Latent reweighting for General Adversarial Networks

DisorGANized : Barthélémy Brégeon, Kenta Vert, Nabil Arbel

November 2024

1 Introduction

Through this project, we aim to improve digit generation on the MNIST dataset using a Generative Adversarial Network (GAN) enhanced through latent reweighting, inspired by the approach proposed in [2]. Standard GANs often face challenges in generating distinct, high-quality digits across classes, as continuous sampling in the latent space can lead to low-quality, blended outputs. To address this, we apply latent reweighting, a technique that adjusts the GAN's sampling distribution by assigning importance weights in the latent space through an auxiliary network.

2 Latent reweighting

Latent reweighting is a method to improve a generative adversarial network after training. It derives from the idea of rejection sampling but the key difference is that the rejection happens within the latent space which reduces the computing cost of rejection.

To do this rejection sampling in the latent space, we can't use the discriminator network. We need to train a neural network w of importance weight. This function w is used to derive a new distribution from the initial latent space distribution γ. We call this new distribution γ^ϕ, ϕ being the parameters of w.

$$for \quad all \quad z \in \mathbb{R}^d, d\gamma^\phi(z) = w^\phi(z) d\gamma(z) \tag{1}$$

For γ^ϕ to be a distribution over $\mathbb{R}^d$, we need $\mathbb{E}_\gamma w = 1$ which we will enforce with a regularizing term in the loss when training w.

The goal is to train w so that when generating images by sampling in the latent space using the learned γ^ϕ distribution we get better results. The reason why we might get better results with reweighting is that often the images we want the generator to generate generally come from different classes which can be close but disconnected. In our case, it's handwritten numbers so we have 10 classes. The generator is trained to derive images for all these classes starting from continuous distribution, in our case gaussian, which can result in generating images "in between" classes. The distribution γ^ϕ trained can be more diverse than a gaussian or uniform distribution, non convex and give high probability to disconnected regions of the latent space which should help.

The main difference with the paper [2] is the pretrained network we used. The paper advises to use a WGAN, however looking at last week's presentation, and trying ourselves to train one, we found that it was too much trouble to use a WGAN. Instead we chose to use our vanilla GAN which yielded pretty good results (FID of around 30, precision of around 50 and recall at around 0.2). We tried to train a WGAN using the original's paper [1] but after 50 epochs the FID was over 250 and the precision and recall were way too low. We decided to not take too much time to get a working WGAN and to focus on our latent reweighting approach.

3 Training procedure

The network w is trained adversarialy just like the generator which means we also retrain the discriminator when training w. If we call α the parameters of the discriminator D the optimization scheme is :

$$\inf_{\phi\in\Phi}\sup_{\alpha\in\Lambda}\mathbb{E}\,D_\alpha(x)-\mathbb{E}\,w_\phi(z)D_\alpha(G(z)) \tag{2}$$

With z folowing the distribution γ.

The objective functions are a bit more complex. In practice we use gradient penalty to train the discriminator and regularizing terms to train w :

$$\sup_{\phi\in\Phi}\mathbb{E}\,w_\phi(z)(D_\alpha(G(z))-\Delta)-\lambda_1(\mathbb{E}\,w_\phi(z)-1)^2-\lambda_2\mathbb{E}\,\max(0,(w_\phi(z)-m))^2 \tag{3}$$

The first regularizing term is to ensure that γ_ϕ is a distribution over $\mathbb{R}^d$ as mentioned before. The second term is to prevent a form of mode collapse where the function w goes to zero almost everywhere and is very high at the points of projection of the real dataset in the latent space. It achieves this simply by penalizing values of w above the threshold m. If m is higher the distribution γ_ϕ will tend to concentrate more around those points of projection.

4 Generating procedures

Latent rejection sampling : Latent rejection sampling is the simplest of the three methods presented in the paper. We exploit the fact that because of the second regularization term in the loss, w is approximately capped by m. This means we can use $\mathbb{P}_a = w(z)/m$ as a probability of acceptance. Because w is normed by the first regularization term this also means that the acceptance rate is $1/m$. This method of rejection sampling aims at sampling from the learned γ_ϕ distribution.

Latent gradient ascent : The principle of latent gradient ascent is to do a few steps of gradient ascent in the latent space with respect to the function w. It is motivated by the fact that maximizing w should generate images closer to the training images. In general it is more expensive than rejection sampling as long as m isn't too high.

Latent gradient ascent with rejection sampling : This method is a mix of the first two. First we do the rejection sampling then we do a few gradient ascent steps in the latent space. This methods is a balance between the other two, the rejection step means we can achieve good results with fewer gradient ascent steps.

5 Training Parameters

5.1 Hyperparameter selection for Latent Reweighting

To achieve optimal results with our latent reweighting GAN model on the MNIST dataset, it was essential to carefully select and fine-tune the hyperparameters. Our approach to this fine-tuning was two-fold: initially guided by intuition and then refined to align with the findings in [2].

The hyperparameters optimized during our experiments included:

- **m** - the maximum value for the acceptance threshold, set to 3, balancing acceptance rates in latent rejection sampling.
- λ_1 - a regularization parameter to enforce that the importance weights define a valid distribution, ensuring $\mathbb{E}_\gamma w = 1$, set to 10.
- λ_2 - a regularization parameter to prevent mode collapse by penalizing values of w above a certain threshold, set to 3.

These hyperparameters were chosen based on their influence on sample quality and diversity, allowing our GAN model to improve upon the vanilla GAN baseline in generating clear and distinct digit images.

5.2 Precision/Recall analysis over training epochs

To evaluate the quality and diversity of our generated samples, we tracked the precision and recall metrics over the course of training epochs.

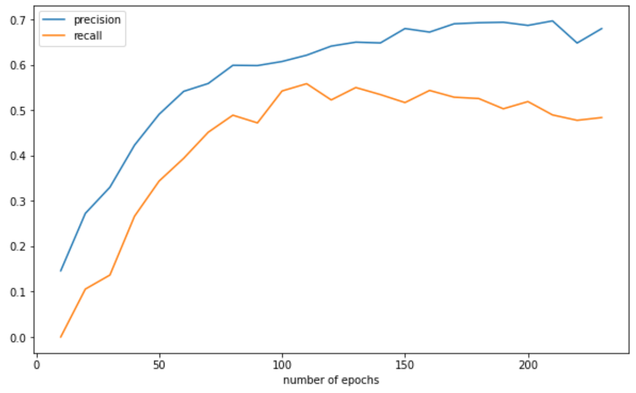

Figure 1: Precision and recall over the training epochs for the latent reweighting GAN

Initially, both precision and recall are low, indicating a lack of realism and diversity in early samples. As training progresses, precision steadily rises to about 0.7 after 150 epochs, reflecting improved sample quality. In contrast, recall increases quickly at first but stagnate around 0.5, highlighting limited diversity despite enhanced realism.

The gap between precision and recall reflects latent reweighting's focus on high-quality sample generation in specific latent regions. While this boosts precision, it limits recall, suggesting a trade-off where additional techniques may be needed to improve sample diversity.

6 Results

We can now discuss the results from our latent reweighting approach. The table below presents the performance of our approach, Fréchet Inception Distance (FID), Precision, and Recall metrics obtained for each method, all evaluated on the website platform. They were also trained less than in the previous graph because it took too much time to train them all for more than 200 epochs.

Method	**FID**	**Precision**	**Recall**
Rejection Sampling (RS)	33.61	0.50	0.21
Gradient Ascent (GA)	32.52	0.52	0.20
Rejection Sampling + Gradient Ascent (RS+GA)	32.81	0.50	0.21

Table 1: Results of Latent Reweighting with different optimization strategies.

The results show minimal variation in performance between the three methods. While Gradient Ascent (GA) achieved the lowest FID score of 32.52 and slightly higher precision of 0.52, all methods—Rejection Sampling (RS), Gradient Ascent, and their combination (RS+GA)—demonstrated comparable FID, precision, and recall values. This consistency suggests that the choice of optimization strategy has a limited impact on overall performance in this latent reweighting setup.

Notably, each approach yielded better performance in FID and precision than the baseline vanilla GAN, reflecting the relative effectiveness of latent reweighting. Future exploration of alternative methods or refined hyperparameters may further enhance results, potentially expanding the utility of latent reweighting.

7 Conclusion

In conclusion, our project successfully enhanced MNIST digit generation by applying latent reweighting to a GAN framework. This technique allowed us to refine the GAN's sampling distribution within the latent space, leading to improved sample quality and class distinctiveness compared to a vanilla GAN baseline. Our results highlight latent reweighting's ability to increase precision by focusing on specific latent regions, though this comes at a slight cost to diversity, as reflected in lower recall.

For future work, exploring alternative reweighting mechanisms or adaptive sampling techniques may help address the observed recall limitations, potentially improving sample diversity without compromising quality. Furthermore, investigating methods to refine latent distribution shaping, such as DRS, could actually further enhance the generation process and expand the applicability of latent reweighting in GAN frameworks.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein GAN". In: (2017). arXiv: `1701.07875 [stat.ML]`. URL: `https://arxiv.org/abs/1701.07875`.

[2] Thibaut Issenhuth et al. "Latent reweighting, an almost free improvement for GANs". In: *arXiv preprint arXiv:2110.09803* (2021).