alpha)p(y) + \alpha p_c(y), \]

i.e. \( p_c(y) = p(y) = p_g(y) \).

Similarly, it can be shown that \( p_g(x) = p(x) = p_c(x) \) by taking integral with respect to \( y \).

Theorem 3.3. The equilibrium of \( \hat{U}(C, G, D) \) is achieved if and only if \( p(x, y) = p_g(x, y) = p_c(x, y) \).

Proof. According to the definition, we have \( \hat{U}(C, G, D) = U(C, G, D) + R_C \), where

\[ R_C = E_{p(x, y)}[-\log p_c(y|x)]. \]

We can rewrite \( R_C \) as:

\[ R_C = E_{p(x, y)}[-\log p_c(y|x)] = E_{p(x, y)}[\log p_g(x, y) - E_{p(x, y)}[\log p(y|x)]] = D(p(x, y)||p_c(x, y)) + E_{p(x, y)}[\mathbb{H}[p(y|x)]] , \]

where \( D(\cdot || \cdot) \) denotes the KL-divergence and \( \mathbb{H}[: \cdot] \) denotes the differential entropy. Because the second term is determined by the data distribution, minimizing \( R_C \) is equivalent to minimizing \( D(p(x, y)||p_c(x, y)) \), which is always non-negative and zero if and only if \( p(x, y) = p_c(x, y) \). Besides, the previous lemmas can also be applied to \( \hat{U}(C, G, D) \), which indicates that \( p(x, y) = p_\alpha(x, y) \) at the global equilibrium and concludes the proof.

Corollary 3.3.1. Adding any divergence (e.g. the KL divergence) between any two of the joint distributions or the conditional distributions or the marginal distributions to \( \hat{U} \) as an additional regularization to be minimized, will not change the global equilibrium of \( \hat{U} \).

Proof. This conclusion can be straightforwardly obtained by the global equilibrium point of \( \hat{U} \) and the definition of a statistical divergence between two distributions.

Pseudo discriminative loss: We now prove the equivalence of minimizing the pseudo discriminative loss \( R_P \) and the KL-divergence \( D(p_g(x, y)||p_c(x, y)) \) as follows:

\[ D(p_g(x, y)||p_c(x, y)) + E_{p_g(x)}[\mathbb{H}[p_g(y|x)]] - D(p_g(x)||p(x)) = \int p_g(x, y) \log \frac{p_g(x, y)}{p_c(x, y)} + p_g(x, y) \log \frac{1}{p_g(x|y)} dx dy - \int p_g(x) \log \frac{p_g(x)}{p(x)} dx \]

\[ = \int p_g(x, y) \log \frac{p_g(x, y)p(x)}{p_c(x, y)p_g(y|x)p_g(x)} dx dy = E_{p_g(x, y)}[-\log p_c(y|x)] = R_P. \]

Note that \( E_{p_g(x)}[\mathbb{H}[p_g(y|x)]] - D(p_g(x)||p(x)) \) is a constant with respect to \( \theta_c \). Therefore, if we only optimize the parameters in \( C \), these two losses are equivalent.

APPENDIX B

SAMPLE QUALITY COMPARISON WITH IMPROVED-GAN

We compare the samples generated from Triple-GAN and Improved-GAN on the MNIST and CIFAR10 datasets as in Fig. 9, where Triple-GAN shares the same architectures of the generator and the number of labeled data with the baseline. It can be seen that Triple-GAN outperforms the Improved-GAN trained with the feature matching criterion on generating indistinguishable samples.
(a) Feature Matching  (b) Triple-GAN

(c) Feature Matching  (d) Triple-GAN

Fig. 9. (a) and (c): Samples generated from Improved-GAN trained with feature matching on MNIST and CIFAR10 datasets. Strange patterns repeat on CIFAR10. (b) and (d): Samples generated from Triple-GAN.

## APPENDIX C
### CLASS-CONDITIONAL GENERATION ON CIFAR10

We show more semi-supervised class-conditional generation results on CIFAR10 in Fig. 10. Again, we can see that Triple-GAN can generate meaningful images in specific classes.

## APPENDIX D
### CONVERGENCE SPEED

Though Triple-GAN has one more network, its convergence speed is at least comparable to that of Improved-GAN, as presented in Fig. 11. Both the models are trained on the SVHN dataset with default settings and Triple-GAN can get good results in tens of epochs. The reason that the learning curve of Triple-GAN is oscillatory may be the larger variance of the gradients due to the presence of discrete variables. Also note that we apply the pseudo discriminative loss at the 200-th epoch and then the test error is reduced significantly in next 100 epochs.

## APPENDIX E
### DETAILED ARCHITECTURES

We list the detailed architectures of Triple-GAN on the MNIST, SVHN and CIFAR10 datasets in Table 6, Table 7, and Table 8, respectively. On Tiny ImageNet, we use the same architectures as on CIFAR10. Besides, all architectures are the same in both semi-supervised learning and the extreme low data regime on the same dataset.

Fig. 10. Samples from Triple-GAN in specific classes on the CIFAR10 dataset.

Fig. 11. The convergence speeds of Improved GAN and Triple-GAN on SVHN.
### TABLE 6
**MNIST**

| Classifier C | Discriminator D | Generator G |
|--------------|-----------------|-------------|
| **Input 28×28 Gray Image** | **Input 28×28 Gray Image, One-hot Class representation** | **Input Class y, Noise z** |
| 5×5 conv. 32 ReLU | MLP 1000 units, ReLU, gaussian noise, weight norm | MLP 500 units, softplus, batch norm |
| 2×2 max-pooling, 0.5 dropout | MLP 500 units, ReLU, gaussian noise, weight norm | MLP 500 units, softplus, batch norm |
| 3×3 conv. 64 ReLU | MLP 250 units, ReLU, gaussian noise, weight norm | MLP 784 units, sigmoid |
| 3×3 conv. 64 ReLU | MLP 250 units, ReLU, gaussian noise, weight norm | |
| 2×2 max-pooling, 0.5 dropout | MLP 250 units, ReLU, gaussian noise, weight norm | |
| 3×3 conv. 128 ReLU | MLP 1 unit, sigmoid, gaussian noise, weight norm | |
| 3×3 conv. 128 ReLU | Global pool | |
| 10-class Softmax | **Input Class y, Noise z** | |

### TABLE 7
**SVHN**

| Classifier C | Discriminator D | Generator G |
|--------------|-----------------|-------------|
| **Input: 32×32 Colored Image** | **Input: 32×32 colored image, class y** | **Input: Class y, Noise z** |
| 0.2 dropout | 0.2 dropout | MLP 8192 units, ReLU, batch norm |
| 3×3 conv. 128 ReLU, batch norm | 3×3 conv. 32, ReLU, weight norm | Reshape 512×4×4, ReLU, batch norm |
| 3×3 conv. 128 ReLU, batch norm | 3×3 conv. 32, ReLU, weight norm, stride 2 | 5×5 deconv. 256, stride 2, ReLU, batch norm |
| 3×3 conv. 128 ReLU, batch norm | 2×2 max-pooling, 0.5 dropout | 0.2 dropout |
| 3×3 conv. 256 ReLU, batch norm | 3×3 conv. 64, ReLU, weight norm | 5×5 deconv. 128, stride 2, ReLU, batch norm |
| 3×3 conv. 256 ReLU, batch norm | 3×3 conv. 64, ReLU, weight norm, stride 2 | |
| 3×3 conv. 256 ReLU, batch norm | 2×2 max-pooling, 0.5 dropout | 0.2 dropout |
| 3×3 conv. 512 ReLU, batch norm | 3×3 conv. 128, ReLU, weight norm | |
| NIN, 256 ReLU, batch norm | NIN, 128 ReLU, batch norm | |
| NIN, 128 ReLU, batch norm | Global pool | MLP 1 unit, sigmoid |
| 10-class Softmax, batch norm | **Input: Class y, Noise z** | sigmoid, weight norm |

### TABLE 8
**CIFAR10**

| Classifier C | Discriminator D | Generator G |
|--------------|-----------------|-------------|
| **Input: 32×32 Colored Image** | **Input: 32×32 colored image, class y** | **Input: Class y, Noise z** |
| Gaussian noise | MLP 1000 units, ReLU, weight norm | MLP 8192 units, ReLU, batch norm |
| 3×3 conv. 128 ReLU, weight norm | 3×3 conv. 32, ReLU, weight norm | Reshape 512×4×4, ReLU, batch norm |
| 3×3 conv. 128 ReLU, weight norm | 3×3 conv. 32, ReLU, weight norm, stride 2 | 5×5 deconv. 256, stride 2, ReLU, batch norm |
| 3×3 conv. 128 ReLU, weight norm | 2×2 max-pooling, 0.5 dropout | 0.2 dropout |
| 3×3 conv. 256 ReLU, weight norm | 3×3 conv. 64, ReLU, weight norm | |
| 3×3 conv. 256 ReLU, weight norm | 3×3 conv. 64, ReLU, weight norm, stride 2 | |
| 3×3 conv. 256 ReLU, weight norm | 2×2 max-pooling, 0.5 dropout | 0.2 dropout |
| 3×3 conv. 512 ReLU, weight norm | 3×3 conv. 128, ReLU, weight norm | |
| NIN, 256 ReLU, weight norm | NIN, 128 ReLU, weight norm | |
| NIN, 128 ReLU, weight norm | Global pool | MLP 1 unit, sigmoid, weight norm |
| Global pool | **Input: Class y, Noise z** | tanh, weight norm |
| 10-class Softmax, batch norm | | |