1 Design Model

Since the performance of different models with different hyper-parameters need to be analyzed and compared, neural network structures of different models applied in this homework must be same for every datasets(in this case,only MNIST hands-writing and CIFAR10) when comparing the performance of different models. This can help us to understand the true influence of some hyper-parameters better.

There are some notes that I have to mention, before we applied the CNN to build up models, we have tried the MLP in VAE,GAN and WGAN, it worked well in all these models in MNIST hands-writing datasets but worked bad in CIFAR10 datasets. There are two reasons to illustrate this phenomenon. MNIST hands-writing datasets(28*28*1) is too simple to train, so it works well even in MLP structure models. Another reason is that MLP structure is so simple that it can not train the more complex datasets(CIFAR10,32*32*3).

Finally, we used deep convolutional structure to build up neural network layers. We first build up our models which can work well in CIFAR10 datasets because it is more complex and harder to train than MNIST hands-writing datasets. Therefore, our base models are built as the following sections illustrate.

1.1 Discriminator or Encoder

The structures of the Discriminator in GAN and WGAN or the structure of the Encoder in VAE are shown as Figure 1

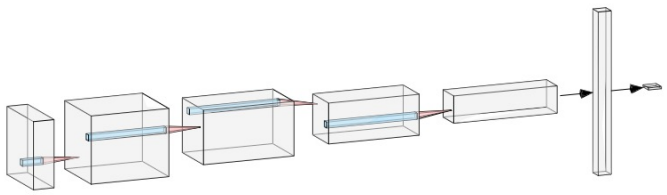


Figure 1: The Structure of Discriminator or Encoder.

The general description is: input images are fed into 4 convolutional layers with stride 2 and padding *same*, then are flatten into a big array before finally sized to the output shape by using $W \cdot X + b$.

1.2 Generator or Decoder

The structures of the Generators in GAN and WGAN or the structure of the Decoder in VAE are shown as Figure 2

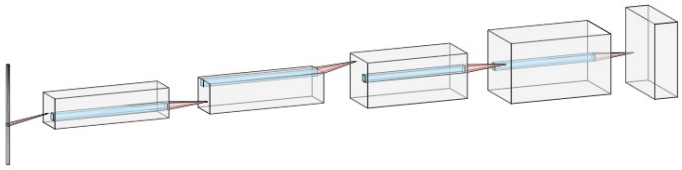


Figure 2: The Structure of Generator or Decoder.

The general description are: The input latent space is fed into 4 de-convolutional layers with stride 2 and padding *same*, with the final layer sizing it back to the same shape as the input image.

2 MNIST Dataset

In the following parts, we will first introduce the results and conclusions on the MNIST datasets, then we will go to the CIFAR-10 datasets in the second part.

In the following, 3 subsections are used for describing the performance of each model as the requirements in the assignment.

In each subsection, the influence of two hyper-parameters (latent size and the number of hidden layers) will be discussed separately.

2.1 VAE

2.1.1 Latent Size

First of all, let us take a look at generated images quality. The results are shown as Figure 3

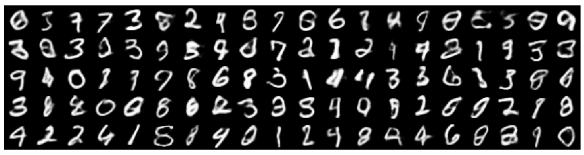


Figure 3: A random sample with final results.

These images of different training time steps are shown as Figure 4. In MNIST handswriting datasets, we used 500 epochs to train. There are five rows in each images result below. For the i-th row, we randomly choose one epoch results in epoch range from $(i -$

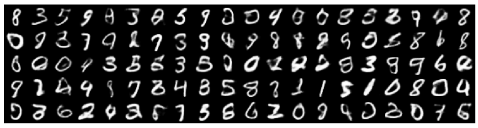
1) $*100$ to $i*100$, and we picked 20 images for each row. So the quality of images improves row by row because the model learned better and better with epoch going by.

To compare the influence of latent size, we fixed the number of hidden layers is 3, and then compare latent size 10,20,50,100 and 200.

(a) Results on latent size 10.

(b) Results on latent size 20.

(c) Results on latent size 50.



(d) Results on latent size 100.

Figure 4: Influence of the latent size on 5 models, with performance increasing row by row

As the figures show, after several thousand training steps, the model's performance becomes relatively promising.

In order to compare the training effort of the VAE model on different latent space sizes, a plot of generator loss function value is created and plotted as Figure 5

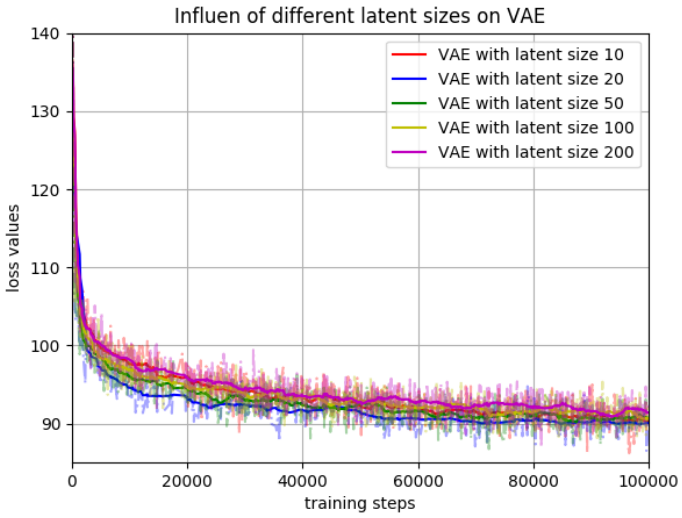


Figure 5: Different latent size.

A brief conclusion is drawn as follow,

- The curves show the performances of the model on different latent size are quite similar. One may say the latent size has very slight influence on VAE.
- One may spot that the performance reaches best when the latent size is 20. This means with latent size increasing to a certain value, the performance of the model reaches an optimal value, and it will not be enhanced further with latent size keeps going up.

2.1.2 Number of Hidden Layers

As before, the good quality results are shown as Figure 6, and increasing quality results are shown as Figure 7.

Figure 6: A random sample with final results.

(a) Results on 3 hidden layers

(b) Results on 5 hidden layers

(c) Results on 5 hidden layers

Figure 7: Influence of the number of hidden layers on 3 models, with performance increasing row by row

In this homework, we used 3 hidden layers as the base model, and selectively add some convolutional layers with stride $(1, 1)$ in the middle of some $(2, 2)$ strides convolutional layers.

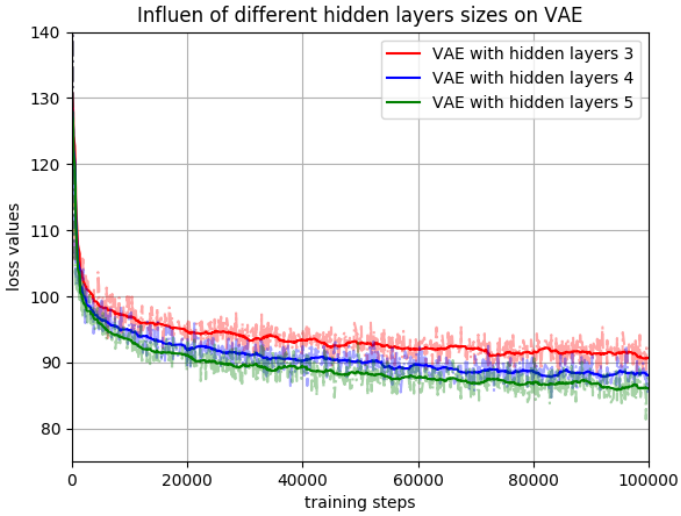


Figure 8: Different number of hidden layers.

In order to compare the training effort of the VAE model on different number of hidden layers, a plot of generator loss function value is created and plotted as Figure 8

A brief conclusion is drawn as follow,

- The loss value of VAE with hidden layers 5 is the lowest, which means VAE with hidden layers 5 learnt better than models with other hidden layers.
- This implies more layers can help the model learn better.

2.2 GAN

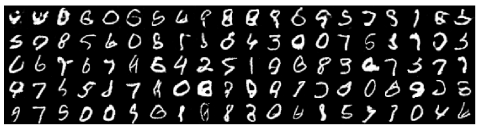
2.2.1 Latent Size

First of all, let us take a look at generated good results, which are shown as Figure 9, and generated images quality of different training time steps. The results are shown as Figure 10

Figure 9: A random sample with final results.

To compare the influence of latent size, we fixed the number of hidden layers is 3, and then compare latent size 10,20,50,100 and 200.

(a) Results on latent size 10.



(b) Results on latent size 20.

(c) Results on latent size 50.

(d) Results on latent size 100.

Figure 10: Influence of the latent size on 5 models, with performance increasing row by row

Intuitively, latent size 100 gets the better result than others. Similarly, too small or too large latent size will reduce the performance of GAN, there is the best result we can get when tuning latent size, and latent size is one of the key hyper-parameters which have to be tuned in real applications.

In order to compare the training effort of the GAN model on different latent space sizes, a plot of generator loss function value is created and plotted as Figure 11

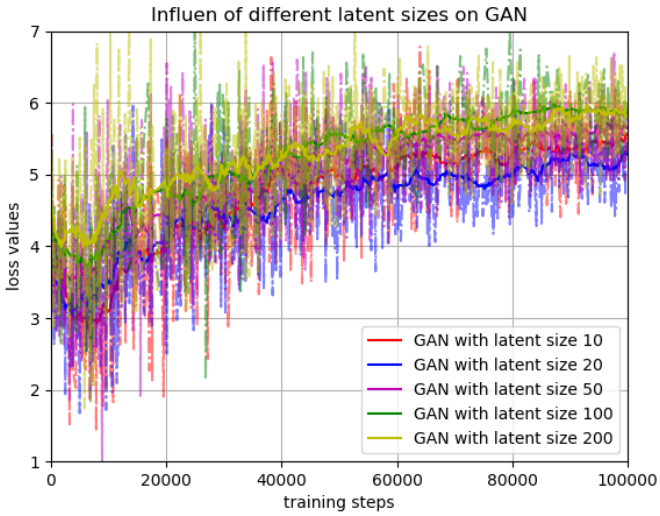


Figure 11: Different latent size.

A brief conclusion is drawn as follow,

- From the overall view, increasing the latent size of generator in GAN will increase the performance
- However, the plot shows the performance of latent size 20 is the worst(lowest loss value) and latent size 100 is the best(highest loss value) in this experiment. But the loss value is still increasing, meaning the models have not completely converged yet. Our epochs are only 500, it is not sufficient to say which latent size can generate best images finally.

2.2.2 Number of Hidden Layers

First of all, let us take a look at the good quality results, which are shown as Figure 12, and increasing quality results are shown as Figure 13.

To compare the influence of the number of hidden layers, we fixed the latent size is 100, and then compare the number of hidden layers 3,4 and 5.

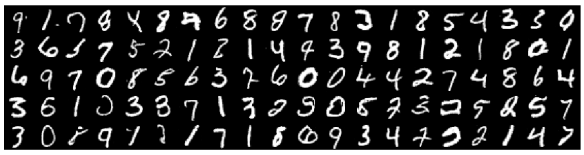


Figure 12: A random sample with final results.

(a) Results on 3 hidden layers

(b) Results on 4 hidden layers

(c) Results on 5 hidden layers

Figure 13: Influence of the number of hidden layers on 3 models, with performance increasing row by row

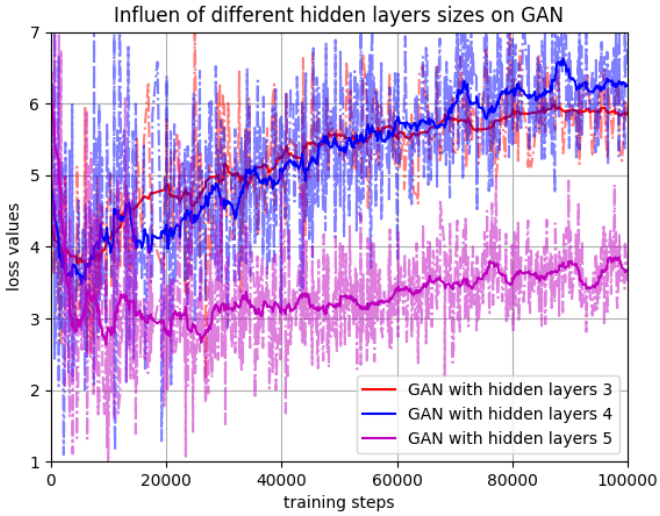


Figure 14: Different number of hidden layers.

A plot of generator loss function value is created and plotted as Figure 14
A brief conclusion is drawn as follow,

- From the figure, it is easy to tell the training efforts basically follow what we learnt from the lecture, that is, the JSD loss (represeted as generator loss) of GAN increases until it converges. This is the key feature that tells us whether the GAN is learning or not.
- One can notice that the overall performances of GAN on 3 and 4 layers are quite similar, but is worse on 5 layers. This may lead to the conclusion that with increasing the hidden layers of the GAN model, it will dampen the model performance. However, another explanation may also be corret: Since the model has more weights to train, it may take time to learn. But we only trained the model on 1000 epochs, it may not be sufficient to say "more layers, worse performance".
- Although the performances of GAN on 3 and 4 layers show some similarities, one can still find out that at the last few epochs, 4 layers GAN outperforms 3 layers model. This may imply that within certain number of layers, the final performance would be better when we increase the model complexity.

2.3 WGAN

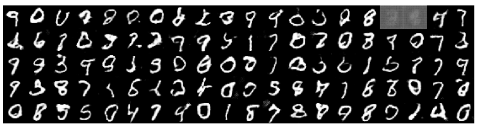
The difference of GAN and WGAN is cost function, and they use different cost function to find the optimal solution. So converting GAN to WGAN needs 3 operations:

- Remove sigmoid activation function of the last dense layer in discriminator
- Change loss function of both generator and discriminator
- Use weight clip when updating the weight or gradient penalty, which is called WGAN-GP.

2.3.1 Latent Size

First of all, let us take a look at the good quality results, which are shown as Figure 15 and increasing quality results are shown as Figure 16.

Figure 15: A random sample with final results.



(a) Results on latent size 10.

(b) Results on latent size 20.

(c) Results on latent size 50.

(d) Results on latent size 100.

Figure 16: Influence of the latent size on 5 models, with performance increasing row by row

Here, latent size 10 gets the better result than others, which is different to the best latent size of GAN, so for each kind of GAN model, there is one matching best latent size to reach the best performance.

In order to compare the training effort of the WGAN model on different latent space sizes, a plot of generator loss function value is created and plotted as Figure 17

A brief conclusion is drawn as follow,

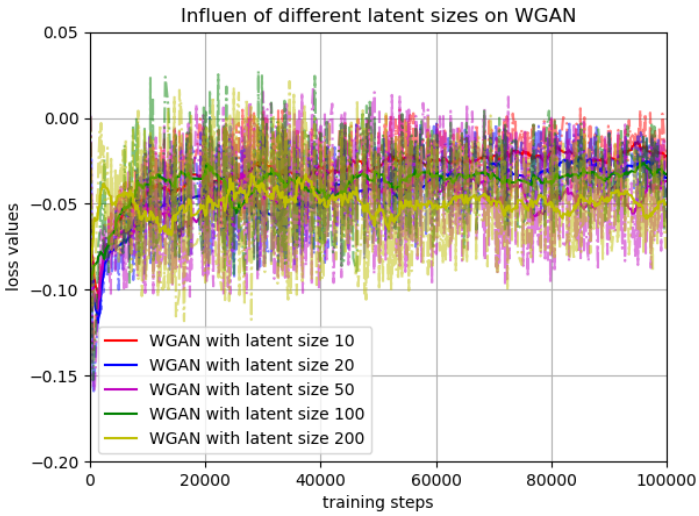


Figure 17: Different latent size.

- The overall training processes agree with the curve we learnt from the lecture, with the training step goes up, the EMD loss (represented as generator loss) decreases.
- It is not hard to find that with the latent size increasing, the loss value becomes greater. This can generally be concluded as, increasing the latent size on the WGAN model (training on MNIST datasets) can worsen the final performance.
- The WGAN is easier to converge than GAN, but the final performance of WGAN is worse than GAN.

2.3.2 Number of Hidden Layers

First of all, let us take a look at the good quality results, which are shown as Figure 18 and increasing quality results are shown as Figure 19

Increasing number of hidden layers in both Critic and Generator simultaneously.

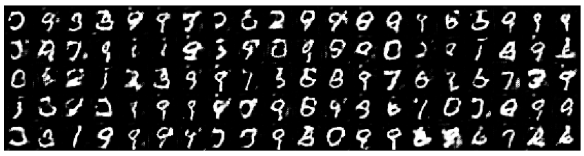


Figure 18: A random sample with final results.

(a) Results on 3 hidden layers

(b) Results on 4 hidden layers

(c) Results on 5 hidden layers

Figure 19: Influence of the number of hidden layers on 3 models, with performance increasing row by row

Intuitively, 5 hidden layers gets the better result than others, which is different to the best number of hidden layers of GAN. So, for each kind of GAN model, there is one matching best number of hidden layers to reach the best performance.

In order to compare the training effort of the WGAN model on different number of hidden layers, a plot of generator loss function value is created and plotted as Figure 20

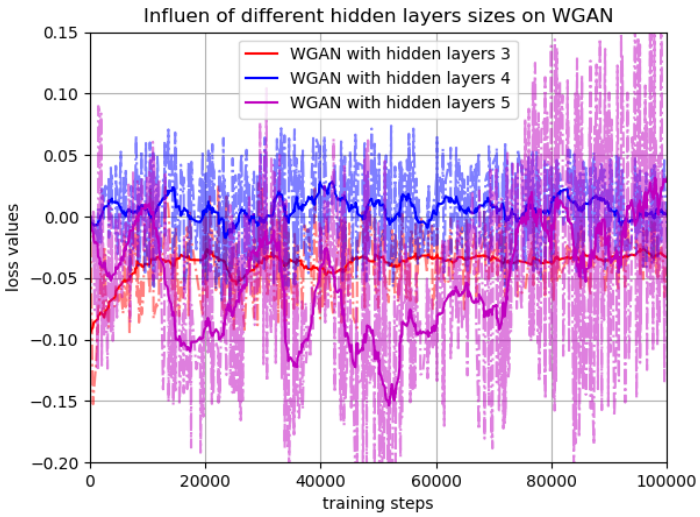


Figure 20: Different number of hidden layers.

A brief conclusion is drawn as follow,

- Still, we can observe that the EMD loss is decreasing through the training process.

- By comparing the final performance of WGAN model on different hidden layers, we can say the more hidden layers the WGAN has, the better performance it will reach.
- It is easy to observe that the model becomes more unstable when we increase the hidden layers. The possible reason for this is that the implementation of WGAN itself is by clipping the weights. Since the more complex the model is, the more weights the model have to clip. As a result, it lead to changing the direction of weights frequently which is the main reason why the loss value fluctuates a lot.

3 CIFAR-10 Dataset

Before dive into the content, we firstly have to apologize for the low quality of the generated images. We have tried several network structures and tuned hyper-parameters multi-times. Also, since the time is limited, we decided to set the training epochs as 1000. The results we are currently presenting in this report is possibly the best we can get.

Back to the report, this section is separated into 3 subsections with each model occupying one subsection.

3.1 VAE

3.1.1 Latent Size

First of all, let us take a look at the good quality results, which are shown as Figure 21

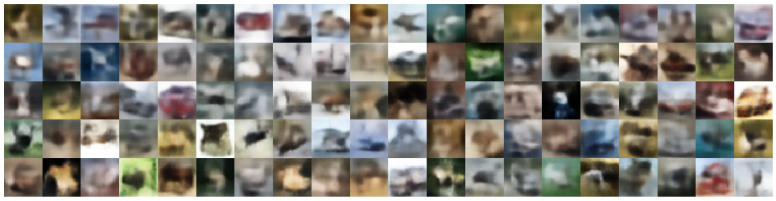


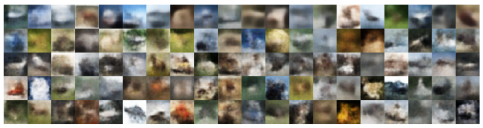
Figure 21: A random sample with final results.

And increasing quality results are shown as Figure 22

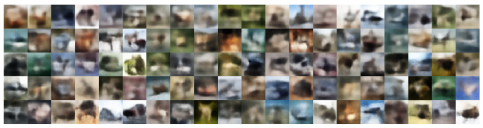
In order to compare the training effort of the VAE model on different latent space sizes, a plot of generator loss function value is created and plotted as Figure 23

- Compared to the VAE model on MNIST dataset, the training process of the VAE on the CIFAR dataset has the similar characters. This is because there exists only one

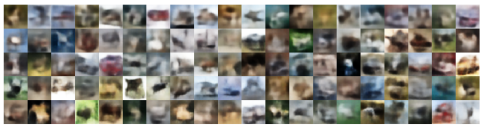
(a) Results on latent size 10.



(b) Results on latent size 20.



(c) Results on latent size 50.



(d) Results on latent size 100.

Figure 22: Influence of the latent size on 4 models, with performance increasing row by row

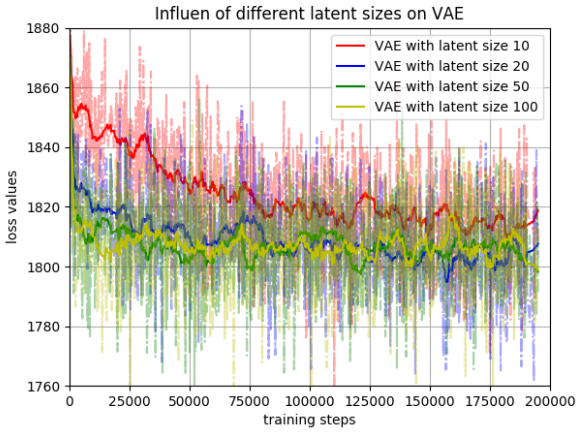


Figure 23: Different latent size.

loss function and both encoder and decoder are trained based on the gradient of that, which means the loss can be general smoother than GAN or WGAN.

- The loss value of VAE with latent size 100 and 50 are the lowest two finally in the plot. VAE needs more information to generate images in CIFAR10 datasets because images in this datasets are complex.
- The loss value of VAE with latent size 10 is the highest from start to end, which means CIFAR10 images is hard to generalize when given small latent size.

3.1.2 Number of Hidden Layers

First of all, let us take a look at the good quality results, which are shown as Figure 24

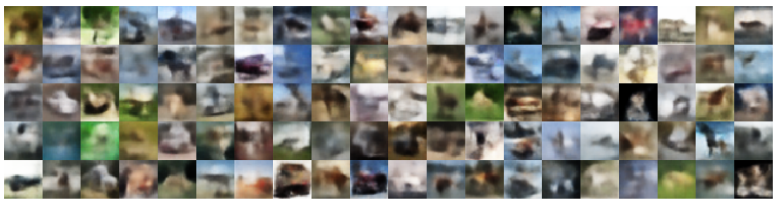
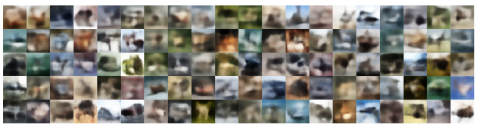
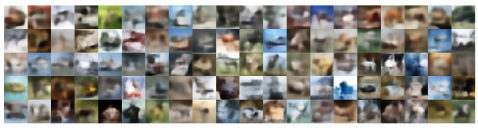


Figure 24: A random sample with final results.

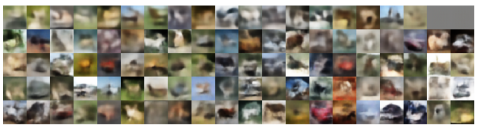
And increasing quality results are shown as Figure 25



(a) Results on 3 hidden layers



(b) Results on 4 hidden layers



(c) Results on 5 hidden layers

Figure 25: Influence of the number of hidden layers on 3 models, with performance increasing row by row

In order to compare the training effort of the VAE model on different number of hidden layers, a plot of generator loss function value is created and plotted as Figure 26

A brief conclusion is drawn as follow,

- The loss value of VAE with 5 hidden layers is the lowest and this model can generate better images with more complex structure. This is possibly because images of CIFAR10 is so complex that VAE model needs more complex structure to learn it.
- Although the performances of the model with different complexities show a slight difference which has been observed and concluded above, One can, in general, say that the influence of the size of hidden layers cannot alter the performance of the model that much.

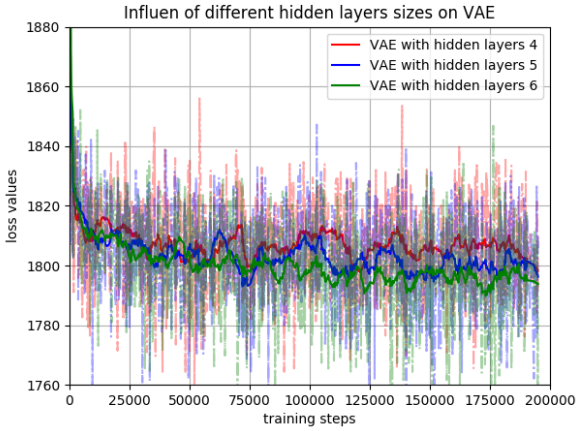


Figure 26: Different number of hidden layers.

3.2 GAN

3.2.1 Latent Size

First of all, let us take a look at the good quality results, which are shown as Figure 27

Figure 27: A random sample with final results.

And increasing quality results are shown as Figure 28

Apparently, the results are way shaper than those of VAE. This means, with certain training skills, GAN could get better prediction results than VAE. That is one possible reason why GAN is so welcomed all over the world.

In order to compare the training effort of the GAN model on different latent space sizes, a plot of generator loss function value is created and plotted as Figure 29

A brief conclusion is drawn as follow,

- Generally speaking, the results follow agree on the knowledge we learnt from the class. The loss value goes up and the convergence speed slows down, which means the model is learning well on the dataset.

(a) Results on latent size 10.

(b) Results on latent size 20.

(c) Results on latent size 50.

(d) Results on latent size 100.

Figure 28: Influence of the latent size on 5 models, with performance increasing row by row

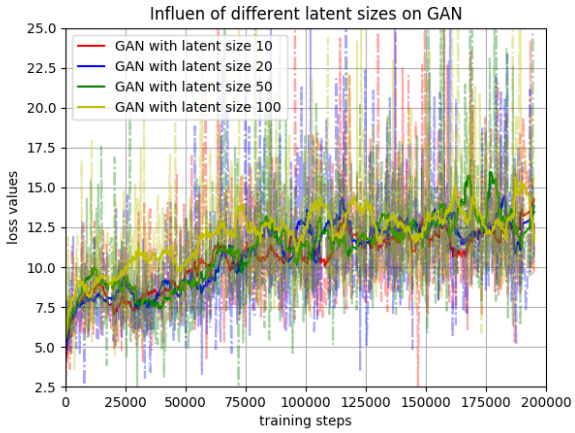


Figure 29: Different latent size.

- One can say that the influence of different latent sizes on GAN can be neglected, since all the performances of these models show a lot similarities.
- In CIFAR10 datasets, we need more hidden layers to train these complex images. The best performances is reached when the number of hidden layers hits 4, although only slight differnece could be captured.
- GAN with latent size 100 can generate well than other latent size models. This is possibly because CIFAR10 so complex that it needs more latent size to learn.

3.2.2 Number of Hidden Layers

First of all, let us take a look at the good quality results, which are shown as Figure 30

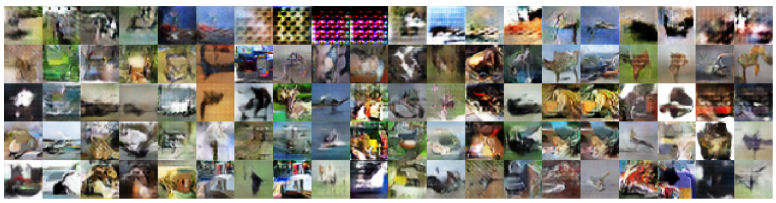


Figure 30: A random sample with final results.

And increasing quality results are shown as Figure 31

(a) Results on 3 hidden layers

(b) Results on 4 hidden layers

(c) Results on 5 hidden layers

Figure 31: Influence of the number of hidden layers on 3 models, with performance increasing row by row

Same as before, the generated images are quite shape. Given more time training on it, we think the result could get better.

In order to compare the training effort of the GAN model on different number of hidden layers, a plot of generator loss function value is created and plotted as Figure 32

A brief conclusion is drawn as follow,

- GAN with 5 hidden layers gets the best result and this means larger and smaller number of hidden layers will reduce the performance.
- The generator loss value is still increasing when we stop iterations.Training GAN in CIFAR10 datasets needs more epochs, this shows that GAN is very difficult to train.
- Another thing we have to mention about the GAN is, the loss value goes extremely wild at the training process goes. This also meets the knowledge we have learnt in

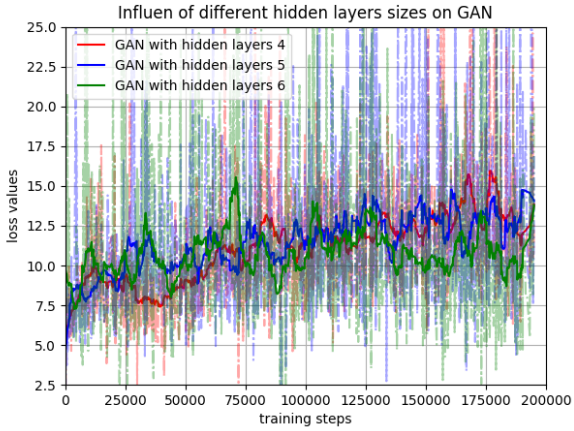


Figure 32: Different number of hidden layers.

the class, which is it begins fluctuating at the beginning, then the magnitude of the oscillation becomes smaller and smaller. And finally, when the loss gets stable, the output results will become significantly promising.

3.3 WGAN

3.3.1 Latent Size

First of all, let us take a look at the good quality results, which are shown as Figure 33

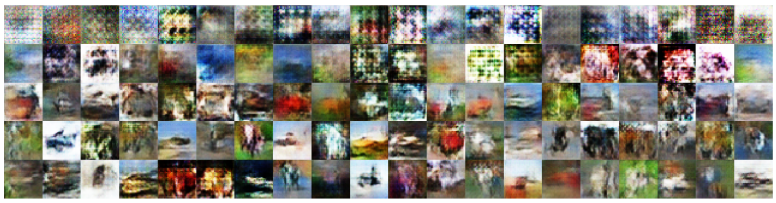


Figure 33: A random sample with final results.

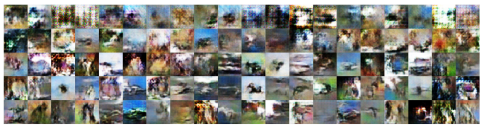
And increasing quality results are shown as Figure 34

Obviously, the quality of generated images of WGAN is worse than that of GAN. This does not necessarily mean that WGAN is no better than GAN. One possible reason is that since the WGAN requires weights clipping, it is naturally harder to train. If given enough time, I honestly think the performance of WGAN would be better than GAN.

(a) Results on latent size 10.

(b) Results on latent size 20.

(c) Results on latent size 50.



(d) Results on latent size 100.

Figure 34: Influence of the latent size on 5 models, with performance increasing row by row

In order to compare the training effort of the WGAN model on different latent space sizes, a plot of generator loss function value is created and plotted as Figure 35

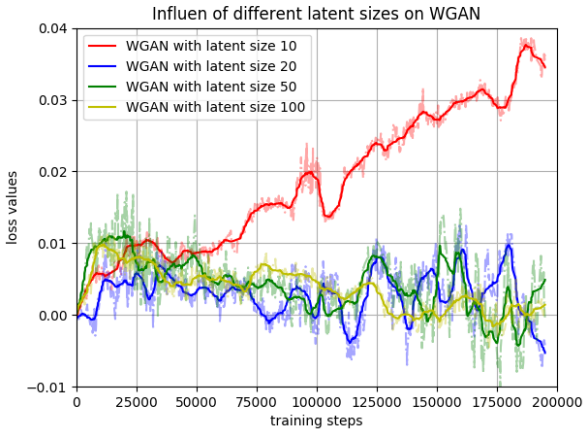


Figure 35: Different latent size.

A brief conclusion is drawn as follow,

- WGAN is typically hard to train. As one can observe, although the training step goes already over 800, the loss still fluctuates a lot.
- WGAN can not generate images well in small latent size. In this plot, the WGAN with latent size 10 learnt extremely bad than other latent sizes. This is possibly because WGAN and other kind of GANs need more information to generate complex images.

- The reason why WGAN with latent size 10 shows un-resonable results is probably that the latent size is too small and it requires WGAN takes much more time to start converge.

3.3.2 Number of Hidden Layers

First of all, let us take a look at the good quality results, which are shown as Figure 36

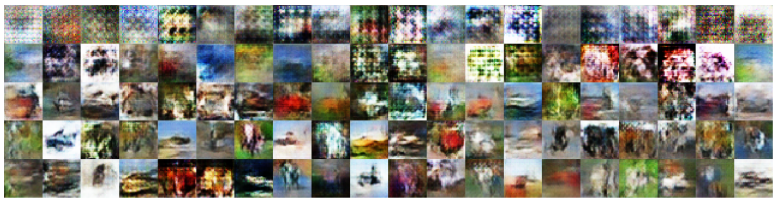
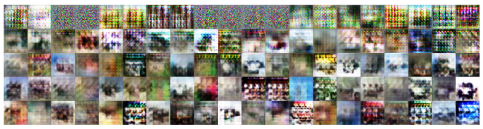


Figure 36: A random sample with final results.

And increasing quality results are shown as Figure 37

(a) Results on 3 hidden layers

(b) Results on 4 hidden layers



(c) Results on 5 hidden layers

Figure 37: Influence of the number of hidden layers on 3 models, with performance increasing row by row

In order to compare the training effort of the WGAN model on different number of hidden layers, a plot of generator loss function value is created and plotted as Figure 38

A brief conclusion is drawn as follow,

- WGAN with more hidden layers first learnt bad and then gradually learnt better and better. Finally, WGAN with 6 hidden layers gets the best results,and this shows that the complex model need to be applied to train complex dataset images.

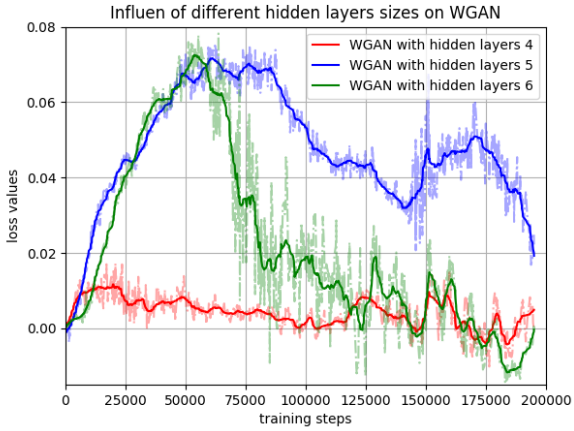


Figure 38: Different number of hidden layers.

- Since the hidden layer size 4 is probably still not deep enough for the model to learn on such dataset, the loss value basically keeps a certain level during the whole training process, though one can still spot it decreases slightly.

4 Other comments about training these models

Some experience of building up hidden layers structure and tuning hyper-parameters:

- Using LeakyRelu as the activation function of each hidden layer can improve the performance better.
- Adding Batch Normalization after each hidden layers rather than dropout, this is because dropout in generator will lose information, which leads to GAN and WGAN harder to train.
- Larger de-convolutional kernel size must be applied in generator when training higher quality images. In this assignment, the image quality of CIFAR10 is higher than MNIST. This is because high quality image features need to be extracted with more information.
- Only add one dropout layer after the flatten layer in discriminator, which can reduce over-fitting.
- Note: without these experience, the models can be trained well in MNIST datasets because the images in MNIST datasets are simple, meaning hands-writing images are learned by models easily. In CIFAR10 datasets, because images are more complex, models can not "tolerate" these problems without fine-tuning.

- Weight clipping range value is a key hyper-parameter to influence the model performance in WGAN, in this assignment, we used 0.01 as the clipping value, which is an experience value online. After we tuning this hyper-parameter, we found this experience value is the best we can find aimed to this assignment.