RankGAN: A Maximum Margin Ranking GAN for Generating Faces

Felix Juefei-Xu*[1][0000−0002−0857−8611], Rahul Dey*[2][0000−0002−3594−5122], Vishnu Naresh Boddeti[2][0000−0002−8918−9385], and Marios Savvides[1]

[1] Carnegie Mellon University, Pittsburgh, PA 15213, USA
[2] Michigan State University, East Lansing, MI 48824, USA

Abstract. We present a new stage-wise learning paradigm for training generative adversarial networks (GANs). The goal of our work is to progressively strengthen the discriminator and thus, the generators, with each subsequent stage without changing the network architecture. We call this proposed method the RankGAN. We first propose a margin-based loss for the GAN discriminator. We then extend it to a margin-based ranking loss to train the multiple stages of RankGAN. We focus on face images from the CelebA dataset in our work and show visual as well as quantitative improvements in face generation and completion tasks over other GAN approaches, including WGAN and LSGAN.

Keywords: Generative adversarial networks · Maximum margin ranking · Face generation.

1 Introduction

Generative modeling approaches can learn from the tremendous amount of data around us to obtain a compact descriptions of the data distribution. Generative models can provide meaningful insight about the physical world that human beings can perceive, insight that can be valuable for machine learning systems. Take visual perception for instance, in order to generate new instances, the generative models must search for intrinsic patterns in the vast amount of visual data and distill its essence. Such systems in turn can be leveraged by machines to improve their ability to understand, describe, and model the visual world.

Recently, three classes of algorithms have emerged as successful generative approaches to model the visual data in an unsupervised manner. *Variational autoencoders* (VAEs) [20] formalize the generative problem as a maximum log-likelihood based learning objective in the framework of probabilistic graphical models with latent variables. The learned latent space allows for efficient reconstruction of new instances. The VAEs are straightforward to train but at the cost of introducing potentially restrictive assumptions about the approximate posterior distribution. Also, their generated samples tend to be slightly blurry. *Autoregressive* models such as PixelRNN [24] and PixelCNN [27] get rid of the

* These authors contribute equally and should be considered co-first authors.

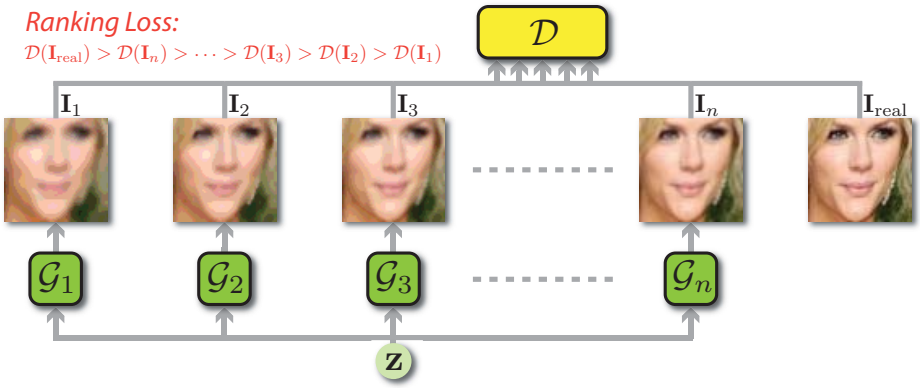

Fig. 1. The RankGAN framework consists of a discriminator that ranks the quality of the generated images from several stages of generators. The ranker guides the generators to learn the subtle nuances in the training data and progressively improve with each stage.

latent variables and instead directly model the conditional distribution of every individual pixel given the previous starting pixels. PixelRNN/CNN have a stable training process via softmax loss and currently give the best log likelihoods on the generated data, indicating high plausibility. However, they lack a latent code and are relatively inefficient during sampling.

Generative adversarial networks bypass maximum-likelihood learning by training a generator using adversarial feedback from a discriminator. Using a latent code, the generator tries to generate realistic-looking data in order to fool the discriminator, while the discriminator learns to classify them apart from the real training instances. This two-player minimax game is played until the Nash equilibrium where the discriminator is no longer able to distinguish real data from the fake ones. The GAN loss is based on a measure of distance between the two distributions as observed by the discriminator. GANs are known to generate highest quality of visual data by far in terms of sharpness and semantics.

Because of the nature of GAN training, the strength (or quality) of the generator, which is the desired end-product, depends directly on the strength of the discriminator. The stronger the discriminator is, the better the generator has to become in generating realistic looking images, and vice-versa. Although a lot of GAN variants have been proposed that try to achieve this by exploring different divergence measures between the real and fake distributions, there has not been much work dedicated to self-improvement of GAN, *i.e.*, progressively improving the GAN based on self-play with the previous versions of itself. One way to achieve this is by making the discriminator not just compare the real and fake samples, but also rank fake samples from various stages of the GAN, thus forcing it to get better in attending to the finer details of images. In this work, we propose a progressive training paradigm to train a GAN based on a maximum margin ranking criterion that improves GANs at later stages keeping the network

capacity same. Thus, our proposed approach is orthogonal to other progressive paradigms such as [18] which increase the network capacity to improve the GAN and the resolution of generated images in a stage-wise manner. We call our proposed method RankGAN.

Our contributions include (1) a margin-based loss function for training the discriminator in a GAN; (2) a self-improving training paradigm where GANs at later stages improve upon their earlier versions using a maximum-margin ranking loss (see Figure 1); and (3) a new way of measuring GAN quality based on image completion tasks.

1.1 Related Work

Since the introduction of Generative Adversarial Networks (GANs) [7], numerous variants of GAN have been proposed to improve upon it. The original GAN formulation suffers from practical problems such as vanishing gradients, mode collapse and training instability. To strive for a more stable GAN training, Zhao *et al*. proposed an energy-based GAN (EBGAN) [31] which views the discriminator as an energy function that assigns low energy to the regions near the data manifold and higher energy to other regions. The authors have shown one instantiation of EBGAN using an autoencoder architecture, with the energy being the reconstruction error. The boundary-seeking GAN (BGAN) [10] extended GANs for discrete data while improving training stability for continuous data. BGAN aims at generating samples that lie on the decision boundary of a current discriminator in training at each update. The hope is that a generator can be trained in this way to match a target distribution at the limit of a perfect discriminator. Nowozin *et al*. [23] showed that the generative-adversarial approach in GAN is a special case of an existing more general variational divergence estimation approach, and that any f-divergence can be used for training generative neural samplers. On these lines, least squares GAN (LSGAN) [22] adopts a least squares loss function for the discriminator, which is equivalent to minimizing the Pearson χ^2 divergence between the real and fake distributions, thus providing smoother gradients to the generator.

Perhaps the most seminal GAN-related work since the inception of the original GAN [7] idea is the Wasserstein GAN (WGAN) [3]. Efforts have been made to fully understand the training dynamics of GANs through theoretical analysis in [2] and [3], which leads to the creation of WGAN. By incorporating the smoother Wasserstein distance metric as the objective, as opposed to the KL or JS divergences, WGAN is able to overcome the problems of vanishing gradient and mode collapse. WGAN also made it possible to first train the discriminator till optimality and then gradually improve the generator making the training and balancing between the generator and the discriminator much easier. Moreover, the new loss function also correlates well with the visual quality of generated images, thus providing a good indicator for training progression.

On the other hand, numerous efforts have been made to improve the training and performance of GANs architecturally. Radford *et al*. proposed the DCGAN [25] architecture that utilized strided convolution and transposed-convolution to

improve the training stability and performance of GANs. The Laplacian GAN (LAPGAN) [5] is a sequential variant of the GAN model that generates images in a coarse-to-fine manner by generating and upsampling in multiple steps. Built upon the idea of sequential generation of images, the recurrent adversarial networks [12] has been proposed to let the recurrent network learn the optimal generation procedure by itself, as opposed to imposing a coarse-to-fine structure on the procedure. The stacked GAN [11] consists of a top-down stack of GANs, each trained to generate plausible lower-level representations, conditioned on higher-level representations. Discriminators are attached to each feature hierarchy to provide intermediate supervision. Each GAN of the stack is first trained independently, and then the stack is trained end-to-end. The generative multi-adversarial networks (GMAN) [6] extends the GANs to multiple discriminators that collectively scrutinize a fixed generator, thus forcing the generator to generate high fidelity samples. Layered recursive generative adversarial networks (LR-GAN) [29] generates images in a recursive fashion. First a background is generated, conditioned on which, the foreground is generated, along with a mask and an affine transformation that together define how the background and foreground should be composed to obtain a complete image.

The introspective adversarial networks (IAN) [4] proposes to hybridize the VAE and the GAN by leveraging the power of the adversarial objective while maintaining the efficient inference mechanism of the VAE.

Among the latest progress in GANs, Karras *et al*. [18] has the most impressive image generation results in terms of resolution and image quality. The key idea is to grow both the generator and discriminator progressively: starting from a low resolution, new layers that model increasingly fine details are added as the training progresses. This both speeds the training up and greatly stabilizes it, allowing us to produce images of unprecedented quality. On the contrary, we focus on improving the performance of GANs without increasing model capacity, making our work orthogonal to [18]. In the following sections, we will first discuss the background and motivation behind our work, followed by details of the proposed approach.

2 Background

We first provide a brief background of a few variants of GAN to motivate the maximum margin ranking based GAN proposed in this paper.

2.1 GAN and WGAN

The GAN framework [7] consists of two components, a Generator $\mathcal{G}_\theta(\mathbf{z}) : \mathbf{z} \rightarrow \mathbf{x}$ that maps a latent vector $\mathbf{z}$ drawn from a known prior $p_\mathbf{z}(\mathbf{z})$ to the data space and a Discriminator $\mathcal{D}_\omega(\mathbf{x}) : \mathbf{x} \rightarrow [0, 1]$ that maps a data sample (real or generated) to a likelihood value in $[0, 1]$. The generator $\mathcal{G}$ and the discriminator $\mathcal{D}$ play adversary to each other in a two-player minimax game while optimizing the

following GAN objective:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} V(\mathcal{G}, \mathcal{D}) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})}[\log(\mathcal{D}(\mathbf{x}))] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}))] \quad (1)$$

where $\mathbf{x}$ is a sample from the data distribution p_{data}. This objective function is designed to learn a generator $\mathcal{G}$ that minimizes the Jensen-Shannon divergence between the real and generated data distributions.

Many of the variants of GAN described in Section 1.1 differ in the objective function that is optimized to minimize the divergence between the real and generated data distributions. Wasserstein GAN [3, 2] has been proposed with the goal of addressing the problems of vanishing gradients and mode collapse in the original GAN. Instead of minimizing the cross-entropy loss, the discriminator in WGAN is optimized to minimize the Wasserstein-1 (Earth Movers') distance $W(\mathbb{P}_r, \mathbb{P}_g)$ between the real and generated distributions.

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Gamma(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\big[\|x - y\|\big] \quad (2)$$

where $\Gamma(\mathbb{P}_r, \mathbb{P}_g)$ is the set of all joint distributions $\gamma(x, y)$ whose marginals are $\mathbb{P}_r$ and $\mathbb{P}_g$ respectively. Given the intractability of finding the infimum in Eqn. (2), WGAN optimizes the dual objective given by the Kantorovich-Rubinstein duality [28] instead, which also constraints the discriminator to be a 1-Lipshichtz function.

2.2 Limitations with GANs and its Variants

An essential part of the adversarial game being played in a GAN is the discriminator, which is modeled as a two-class classifier. Thus, intuitively, the stronger the discriminator, the stronger (better) should be the generator. In the original GAN, stronger discriminator led to problems like vanishing gradients [2]. Variants like WGAN and LSGAN attempt to solve this problem by proposing new loss functions that represent different divergence measures. We illustrate this effect in Figure 2. The scores of the standard GAN model saturate and thus provide no useful gradients to the discriminator. The WGAN model has a constant gradient of one while RankGAN model (described in the next section) has a gradient that depends on the slope of the linear decision boundary. Therefore, from a classification loss perspective, RankGAN generalizes the loss of the WGAN critic. In practice, these variants don't easily reach convergence, partially because of limited network capacity and finite sample size of datasets. Loss functions for optimizing the discriminator are typically averaged over the entire dataset or a mini-batch of samples. As a result, the discriminator often keeps on increasing the margin between well-separated real and fake samples while struggling to classify the more difficult cases. Furthermore, we argue that a margin-based loss, as in the case of support vector machines, enables the discriminator to focus on the difficult cases once the easier ones have been well classified, making it a more effective classifier. Going one step further, by ranking several versions of the generator, the discriminator would more effectively

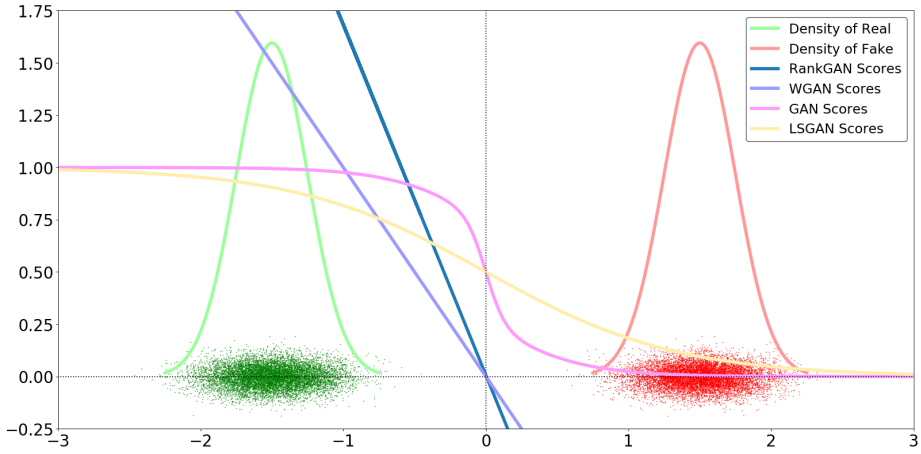

Fig. 2. Scores of the optimal discriminator for GAN, WGAN, LSGAN and RankGAN when learning to differentiate between two normal distributions. The GAN scores are saturated and hence results in vanishing gradients. The WGAN and RankGAN models do not suffer from this problem. See text for more details.

learn the subtle nuances in the training data. The supervision from such a strong discriminator would progressively improve the generators. This intuition forms the basic motivation behind our proposed approach.

3 Proposed Method: RankGAN

In this section, we describe our proposed GAN training framework - RankGAN. This model is designed to address some of the limitations of traditional GAN variants. RankGAN is a stage-wise GAN training paradigm which aims at improving the GAN convergence at each stage by ranking one version of GAN against previous versions without changing the network architecture (see Figure 3). The two basic aspects of our proposed approach are the following:

- We first adopt a margin based loss for the discriminator of the GAN, as opposed to the cross-entropy loss of the original GAN and the WGAN loss. We refer to this model as MarginGAN.
- We extend the margin-based loss into a margin-based ranking loss. This enables the discriminator to rank multiple stages of generators by comparing the scores of the generated samples to those of the real samples (see Figure 4 for an illustration). By applying certain constraints on the discriminator, which we will describe later, we can use this mechanism to steadily improve the discriminator at each stage, thereby improving the quality of generated samples.

The complete RankGAN training flow is shown in Algorithm 1. We now describe the various novelties in our approach.

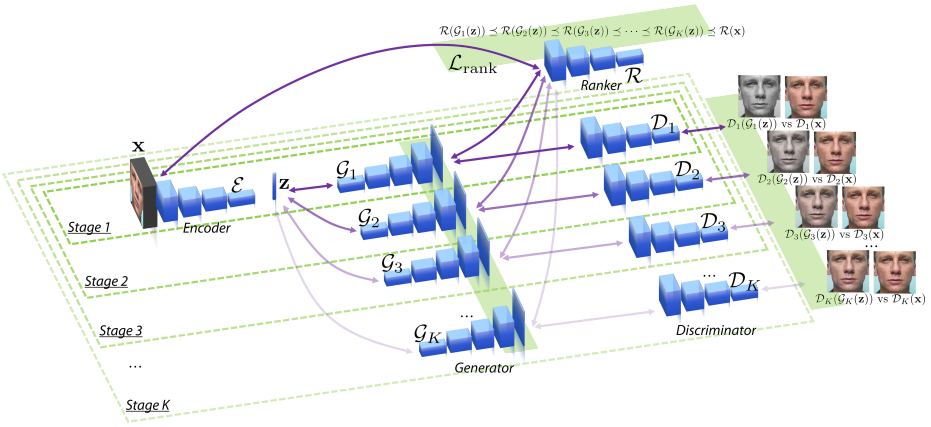

Fig. 3. Overall flowchart of the proposed RankGAN method. Our model consists of (1) An encoder that maps an image to a latent representation. (2) A series of generators that are learned in a stage-wise manner. (3) A series of discriminators that are learned to differentiate between the real and the generated data. (4) A ranker that ranks the real face image and the corresponding generated face images at each stage. In practice the discriminator and the ranker are combined into a single model.

3.1 Margin Loss

The intuition behind the MarginGAN loss is as follows. WGAN loss treats a gap of 10 or 1 equally and it tries to increase the gap even further. The MarginGAN loss will focus on increasing separation of examples with gap 1 and leave the samples with separation 10, which ensures a better discriminator, hence a better generator. The ϵ-margin loss is given by:

$$\mathcal{L}_{\text{margin}} = [\mathcal{D}_w(\mathcal{G}_\theta(\mathbf{z})) + \epsilon - \mathcal{D}_w(\mathbf{x})]_+ \tag{3}$$

where $[x]_+ = \max(0, x)$ is the hinge loss. The margin loss becomes equal to the WGAN loss when the margin $\epsilon \to \infty$, hence the generalization.

3.2 Ranking Loss

The ranking loss uses margin loss to train the generator of our GAN by ranking it against previous version of itself. For stage i discriminator $\mathcal{D}_i$ and generator $\mathcal{G}_i$, the ranking loss is given by:

$$\begin{aligned} \mathcal{L}_{\text{disc\_rank}} &= [\mathcal{D}_i(\mathcal{G}_i(\mathbf{z})) - \mathcal{D}_i(\mathcal{G}_{i-1}(\mathbf{z}))]_+ \\ \mathcal{L}_{\text{gen\_rank}} &= [\mathcal{D}_i(\mathbf{x}) - \mathcal{D}_i(\mathcal{G}_i(\mathbf{z}))]_+ \end{aligned} \tag{4}$$

The ranking losses for the discriminator and the generator are thus zero margin loss functions ($\epsilon \to 0$) where the discriminator $\mathcal{D}_i$ is trying to have a zero margin between $\mathcal{D}_i(\mathcal{G}_i(\mathbf{z}))$ and $\mathcal{D}_i(\mathcal{G}_{i-1}(\mathbf{z}))$, while the generator is trying to have zero

margin between $\mathcal{D}_i(\mathcal{G}_i(\mathbf{z}))$ and $\mathcal{D}_i(\mathbf{x})$ (see Figure 4). The discriminator is trying to push $\mathcal{D}_i(\mathcal{G}_i(\mathbf{z}))$ down to $\mathcal{D}_i(\mathcal{G}_{i-1}(\mathbf{z}))$ so that it gives the same score to the fake samples generated by stage i generator as those generated by stage $i-1$ generator. In other words, the discriminator is trying to become as good in detecting fake samples from $\mathcal{G}_i$ as it is in detecting fake samples from $\mathcal{G}_{i-1}$. This forces the generator to 'work harder' to fool the discriminator and give the same score to the fake samples $\mathcal{G}_i(\mathbf{z})$ as to the real samples. This adversarial game leads to the self-improvement of GAN with subsequent stages.

3.3 Encoder $\mathcal{E}$

Although RankGAN works even without an encoder, in practice, we have observed that adding an encoder improves the performance and training convergence of RankGAN considerably. This is because adding an encoder allows the discriminator to rank generated and real samples based on image quality and realisticity rather than identity. To obtain the encoder, we first train a VAE [20] in the zeroth stage. After the VAE is trained, the encoder is frozen and forms the first component of the RankGAN architecture (see Figure 3). During RankGAN training, the encoder takes the real image $\mathbf{x}$ and outputs a mean $\mu(\mathbf{x})$ and variance $\mathbf{\Sigma}(\mathbf{x})$ to sample the latent vector as $\mathbf{z} \sim \mathcal{N}(\mu(\mathbf{x}), \mathbf{\Sigma}(\mathbf{x}))$ which is used by the subsequent stage generators to generate fake samples for training. The VAE decoder can also be used as the zeroth stage generator.

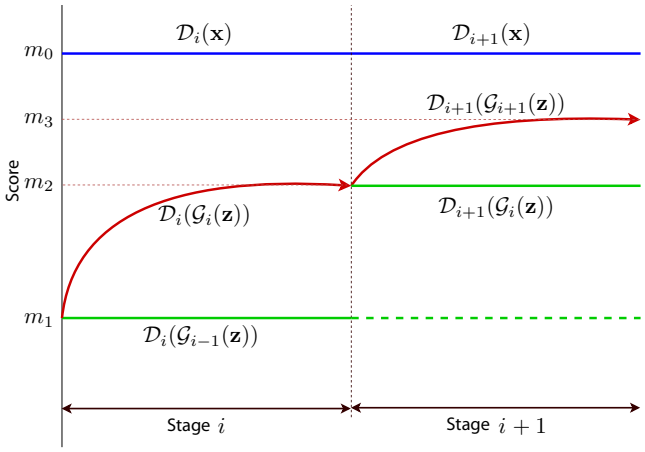

Fig. 4. RankGAN stage-wise training progression following $\mathcal{D}_i(\mathbf{x}) > \mathcal{D}_i(\mathcal{G}_i(\mathbf{z})) > \mathcal{D}_i(\mathcal{G}_{i-1}(\mathbf{z}))$. At stage i, $\mathcal{D}_i(\mathbf{x})$ and $\mathcal{D}_i(\mathcal{G}_{i-1}(\mathbf{z}))$ are clamped at the initial margins m_0 and m_1, respectively while $\mathcal{D}_i(\mathcal{G}_i(\mathbf{z}))$ slowly increases from m_1 to m_2 (point of Nash equilibrium) at the end of stage i. The same is repeated at stage $i+1$, where $\mathcal{D}_{i+1}(\mathbf{x})$ and $\mathcal{D}_{i+1}(\mathcal{G}_i(\mathbf{z}))$ are clamped at margins m_0 and m_2 respectively while $\mathcal{D}_{i+1}(\mathcal{G}_{i+1}(\mathbf{z}))$ slowly increases from m_2 to m_3 till convergence.

3.4 Discriminator Penalties

We enforce Lipschitz constrain on the discriminator using gradient penalty (GP) as proposed by Gulrajani *et al.* [8]. GP penalizes the norm of the gradient of the discriminator w.r.t. its input $\hat{\mathbf{x}} \sim \mathbb{P}_{\hat{\mathbf{x}}}$, which enforces a soft version of the constraint. The GP loss is given by:

$$\mathcal{L}_{\text{gp}} = \mathbb{E}_{\hat{\mathbf{x}} \sim \mathbb{P}_{\hat{\mathbf{x}}}} \left[(\|\nabla_{\hat{\mathbf{x}}} \mathcal{D}(\hat{\mathbf{x}})\|_2 - 1)^2 \right] \tag{5}$$

In addition, Eqn. (4) does not prevent the discriminator from cheating by letting $\mathcal{D}_i(\mathbf{x})$ and $\mathcal{D}_i(\mathcal{G}_{i-1}(\mathbf{z}))$ to simultaneously converge to the level of $\mathcal{D}_i(\mathcal{G}_i(\mathbf{z}))$ (blue and green curves converging towards the red curve in Figure 4), thereby defeating the purpose of training. To prevent this, we add a penalty term to the overall ranking loss given by:

$$\mathcal{L}_{\text{clamp}} = \left[m_i^{\text{high}} - \mathcal{D}_i(\mathbf{x}) \right]_+ + \left[\mathcal{D}_i(\mathcal{G}_{i-1}(\mathbf{z})) - m_i^{\text{low}} \right]_+ \tag{6}$$

where m_i^{high} and m_i^{low} are the high and low margins for stage-i RankGAN respectively. Thus, the clamping loss constraints the discriminator so as not to let $\mathcal{D}_i(\mathbf{x})$ go below m_i^{high} and $\mathcal{D}_{i-1}(\mathcal{G}_i(\mathbf{z}))$ go above m_i^{low}. We call this **Discriminator Clamping**. The overall discriminator loss thus becomes:

$$\mathcal{L}_{\text{disc}} = \mathcal{L}_{\text{disc\_rank}} + \lambda_{\text{gp}} \mathcal{L}_{\text{gp}} + \lambda_{\text{clamp}} \mathcal{L}_{\text{clamp}} \tag{7}$$

In our experiments, we find $\lambda_{\text{gp}} = 10$ and $\lambda_{\text{clamp}} = 1000$ to give good results.

4 Experiments

In this section, we describe our experiments evaluating the effectiveness of the RankGAN against traditional GAN variants *i.e.*, WGAN and LSGAN. For this purpose, we trained the RankGAN, WGAN and LSGAN models on face images and evaluated their performance on face generation and face completion tasks. Due to space limit, we will omit some implementation details in the paper. Full implementation details will be made publicly available.

4.1 Database and Metrics

We use the **CelebA** dataset [21] which is a large-scale face attributes dataset with more than 200K celebrity images covering large pose variations and background clutter. The face images are pre-processed and aligned into an image size of 64×64 while keeping a 90-10 training-testing split.

To compare the performance of RankGAN and other GAN variants quantitatively, we computed several metrices including Inception Score [26] and Fréchet Inception distance (FID) [9]. Although, Inception score has rarely been used to evaluate face generation models before, we argue that since it is based on sample entropy, it will favor sharper and more feature-full images. The FID, on the other hand, captures the similarity of the generated images to the real ones, thus capturing their realisticity and fidelity.

Algorithm 1: RankGAN Training

$\alpha_{\mathcal{D}}, \alpha_{\mathcal{G}} \leftarrow 5e-5, \alpha_{\mathcal{E}} \leftarrow 1e-4$;
for $i = 1$ ... *nstages* **do**
  if $i = 1$ **then**
    train **VAE** with **Encoder** $\mathcal{E}$ and **Decoder** $\mathcal{G}_1$;
    train **Discriminator** $\mathcal{D}_1$ for 1 epoch using WGAN loss of Eqn. 5;
  else
    $j, k \leftarrow 0, 0$;
    initialize $\mathcal{D}_i \leftarrow \mathcal{D}_{i-1}$ and $\mathcal{G}_i \leftarrow \mathcal{G}_{i-1}$;
    freeze $\mathcal{D}_{i-1}$ and $\mathcal{G}_{i-1}$;
    compute $m_i^{\text{high}} = \mathbb{E}[\mathcal{D}_{i-1}(\mathbf{x}_{\text{val}})]$ and $m_i^{\text{low}} = \mathbb{E}[\mathcal{D}_{i-1}(\mathcal{G}_{i-1}(\mathbf{z}))]$;
    while $j <$ *nepochs* **do**
      while $k < 5$ **do**
        obtain real samples $\mathbf{x}$ and latent vectors $\mathbf{z} \sim \mathcal{E}(\mathbf{x})$;
        compute $\mathcal{L}_{\text{disc}}$ using Eqn. 7;
        optimize $\mathcal{D}_i$ using $AdamOptimizer(\alpha_{\mathcal{D}}, \beta_1 = 0, \beta_2 = 0.99)$;
        $j \leftarrow j + 1$, $k \leftarrow k + 1$
      end
      compute $\mathcal{L}_{\text{gen}}$ using Eqn. 4;
      optimize $\mathcal{G}_i$ using $AdamOptimizer(\alpha_{\mathcal{G}}, \beta_1 = 0, \beta_2 = 0.99)$;
      $k \leftarrow 0$
    end
  end
end

4.2 Evaluations on Face Generation Tasks

For all the experiments presented in this paper, we use the same network architecture based on the one used in [18]. Both the discriminators and generators are optimized using the Adam optimizer [19] with $\beta_1 = 0.0$ and $\beta_2 = 0.99$ and a learning rate of $5e-5$. The criterion to end a stage is based on the convergence of that particular stage and is determined empirically. In practice, we terminate a stage when either the discriminator gap stabilizes for 10-20 epochs or at least 200 stage-epochs are finished, whichever is earlier. Lastly, no data augmentation was used for any of our experiments.

Figure 5 shows the visual progression of Open-Set face generation results from various stages in RankGAN when the latent vector $\mathbf{z}$ is obtained by passing the input faces through the encoder $\mathcal{E}$. Figure 6 shows the visual progression of face generation results when the latent vectors $\mathbf{z}$'s are randomly generated without the encoder $\mathcal{E}$. In both the cases, we can clearly see that as the stage progresses, RankGAN is able to generate sharper face images which are visually more appealing.

Quantitative results are consolidated in Table 1 with FID (the lower the better) and Inception score (the higher the better). As can be seen, as the training progresses from stage-1 to stage-3, the trend conforms with the visual results where stage-3 yields the highest Inception score and the lowest FID.

Table 1. Quantitative results for face image generation with and without the encoder.

	With Encoder		Without Encoder	
	FID	Inception Score	FID	Inception Score
Real	N/A	2.51	N/A	2.51
Stage-1	122.17	1.54	140.45	1.54
Stage-2	60.45	1.78	75.53	1.75
Stage-3	**46.01**	**1.89**	**63.34**	**1.91**

(a) Input faces. (b) Stage 1 generated faces, Open Set.

(c) Stage 2 generated faces, Open Set. (d) Stage 3 generated faces, Open Set.

Fig. 5. Face generation with RankGAN. Latent vectors $\mathbf{z}$'s are obtained by passing the input faces through the encoder $\mathcal{E}$.

(a) Stage 1, Open Set. (b) Stage 2, Open Set. (c) Stage 3, Open Set.

Fig. 6. Face generation with RankGAN. Latent vectors $\mathbf{z}$'s are randomly generated w/o encoder $\mathcal{E}$.

4.3 Evaluations on Face Completion Tasks

A good generative model should perform well on missing data problems. Motivated by this argument, we propose to use image completion as a quality measure for GAN models. In short, the quality of the GAN models can be quantitatively measured by the image completion fidelity, in terms of PSNR, SSIM and other metrics. Traditional shallow methods [16, 17] have shown some promising results but still struggle when dealing with face variations. Deep learning methods based

Fig. 7. Interpolation between two latent vectors which are obtained by passing the input faces through the encoder $\mathcal{E}$. The 3 rows within each montage correspond to Stage 1, 2, and 3 in RankGAN.

Fig. 8. Interpolation between two latent vectors that are randomly selected (without the encoder $\mathcal{E}$) from a unit normal distribution. The 3 rows within each montage correspond to Stage 1, 2, and 3 in RankGAN.

on GANs are expected to handle image variations much more effectively. To take on the image completion task, we need to utilize both the $\mathcal{G}$ and $\mathcal{D}$ from the RankGAN and the baselines WGAN and LSGAN, pre-trained with uncorrupted data. After training, $\mathcal{G}$ is able to embed the images from p_{data} onto some non-linear manifold of $\mathbf{z}$. An image that is not from p_{data} (*e.g.*, with missing pixels) should not lie on the learned manifold. We seek to recover the image $\hat{\mathbf{y}}$ on the manifold "closest" to the corrupted image $\mathbf{y}$ as the image completion result. To quantify the "closest" mapping from $\mathbf{y}$ to the reconstruction, we define a function consisting of contextual and perceptual losses [30]. The **contextual loss** measures the fidelity between the reconstructed image portion and the uncorrupted image portion, and is defined as:

$$\mathcal{L}_{\text{contextual}}(\mathbf{z}) = \|\mathbf{M} \odot \mathcal{G}(\mathbf{z}) - \mathbf{M} \odot \mathbf{y}\|_1 \tag{8}$$

where $\mathbf{M}$ is the binary mask of the uncorrupted region and $\odot$ denotes the Hadamard product. The **perceptual loss** encourages the reconstructed image to be similar to the samples drawn from the training set (true distribution p_{data}). This is achieved by updating $\mathbf{z}$ to fool $\mathcal{D}$, or equivalently by maximizing $\mathcal{D}(\mathcal{G}(\mathbf{z}))$. As a result, $\mathcal{D}$ will predict $\mathcal{G}(\mathbf{z})$ to be from the real data with a high probability.

$$\mathcal{L}_{\text{perceptual}}(\mathbf{z}) = -\mathcal{D}(\mathcal{G}(\mathbf{z})) \tag{9}$$

Thus, $\mathbf{z}$ can be updated, using backpropagation, to lie closest to the corrupted image in the latent representation space by optimizing the objective function:

$$\hat{\mathbf{z}} = \arg\min_{\mathbf{z}} (\mathcal{L}_{\text{contextual}}(\mathbf{z}) + \lambda \mathcal{L}_{\text{perceptual}}(\mathbf{z})) \tag{10}$$

where λ (set to 10 in our experiments) is a weighting parameter. After finding the optimal solution $\hat{\mathbf{z}}$, the reconstructed image $\mathbf{y}_{\text{completed}}$ can be obtained by:

$$\mathbf{y}_{\text{completed}} = \mathbf{M} \odot \mathbf{y} + (1 - \mathbf{M}) \odot \mathcal{G}(\hat{\mathbf{z}}) \tag{11}$$

Table 2. Data: CelebA, Mask: Center Large

	FID	Inception	PSNR	SSIM	OpenFace (AUC)	PittPatt (AUC)
Original	N/A	2.3286	N/A	N/A	1.0000 (0.9965)	19.6092 (0.9109)
Stage-1	27.09	2.1524	22.76	0.7405	0.6726 (0.9724)	10.2502 (0.7134)
Stage-2	23.69	2.1949	21.87	0.7267	0.6771 (0.9573)	9.9718 (0.8214)
Stage-3	27.31	**2.2846**	**23.30**	**0.7493**	**0.6789 (0.9749)**	**10.4102 (0.7922)**
WGAN	**17.03**	2.2771	23.26	0.7362	0.5554 (0.9156)	8.1031 (0.7373)
LSGAN	23.93	2.2636	23.11	0.7361	0.6676 (0.9659)	10.1482 (0.7154)

Metrics: In addition to the FID and Inception Score, we used metrics such as PSNR [14], SSIM, OpenFace [1] feature distance under normalized cosine similarity (NCS) [13] and PittPatt face matching score [15] to measure fidelity between the original and reconstructed face images. The last two are off-the-shelf face matchers that can be used to examine the similarity between pairs of face images. For these two matchers, we also obtain the area under the ROC curves (AUC) score as an auxiliary metric.

Occlusion Masks: We carried out face completion experiments on four types of facial masks, which we termed as: 'Center Small', 'Center Large', 'Periocular Small', and 'Periocular Large'.

Open-Set: It is important to note that all of our experiments are carried out in an Open-Set fashion, *i.e.*, none of the images and subjects were seen during training. This is of course a more challenging setting than Closed-Set and reflects the generalization performance of these models.

Discussion: Due to lack of space, we only show results based on the Center Large mask in the main paper (more qualitative and quantitative results can be found in the supplementary). These results have been summarized in Table 2 and can be visualized in Figure 9. As can be seen in Table 2, RankGAN Stage-3 outperforms all other baselines in all metrics except FID. The lower FID value for WGAN can be attributed to the fact that FID captures distance between two curves and is, in a way, similar to the Wasserstein distance that is minimized in the case of WGAN. The Stage-3 images appear to be both sharp (as measured by the Inception Score) as well as fidelity-preserving as compared to the original images (as measured by identity matching metrics). All the four identity-based metrics, PSNR, SSIM, OpenFace scores, and PittPatt scores are higher for Stage-3 of RankGAN. This is due to the fact that our formulation enforces identity-preservation through the encoder and the ranking loss.

5 Conclusions

In this work, we introduced a new loss function to train GANs - the margin loss, that leads to a better discriminator and in turn a better generator. We

(a) Original faces. (b) Masked faces. ('Center Large')

(c) WGAN completion, Open Set. (d) LSGAN completion, Open Set.

(e) Stage 1 completion, Open Set. (f) Stage 2 completion, Open Set.

(g) Stage 3 completion, Open Set.

Fig. 9. Best completion results with RankGAN on CelebA, 'Center Large' mask.

then extended the margin loss to a margin-based ranking loss and evolved a new multi-stage GAN training paradigm that progressively strengthens both the discriminator and the generator. We also proposed a new way of measuring GAN quality based on image completion tasks. We have seen both visual and quantitative improvements over the baselines WGAN and LS-GAN on face generation and completion tasks.

References

1. Amos, B., Bartosz, L., Satyanarayanan, M.: Openface: A general-purpose face recognition library with mobile applications. Tech. rep., CMU-CS-16-118, CMU School of Computer Science (2016)
2. Arjovsky, M., Bottou, L.: Towards principled methods for training generative adversarial networks. ICLR (under review) (2017)
3. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein GAN. arXiv preprint arXiv:1701.07875 (2017)
4. Brock, A., Lim, T., Ritchie, J., Weston, N.: Neural photo editing with introspective adversarial networks. arXiv preprint arXiv:1609.07093 (2016)
5. Denton, E.L., Chintala, S., Fergus, R., et al.: Deep generative image models using a laplacian pyramid of adversarial networks. In: NIPS. pp. 1486–1494 (2015)
6. Durugkar, I., Gemp, I., Mahadevan, S.: Generative multi-adversarial networks. arXiv preprint arXiv:1611.01673 (2016)
7. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS. pp. 2672–2680 (2014)
8. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: NIPS. pp. 5769–5779 (2017)
9. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Klambauer, G., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a nash equilibrium. arXiv preprint arXiv:1706.08500 (2017)
10. Hjelm, R.D., Jacob, A.P., Che, T., Cho, K., Bengio, Y.: Boundary-seeking generative adversarial networks. arXiv preprint arXiv:1702.08431 (2017)
11. Huang, X., Li, Y., Poursaeed, O., Hopcroft, J., Belongie, S.: Stacked generative adversarial networks. arXiv preprint arXiv:1612.04357 (2016)
12. Im, D.J., Kim, C.D., Jiang, H., Memisevic, R.: Generating images with recurrent adversarial networks. arXiv preprint arXiv:1602.05110 (2016)
13. Juefei-Xu, F., Luu, K., Savvides, M.: Spartans: Single-sample Periocular-based Alignment-robust Recognition Technique Applied to Non-frontal Scenarios. IEEE TIP **24**(12), 4780–4795 (Dec 2015)
14. Juefei-Xu, F., Pal, D.K., Savvides, M.: NIR-VIS Heterogeneous Face Recognition via Cross-Spectral Joint Dictionary Learning and Reconstruction. In: CVPRW. pp. 141–150 (June 2015)
15. Juefei-Xu, F., Pal, D.K., Singh, K., Savvides, M.: A Preliminary Investigation on the Sensitivity of COTS Face Recognition Systems to Forensic Analyst-style Face Processing for Occlusions. In: CVPRW. pp. 25–33 (June 2015)
16. Juefei-Xu, F., Pal, D.K., Savvides, M.: Hallucinating the Full Face from the Periocular Region via Dimensionally Weighted K-SVD. In: CVPRW. pp. 1–8 (June 2014)
17. Juefei-Xu, F., Savvides, M.: Fastfood Dictionary Learning for Periocular-Based Full Face Hallucination. In: BTAS. pp. 1–8 (Sept 2016)
18. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196 (2017)
19. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
20. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)

21. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: ICCV (December 2015)
22. Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z.: Least squares generative adversarial networks. arXiv preprint arXiv:1611.04076 (2017)
23. Nowozin, S., Cseke, B., Tomioka, R.: f-gan: Training generative neural samplers using variational divergence minimization. arXiv preprint arXiv:1606.00709 (2016)
24. van den Oord, A., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. arXiv preprint arXiv:1601.06759 (2016)
25. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with dcgan. arXiv preprint arXiv:1511.06434 (2015)
26. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. In: NIPS. pp. 2226–2234 (2016)
27. van den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., Kavukcuoglu, K.: Conditional image generation with pixelcnn decoders. arXiv preprint arXiv:1606.05328 (2016)
28. Villani, C.: Optimal transport: old and new, vol. 338. Springer Science & Business Media (2008)
29. Yang, J., Kannan, A., Batra, B., Parikh, D.: Lr-gan - layered recursive generative adversarial networks for image generation. ICLR (under review) (2017)
30. Yeh, R., Chen, C., Lim, T.Y., Hasegawa-Johnson, M., Do, M.N.: Semantic image inpainting with perceptual and contextual losses. arXiv preprint arXiv:1607.07539 (2016)
31. Zhao, J., Mathieu, M., LeCun, Y.: Energy-based generative adversarial network. arXiv preprint arXiv:1609.03126 (2016)