COGS 185 FINAL PROJECT
WGAN-GP ON IMAGE GENERATION AND IMAGE RECONSTRUCTION USING VQ-VAE

Mingyang Yao
Department of Cognitive Science
University of California, San Diego
m5yao@ucsd.edu

ABSTRACT

The Generative Adversarial Network (GAN) is a class of classical deep-learning techniques in image generation, and Wasserstein GAN with Gradient Penalty (WGAN-GP) is one representative GAN that can provide stable training and alleviate mode collapse in traditional GAN. We report the WGAN-GP application on the customized Japanese Anime Scene dataset. To practice capture key features of scene images, we also experiment with the Vector Quantized Variational Autoencoders (VQ-VAE) to reconstruct images.

***K*eywords** WGAN · Image Generation

1 Introduction

GAN, proposed in 2014, is a minimax-based model with a discriminator and a generator. The mechanism is to let the generator and discriminator "compete" with each other. The generator will update its weight with the result from the discriminator on its generated samples, that is, trying to fool the discriminator, while the discriminator is trained with real samples in the training data and fake samples generated by the generator, that is, trying to not be fooled by the generator[1]. Then, during training, the generator and discriminator will improve alternatively and the generator will finally produce good generated samples.
However, traditional GAN experiences issues with unstable training and the risk of mode collapse (i.e. the generator takes the shortcut to fool the discriminator by learning only a small subset of characters in the training data) [2]. WGAN, though greatly solving the problem of stability during the training, does not tackle the problem of convergence. Therefore, we choose to use the improved WGAN, WGAN-GP, using gradient penalty to enforce a Lipschitz constraint instead of weight clipping [3].
In previous work, most image-generation tasks focused on generating images with a clear main object. Natural scenes in Anime style usually involve multiple components in the image that are correlated with each other and have complicated spatial structures. Therefore, we choose to explore image compression and generation with Anime scenes in this project. I collect high-quality anime scene images from multiple datasets to train a WGAN-GP that generates scenes without one main object and emphasizes the characters of a natural setting.
With the same dataset, we train a VQ-VAE, too to offer additional insights on the performance of our WGAN-GP as it uses a discrete representation and avoids the “posterior collapse” which means the latent factors are ignored by the decoder[7].

2 Method

2.1 Package and Hardware Used

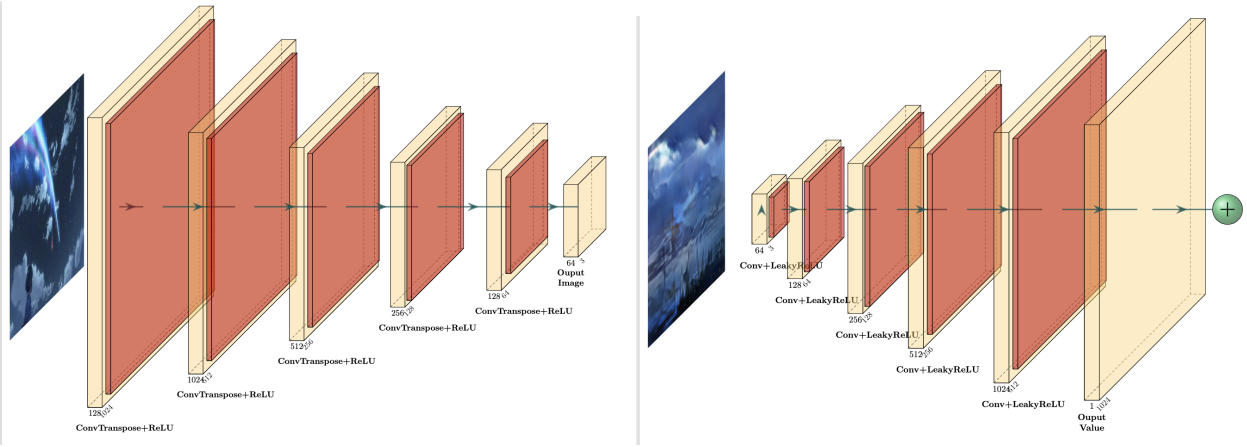
All model-related code is implemented with PyTorch and the DataHub as well as Google Colab are used for training.

2.2 Dataset

The Anime Scene Dataset used to train the WGAN-GP is a customized dataset that combines three publicly available datasets. We used Japanese Anime Scenes [4], Studio Ghibli landscape [5], and part of the dataset from the recent Anime Image-to-Image model, Scenimefy [6] (including parts from *5 Centimeters Per Second, A Gathering of Cats, Children Who Chase Lost Voices, Cross Road, The Garden of Words, Weathering with You* and *Your Name*). In total, we obtain 4972 1080 $\times$ 1080 images. To reduce the complexity, I compress the image to 512 $\times$ 512 for faster training. You can access the dataset from here

2.3 WGAN-GP Architecture and Hyperparameters

We trained the WGAN-GP to generate the image of resolution 128 $\times$ 128, so we take the 6-layer Convolution-based model. In the generator, we use 6 layers of ConvTranspose2D to let the latent factor broadcast to 1024 and finally converge to a 128$\times$128$\times$3 tensor. In the discriminator, we also use 6 layers of Conv2D with spectral normalization applied on each convolution layer to stabilize the training. For details about model setting and dimensions in each layer, please refer to the code.



(a) Generator Architecture

(b) Discriminator Architecture

We use RMSprop optimizer in PyTorch with different learning rates in different stages of the training and we take 128 latent factors as the size of input noise of the generator to capture the characteristics of training images. We also applied a batch norm in the generator to capture the overall characteristics. The batch size is switched between 256 and 128 depending on the loss in the training.
Sometimes, to ensure the discriminator has a better performance and is able to distinguish real and fake images, we need to train the discriminator more frequently. In this report, we call the number of times the discriminator is trained to be n_critic, and n_critic is varied depending on the loss of the discriminator during the training session, too.

2.4 VQ-VAE Architecture and Hyperparameters

We trained the VQ-VAE to reconstruct images of dimension 256$\times$256, and we also used a convolution-based model for the encoder and decoder. The encoder and decoder both contain five layers of 512 neurons. Input and output are both 3$\times$256$\times$256 tensor that can be converted to 256$\times$256 images. The encoder will first converge to weights with the length of the latent factor (a hyperparameter), and then, the decoder will reconstruct the image from the embedded latent factors. Due to model complexity, memory constraints of available GPU and to ensure the model's performance, I only make modifications to the size of the latent vector. We keep the batch size to 64 as it's the largest size we could use for the available memory. We use Adam optimizer with default parameters and a learning rate of 0.00005 and commitment_beta of 0.25 in the embedding. For further details of the model design, please refer to the code, and the following is a rough idea for the architecture.

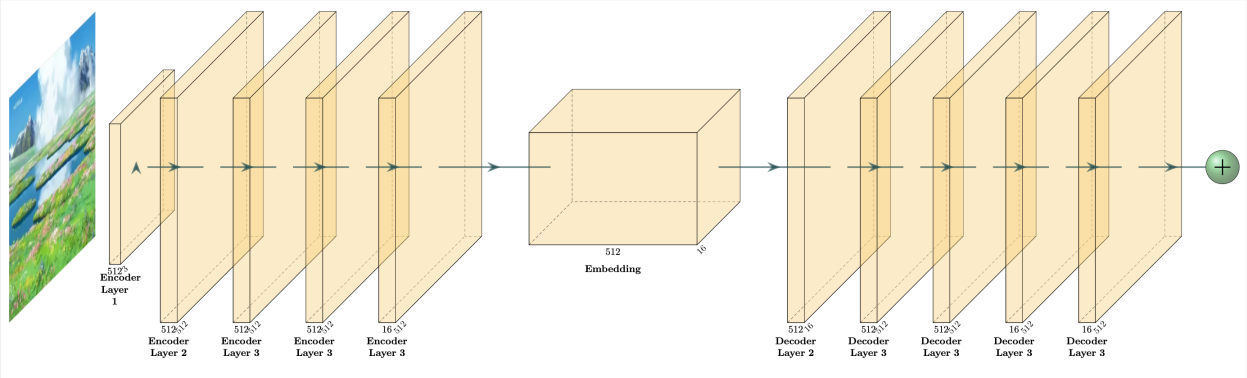


Figure 2: VQ-VAE Architecture

2.5 Loss Calculation

2.5.1 WGAN-GP

We use the gradient penalty in WGAN-GP to calculate loss, and it's calculated as follows.

Algorithm 1 Compute Gradient Penalty for WGAN-GP

Require: Discriminator D, Real Samples x_{real}, Fake Samples x_{fake}
1: Draw random value $\alpha \sim \text{Uniform}(0, 1)$
2: Interpolates: $x_{\text{interp}} = \alpha x_{\text{real}} + (1 - \alpha) x_{\text{fake}}$
3: Discriminator Output: $d_{\text{interp}} = D(x_{\text{interp}})$
4: Gradients: $\nabla_{x_{\text{interp}}} d_{\text{interp}}$
5: Gradient Norm: $\|\nabla_{x_{\text{interp}}} d_{\text{interp}}\|_2$
6: Gradient Penalty = $\lambda \left(\|\nabla_{x_{\text{interp}}} d_{\text{interp}}\|_2 - 1\right)^2$
7: **return** Gradient Penalty

2.5.2 VQ-VAE

We use Mean Square Error as loss criteria for evaluating the reconstruction. The MSE is calculated as follows.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2 \tag{1}$$

We also track the total loss and use it to guide the training process. The total loss is calculated as follows.

$$\text{Total Loss} = \text{Reconstruct Loss} + \text{Commitment Loss} \times \text{Commitment Beta} + \text{Codebook Loss} \tag{2}$$

2.6 Evaluation Techniques

2.6.1 WGAN-GP

We use the inception score to evaluate the generated images for their quality and diversity of WGAN-GP. For each checkpoint, we sample 1000 images to calculate the inception score with the inception v3 model. We also track the discriminator and generator loss during the training.

2.6.2 VQ-VAE

We calculate inception scores for both original and reconstructed images in the test set. By comparing two inception scores, we could see clues about the quality of reconstruction. With the standard variance of the scores, we could also see if the inception score of reconstructed images is within 1 z-score (1 standard deviation) of the score from the original images. We also track the codebook(Embedding module loss), reconstruction loss, and total loss.

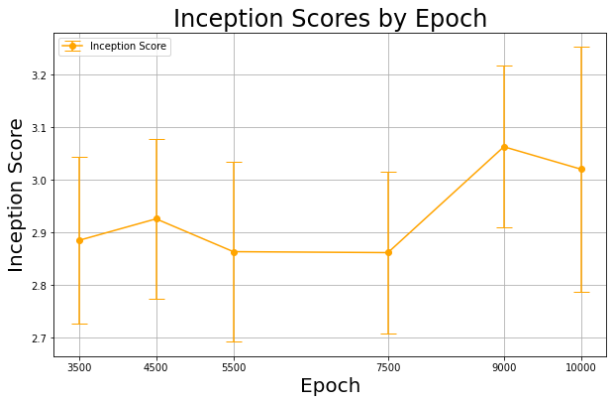
3 Experiment

3.1 WGAN-GP

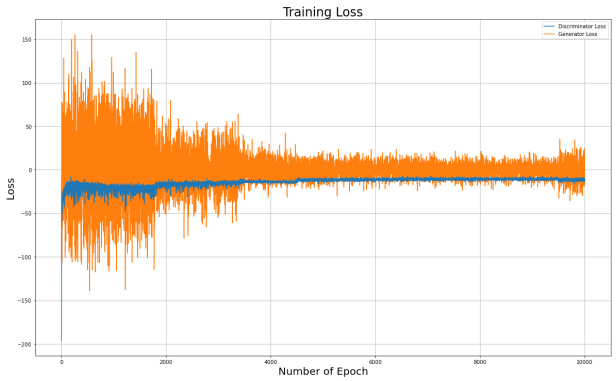
We trained the model with different hyperparameters, including learning rate, n_critic, and batch size. We used Xavier normal to initialize the model's weights, and we applied batch norm to every convolution layer to accelerate the convergence of the training and capture the overall structure of the training data instead of being shifted by extreme structure or colors in outlier image. and the following table displays the different hyperparameters in the training.

Training Schedule			
Epoch	N Critic	Learning Rate	Batch Size
0 - 2000	5	0.00005	256
2000 - 3000	5	0.00004	256
3000 - 4000	5	0.00001	128
4000 - 4500	3	0.00001	256
4500 - 5000	2	0.00002	128
5000 - 6500	1	0.00002	128
6500 - 9500	1	0.00001	128
9500 - 10000	1	0.00001	256

The discriminator and generator loss curve throughout the training period as well as the inception scores calculated with 1000 generated samples with checkpoints at different epochs are shown below to visualize the training process.



(a) Inception Score By Epoch



(b) Generator and Discriminator Loss

The Inception score experienced a slow improvement in the training from the graph. At the same time, the training loss of the discriminator experienced great oscillation before around 3800 epochs. After that, the discriminator reached a much more stable performance to guide the generator. The generator loss has a weak trend of becoming less negative, indicating that the generator is effectively learning.
We finally reached the inception score of 3.020332332344092 ± 0.23177706748839272. Below is a generated sample from the model.

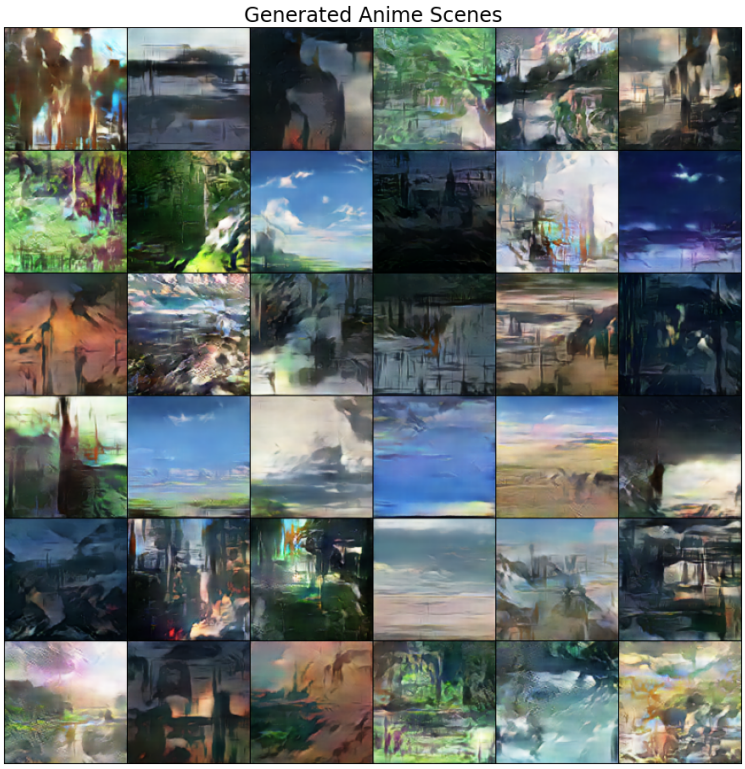


Figure 4: Generated Samples

3.2 VQ-VAE

In the experiment, I trained three models with latent factors of 16, 20, and 22. All other hyperparameters are kept the same to control variables as we want to investigate the influence of the size of the latent factors on the reconstruction quality. We trained all models for 50 epochs, and their loss curves are displayed below.

(a) Loss of 16 Latent Factor

(b) Loss of 20 Latent Factor

(c) Loss of 20 Latent Factor

Then, I calculated three pairs of inception scores of original and reconstructed images in the test set to evaluate the quality of reconstruction. Closer scores can indicate better reconstruction and capability of feature capture.

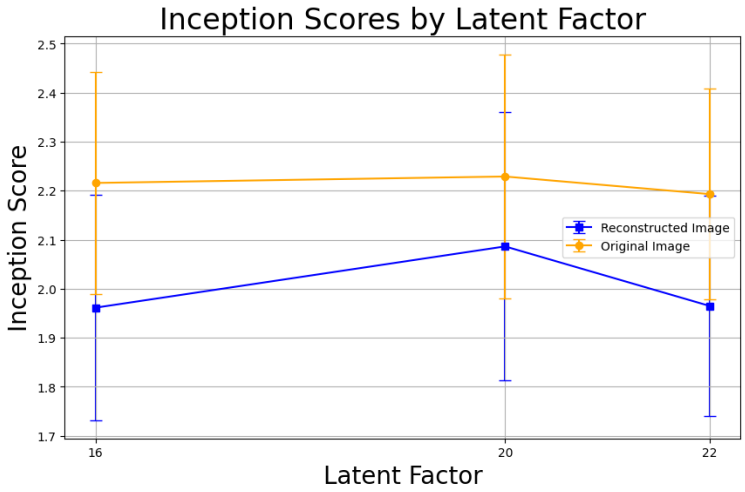


Figure 6: Inception Score by Latent Factor

By observing the inception score comparison between real and reconstructed images, we find that the model with latent factor 20 seems to give the best construction, because the reconstructed inception mean is closest to the original one and approximately 0.5 standard deviation below the original score, which indicates an outstanding reconstruction on the image quality. The following is an example of the comparison between the original and reconstructed images of this model.

(a) True Images

(b) Reconstructed Image

4 Conclusion and Discussion

4.1 WGAN-GP

By training a WGAN-GP for 10000 epochs in total, we have obtained a first-look good generated images. Many generated images have already captured the common elements in the Anime Scenes, such as sky, cloud, and grassland, especially the colors and hues that frequently appear in Japanese Anime (like purple, pink, and bright blue in the sky). In training, the loss of the discriminator gradually froze around a small interval, and the inferred cause is the n_critic keeps large for overly long periods that the discriminator was much better than the generator. Therefore, we gradually decrease the frequency we trained the discriminator as long as its loss is oscillating in a reasonable range. The use of gradient penalty also stabilizes the training and ensure there is no gradient vanishing or explosion during the training and the adversarial nature of GAN is shown well in the training process.

Despite a satisfactory output after training 10000 epoch, our model still fails to generate a reasonably clear picture that a 128×128 resolution should display. It's possibly due to insufficient training of the model. According to Lee's deep learning course, previous scientists obtained well-structured and lifelike Anime faces using GAN after 50000 epochs of training, and minor flaws still exist in a minority of generated samples [8]. Further, anime scenes have more diverse elements, subjects, and structures compared with anime faces. Therefore, with a sufficient amount of hardware and

time, we could train additional epochs to obtain a better result on the basis of our model.

4.2 VQ-VAE

Exploring different sizes of latent factors, we have trained an effective autoencoder that can reconstruct the original image in a surprising quality. By visual observation, the overall color, structure, and main objects (if exist) are almost perfectly reconstructed. The main shortcoming of the model is that the reconstructed images are slightly blurred and less bright than the ground-truth images. The possible reason is the avoidance of extreme value in the model as outputting extreme color may increase the risks of high loss. With more powerful hardware, we may experiment with different neural network architectures and larger latent factors to explore better autoencoders to capture information in the images. Overall, compared with the size of our encoded latent factors (less than 32), the quality of image compression and feature extraction is extraordinary.

5 Discussion

The VQ-VAE trained on the same dataset offers insights into our WGAN-GP and the inception score as a metric to evaluate generative models.

According to inception scores calculated with the original training images, the inception scores calculated with generated images by WGAN-GP are even higher for some checkpoints, and generated images outperform the reconstructed images according to the inception scores. It suggests that our generative model indeed captures the essential elements of Anime Scenes though the complex structure and relationships among elements still need further training or improvement with the model architecture.

However, this surprising inception score points out the potential bias or issues with inception score as a metric to measure generative models. In 2018, Barratt and Sharma [9] proposed the potential bias and issues with the inception score. Since the inception score is based on classification and the model is trained by the ImageNet dataset, when this model is applied to models trained with other datasets, the divergence of included labels of images makes the score biased, such as CIFAR-10 [9]. That is, the original classification model would not give valid results when it's required to recognize unseen objects. Furthermore, another problem with object-classification-based metrics is their emphasis on the identification of one or two main objects, which causes low entropy before the softmax layer. On the other hand, some images contain multiple objects without a main target, like anime scenes, natural scenes, and room layouts. For these images, in addition to containing classes that are not included in the training data of the inception model, multiple elements and complex structures result in a nature of high entropy, which is the opposite of the core idea of the inception score. Therefore, we may need more updated evaluation metrics for generative models.

This shortcoming is also supported by the score comparison between VQ-VAE and WGAN-GP. By visual inspection, the generated images are obviously worse than the quality of reconstructed images using autoencoder, but it received higher inception scores. The possible explanation is more monotonous color-blocks and simple structures in generated images are favored by the logic of inception score.

6 Code Availability

The code and dataset for this project are here: `https://github.com/AndyWeasley2004/WGAN-GP-and-VQ-VAE-with-Anime-Scenes`

7 Acknowledgement

The implementation of VQ-VAE used in this project is from a VAE tutorial. `https://github.com/Jackson-Kang/Pytorch-VAE-tutorial`

References

[1] Arjovsky, M., Chintala, S., & Bottou, L. (2017, December 6). Wasserstein Gan. arXiv.org. https://arxiv.org/abs/1701.07875

[2] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014, June 10). Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1406.2661

[3] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017, December 25). Improved training of Wasserstein Gans. arXiv.org. https://arxiv.org/abs/1704.00028

[4] Rhapsody. (2020, March 12). Japanese anime scenes. Kaggle. https://www.kaggle.com/datasets/weiwangk/japanese-anime-scenes

[5] Gomes, P. (2021, March 27). Studio ghibli landscape. Kaggle. https://www.kaggle.com/datasets/patrickgomes/studio-ghibli-landscape

[6] Jiang, Y., Jiang, L., Yang, S., & Loy, C. C. (2023, August 24). Scenimefy: Learning to craft anime scene via semi-supervised image-to-image translation. arXiv.org. https://arxiv.org/abs/2308.12968

[7] Oord, A. van den, Vinyals, O., & Kavukcuoglu, K. (2018). Neural Discrete Representation Learning. ArXiv:1711.00937 [Cs]. https://arxiv.org/abs/1711.00937

[8] Hung-yi Lee. (2021, April 9). 2021 (Generative Adversarial Network, GAN) () – . YouTube. https://www.youtube.com/watch?v=4OWp0wDu6Xw&list=PLJV_el3uVTsMhtt7_Y6sgTHGHp1Vb2P2J&index=14

[9] Barratt, S., & Sharma, R. (2018). A Note on the Inception Score. ArXiv (Cornell University). https://doi.org/10.48550/arxiv.1801.01973

[10] Kang, M. (2024, June 4). Jackson-Kang/Pytorch-VAE-tutorial. GitHub. https://github.com/Jackson-Kang/Pytorch-VAE-tutorial