Assignment-3 Report

COMP4450

Keyang Qian u7261302
Ruofan Yang u7307578
Ke Ran u7261338

28/05/2022

Contents

Question 1

Our research topic is novelty detection using generative adversarial network.

For novelty detection, novelty is situations which can hardly be considered or expected, which could cause unexpected reaction or consequence in application. We aim to detect the novelty, which is precondition for preparing the effective solutions.

And Generative adversarial network would be implemented for novelty detection. GAN is normally used for image processing at first, and could be used for classification. It contains two networks, generative network and discriminative network. From usage of classification aspect, Generator network take in-class image and noise as input, learning the distribution of in-class images, and generate fake image. Discriminator network take the generated fake image or the in-class image as input, and discriminate whether it's fake image or real image. And they will be trained alternatively, compact with each other for lower losses.

Can we use GAN to detect novelty? To figure out, we let generator learn the distribution of in-class images, and mark all the out-class images that could hardly expected as novelty. Then outliers that insufficiently or never observed for generator during training will have strange reconstructed output, which could easily be distinguished by the trained discriminator. That would be the basic idea, but in practical, things are far more complicated, waiting for us to do research on.

In our research, we would attempt to build GAN network with some improvements as a one-class classifier to detect the novelty.

Question 2

The reason for us choosing this research topic can divide into two parts. First, why do we want to do novelty detection. Traditionally, agents are trained within a closed space and are tested using all known datasets. However, as more and more AI agents are used in the real world and some agents are required when modelling some situations that will cause difficult unforeseen challenges, the relationship between agents and humans is getting closer. As the real world is unlike the training environments, it is full of unknowns and accidence, and the traditional agents cannot deal with such situations and may cause the failure of task or even financial loss and personal injury.

Overall, the agents need the ability to detect novelties and adapt to them to successfully do tasks in the real world. The first ability is the topic we focus on. For example, a semantic segmentation network cannot model strollers on the upper row or street market on the lower row, because they are not in the K closed-set classes of the training set. This kind of misclassification can be a disaster when fed into the auto vehicle. Moreover, if we can perfect implement the novelty detection function on an agent. Some agents around us might contain the networks designed by us. Also, we achieve the first step that agents can learn from the environments by themselves. And then we are getting closer to the dream that we can not only use the agents to adventure by simulating some situations and environments that are hard or even impossible to reach but also can help us overcome more difficult and elusive games or knowledge.

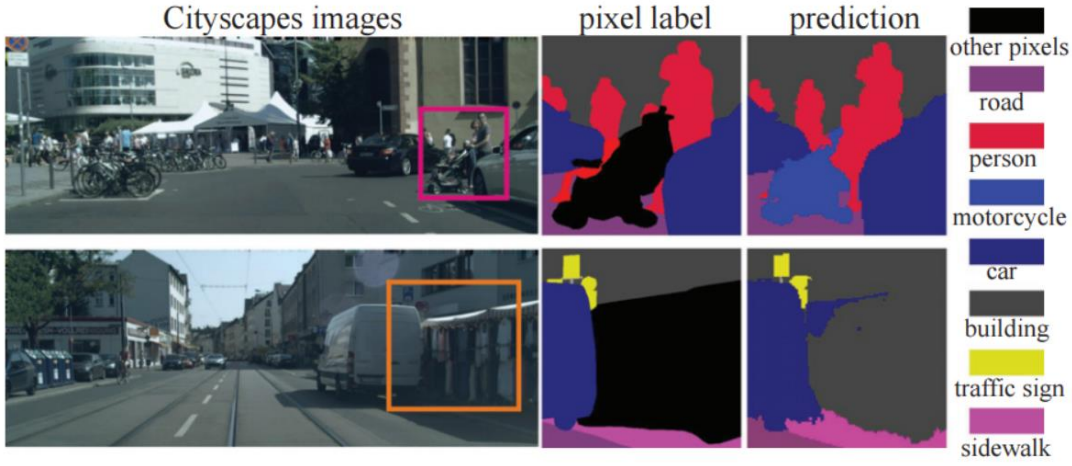


Figure 2.1: Some semantic segmentation network results. Image retrieved from *OpenGAN: Open-Set Recognition via Open Data Generation*[4]

Then the reason for us to use GAN to detect novelties is sure because GAN network has many advantages over others. But it is also because this kind of adversarial learning method is very interesting, the generator and discriminator compare against each other and will improve if one of them improved, like in the real world, people can always make larger progress with the pressure from opponents. Also, the same as in real life, people will not attempt to make progress if there is no comparison object or if the difference is huge. The components will also stop training if one of the networks is too strong. These all make the GAN network easy to understand and funny. Back to the performance, the component of GAN, the discriminator is a born network to detect the novelties. It can be simply used to classify the novelties and known data. And the ability can be easily improved by training the generator's ability to generate novelties that look more similar to the known data. Moreover, as GAN is good at unsupervised learning which in the real world, most data do not have labels. At last, the experiment result shown in the paper *Novelty detection with GAN*[3], points out that the performance of GAN is comparable to several popular novelty detection methods, and sometimes outperforms them.

Question 3

Current state-of-the-art summary:

In the area of novelty detection using GAN, almost all researchers take focuses and experiments to improve the performance of the GAN. And they mainly proceed with the following respects: change the loss functions and add a penalty to enforce the assumptions, avoid mode collapse, optimize the model, and change the supervised way.

With the respect to mode collapse, the reason it will happen is that once the generator can cheat on the discriminator, the generator will only produce the least real data. And it will happen on the discriminator loss, the cross-entropy. As an improvement on this, the paper *TransductGAN: a Transductive Adversarial Model for Novelty Detection*[12] *published* in 2022 with the TransductGAN model has the flexibility of using any types of discriminator loss mentioned in[15][19], to avoid using the cross-entropy can avoid the problem of mode collapse.

Then optimizing the model is the most diverse and important way to improve the GAN model. And many researchers have made much effort on this.

One way is to introduce the latent representations and separate the in-class examples with the novelties by different bounded spaces. Researchers uses a denoising auto-encoder network to learn the latent representation of inliers and reconstruct the denoised version of the image. The denoising auto-encoder network helps reduce overfitting and improve the generalizability. Also, by introducing the tanh activation, it can bound the latent space in the encoder's output layer. At last, we need to use a latent discriminator to ensure latent representations of inliers are distributed uniformly across the latent space and a classifier to compare how well a given image resembles with given class's content. This kind of method of improving GAN is raised in the paper *OCGAN: One-class Novelty Detection Using GANs with Constrained Latent Representations*[6], the paper constructs the GAN with these components and have had test which has achieved a result that about 98 percent of outliers can be detected in the MNIST dataset. But this kind of one-class classifier works worse if the more concepts are presented in the image, so future works are needed when testing on more complex dataset.

Some researchers even change the deep learning related method, and change it to the transductive method. As Inductive learning cares about the error rate for the whole examples, while transductive learning, only cares about the decision tree of the testing case and unlabelled test set will only be incorporated at training time, so agents can take more focus on the testing cases. And this learning method changes the novel detection to semi-supervised.

We can also add an adversarial auto-encoder to map examples to the latent space based on the novel transductive novelty detection method. It can separate the inliers from novelties and learn to generate images from both inliers and novelties with the help of using two Gaussians in latent space. It solves the problem that how to tune the model hyper-parameters at the decision rule level. It also uses the standard binary classifier to do the novelty detection using the generated example of both the outliers and inliers which helps improve the ability. These changes are introduced in *TransductGAN: a Transductive Adversarial Model for Novelty Detection*[12]. As a result of this model, it can detect an approach to 100 percent of the novelties in the MNIST dataset, and about 90 percent in the more complex dataset CIFAR10. But it will perform worse if the proportion of novelties is large and has a lack that it needs to know the contamination rate.

The origin function used in many current GAN will cause the GAN to be trained unbalanced and makes it hard to train if the discriminator is trained too well, then it cannot do gradient descent and the generate loss will be close to constant.

In paper *Wasserstein generative adversarial networks*[15], it changes the loss function and uses the Wasserstein distance, to replace the origin loss function in basic GAN. It can solve the unbalance problem by replacing the JS divergence and just needs to change few things in the model. Furthermore, in *Improved training of wasserstein gans*[18], it uses the gradient penalty to replace the weight clipping and adds a penalty term based on the WGAN loss function. It solves the problem that the parameter is binarized and the training is hard to adjust. But rare GAN models that used to detect novelties use the WGAN loss function, although they suffer the problems that training is easy to be unbalanced and fail as mentioned above.

Paper summary 1: *OCGAN: One-class Novelty Detection Using GANs with Constrained Latent Representations*[6] (Summary author: Ruofan Yang)

Full reference: Perera, P., Nallapati, R., & Xiang, B. (2019). Ocgan: One-class novelty detection using gans with constrained latent representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2898-2906).

The paper presents a GAN model to do one-class novelty detection called OCGAN by learning the latent representations of inliers. The paper, first points out the lack of one-class novelty that the formulation assumes any negative training data is unavailable. But with the help of learning the representative latent space, the paper can do novelty detection using the strategy that detecting the difference between the real images and reconstructions. The paper wants to ensure latent space only learns one given class example and all generated examples are from known class. And it explains the problem that although many models can reconstruct the inliers well, outliers will also reconstruct well in latent space which is bad, and it wants to force the latent space to represent only the given class so that all examples will reconstruct similar to the given class.

To achieve this, the paper proposes an OCGAN that contains four components. Denoising auto-encoder is used to learn the representation for a given concept and generate images from latent space. It can help avoid over-fitting, improve the generalizability and get a larger latent dimension. It also contains a tanh activation to bounded the latent space. Latent discriminator is trained and is to distinguish latent representation of inliers and random samples, to ensure latent representations of inliers are distributed uniformly across the latent space. Visual discriminator is to distinguish images of given class and generated random sample and is trained based on these too, it is to ensure all samples are from one class. Then it introduces an informative-negative sample which means the poor quality samples generated. It is to train the generator to produce better images. At last, it has a classifier that is trained using reconstructive inliers and random images to classify how similar a given image is compared to the given class images.

The whole GAN first trains the classifier and then trains two discriminators adversarially, then the agent looks for informative-negative sample in latent space and does the gradient descent, then trains the auto-encoder adversarially with discriminators. To choose the hyper-parameters, researchers will select the model which produces the least MSE.

The paper used COIL100, fMNIST, MNIST, and CIFAR10 datasets to test the behavior which set one class as the known class, and others as unknown. All results show that OCGAN can detect more outliers compared to other methods, and as more components are added to the GAN model, the result is better. Also for dataset like CIFAR10 which has many concepts, the result can achieve above 60 percent, and for others that have only one concept, the results are all above 90 percent.

The results show that OCGAN does perform better than other methods and improves the ability to detect one-class novelties. And each component of it is useful in

improving the performance. It also shows the problem that it will perform worse in datasets with complex structures and should keep working on it.

Paper summary 2: *Generative adversarial network based novelty detection using minimized reconstruction error*[20] (Summary author: Ke Ran)

Full reference: Wang, H. G., Li, X., & Zhang, T. (2018). Generative adversarial network based novelty detection using minimized reconstruction error. *Frontiers of Information Technology & Electronic Engineering, 19*(1), 116-125.

The paper proposes OpenGAN combining with several technical insights for open-set recognition. First, the paper introduces the basic background of novelty detection using k-way classifier with the problems existing for that. On the one hand, discriminatively learning an open-vs-closed binary discriminator by exploiting some outlier data as the open-set generalizes poorly to diverse open test data due to over-fitting to the training outliers, which unlikely exhaustively span the open world. One the other hand, unsupervised learning the closed-set data distribution with a GAN and using its discriminator as the open-set likelihood function does not work well, presumably due to the unstable training of GANs.

To solve this, the paper proposes an OpenGAN by incorporating two technical insights. 1) training an open-vs-closed classifier on OTS features rather than pixels, the paper build the discriminator over the features computed by the closed-world K-way networks. 2) adversarialy synthesizing fake open data to augment the set of open-training data. The paper augments the available set of real open training examples with adversarially synthesized “fake” data. The improvement of methodology of OpenGAN is mainly on that OpenGAN trains with both the real open and closed-set data and the fake open-data into a single (GAN-like) mini-max optimization over D and G. In this case, the paper does not learn a separate open-set model but directly use the already-trained discriminator as the open-set likelihood function. The paper build GANs over OTS feature embeddings learned by the closed-world K-way classification Networks since generating high-dimensional realistic images is challenging and may not be necessary to open-set recognition. Instead of relying on the reconstruction error for open-set discrimination, the paper directly use the discriminator as the open-set likelihood function.

The paper used MNIST/CIFAR/SVHN/TinyImageNet for Open-set discrimination Test. OpenGAN performs the best competing with prior methods such as GDM and GMM, suggesting that the GAN-discriminator is a powerful open likelihood function. The paper used TinyImageNet for Cross-Dataset Open-Set Recognition, this result further shows the merit of generating fake open examples to augment heavily-biased open-set training data and the technique insights. But the results also imply a failure mode of OpenGAN because the open-set data used in training could be quite different from those in testing, potentially leading to an OpenGAN that perform poorly in the real open world. Finally, the paper using Cityscapes for Open-Set Semantic Segmentation, with results also better than previous methods.

With OpenGAN, this paper shows using GAN-discriminator does achieve the state-of-the-art on open-set discrimination, once being selected using a val-set of real

outlier examples. This is effective even when the outlier validation examples are sparsely sampled or strongly biased. OpenGAN significantly outperforms prior art on both open-set image recognition and semantic segmentation.

Paper summary 3: *P-KDGAN: Progressive Knowledge Distillation with GANs for One-class Novelty Detection*[7] (Summary author: Keyang Qian)

Full reference: Zhang, Z., Chen, S., & Sun, L. (2020). P-kdgan: Progressive knowledge distillation with gans for one-class novelty detection. *arXiv preprint arXiv:2007.06963*.

The paper proposes P-KDGAN, an encoder-decoder-encoder pipeline based GAN model to do one-class novelty detection and achieve state-of-the-art performance. The paper also proposed the progressive knowledge distillation with GANs, which is a novel exploration that applies the knowledge distillation on two standard GANs. The two-step progressive learning can continuously improve the performance and reduce shock of the student network, in which the designed distillation loss plays an important role.

First, the paper introduces the basic background of Knowledge Distillation (KD). And the Ganomaly structure, namely the encoder-decoder-encoder structure which is more suitable and widely used for one-class novelty detection. Then, they combine the KD with GAN loss, designed four distillation structures for training and their corresponding loss function. They will be applied in different training step. E.g., loss 1) KDGAN1 is the training of student network, which only depends on the distillation loss *Kl*, its training speed is the fastest but it results in poor detection performance. After review the previous work on one-class novelty detection, Ganomaly structure and method of Knowledge Distillation, they introduced the P-KDGAN methodology.

The P-KDGAN method is composed of three modules. 1) Knowledge distillation with GANs (KDGAN), in which the distillation losses based on Ganomaly framework are proposed for transferring knowledge from the teacher GAN to the student GAN, which is the basic training structure. 2) Four distillation structures are designed for KDGAN, they would be implemented for different training step with corresponding loss combination. 3) The two-step progressive learning of KDGAN can continuously improve the performance of the student GAN. And the progressive learning of KDGAN, is a two-step approach that continuously improves the performance of the student GAN and achieves better performance than the single step methods.

The paper used CIFAR-10, MNIST and FMNIST datasets to quantify the performance results using AUC of ROC. The model achieves the best performance on three datasets, which proves the effectiveness of the progressive learning of KDGAN. And the discrepancy in the performance of variant KDGANs consist of different distillation structures concludes that student networks with random initialization can only learn the basic knowledge of the teacher networks, and the fine-training in the second step of P-KDGAN can further improve performance. With P-KDGAN, this paper does achieve the state-of-the-art on one-class classifier novelty detection with a compressed structure using KD.

However, the paper only proposed a theoretical method that can be used to compress other GANs-based applications, maybe a further systematic methodology is necessary

on this topic for general implementation that could greatly promote the progress of GANs-based application.

Question 4

Before and during the first week, we first learned to write GAN using pytorch. We're all newbies in the field of both reinforcement learning and even pytorch, which makes the project a great challenge to us.

We did literature review in this field meanwhile. Since novelty detection using GANs is not a very big field, we could have a look at a large proportion of recent papers in the field. Then we discussed about 13 relevant papers together in total and shared interesting points in these papers and how these could be used and improved in our research when we gradually settled our ideas and works to be done.

In the second week, we initialize our model by implementing a normal GAN model that is used in novelty detection. We decided to implement a shallow network imitating a recent paper[1] in 2018 as our initial model, then tried to found issues and applied our ideas instead of training a complex big model for higher scores. The structure of the model is shown as **Figure 4.1** followed. The structure of the network is shown in **Figure 4.2**.

As previous researches[2] investigated, it's useful and widely-used to train an auto-encoder on samples from target class (inliers) and use it for novelty detection. The auto-encoder would learn features from the inlier training samples, thus for outlier samples (novelty) the reconstruction error would be high for us to distinguish them and reconstruction result would be bad. To either train the auto-encoder or to train the final discriminator for novelty detection, we use a D network and adversarially train these together as a GAN network.

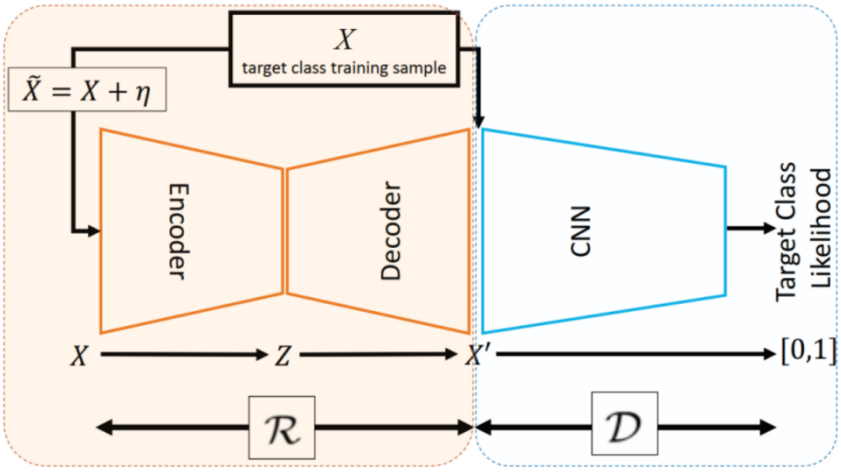
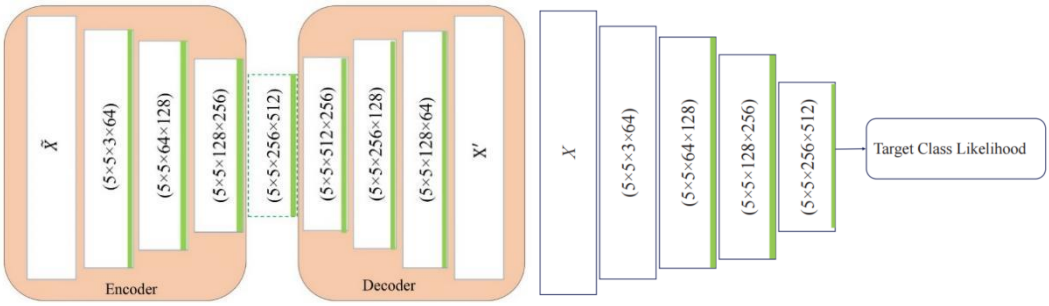


Figure 4.1: Our GAN model. Networks are implemented by CNN networks. Use the target class samples with Gaussian noises as network input to strengthen robustness of R network and add difficulty to the initial D network. Reference souce: [1]



(1) (2)

Figure 4.2: Architecture of R network (1) composed of Encoder and Decoder and D network (2). The parameters for each convolution layer are shown in the form of (first dimension of kernel × second dimension of kernel × number of input channels × number of output channels). Reference source: [1]

We used the classical GAN target loss function as our initial loss function to train the model, as shown in **Figure 4.3**, except that we added some Gaussian noises in the R network input.

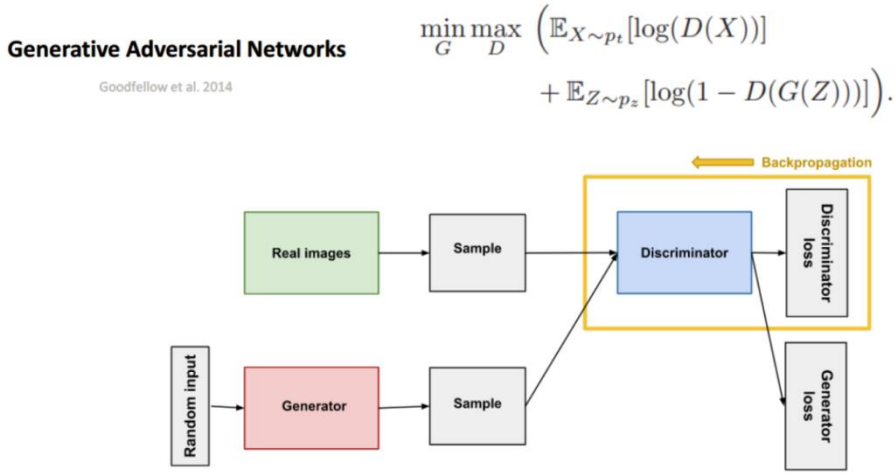


Figure 4.3: Classic GAN loss function and the loss structure. Reference source:[5].

As for reconstruction loss, we use the Mean Square Error (MSE) between the real image and the reconstructed image, as **(1)** has shown.

$$Loss_R = ||X - X'||^2 \quad \textbf{(1)}$$

The initial loss functions, including reconstruction loss, G loss and D loss, are shown in the **Code Block 1** below.

```
def R_Loss(d_net: torch.nn.Module, x_real: torch.Tensor, x_fake: torch.Tensor, lambd: float) -> dict:

    pred = d_net(x_fake)
    y = torch.ones_like(pred)

    rec_loss = F.mse_loss(x_fake, x_real)
    gen_loss = F.binary_cross_entropy_with_logits(pred, y)

    L_r = gen_loss + lambd * rec_loss

```

```
    return {'rec_loss' : rec_loss, 'gen_loss' : gen_loss, 'L_r' : L_r
}

def D_Loss(d_net: torch.nn.Module, x_real: torch.Tensor, x_fake: torc
h.Tensor) -> torch.Tensor:

    pred_real = d_net(x_real)
    pred_fake = d_net(x_fake.detach())

    y_real = torch.ones_like(pred_real)
    y_fake = torch.zeros_like(pred_fake)

    real_loss = F.binary_cross_entropy_with_logits(pred_real, y_real)

    fake_loss = F.binary_cross_entropy_with_logits(pred_fake, y_fake)


    return real_loss + fake_loss
```

Code Block 1

And we used MNIST dataset as our initial testbed, number '1' as our target number. Samples in label '1' are our inlier training samples, because MNIST is widely used in the field of novelty detection[5][6][7][8], reliable, easy for implementation and result comparison. We did ToTensor() and normalization to the training, validation and testing set.

We extract some sample images from the test dataset to visualize and measure our model by comparing the reconstruction loss and discriminator output (D(X)) of inliers and outliers. Some results for our initial model after doing 80 epochs are shown in **Figure 4.4**.

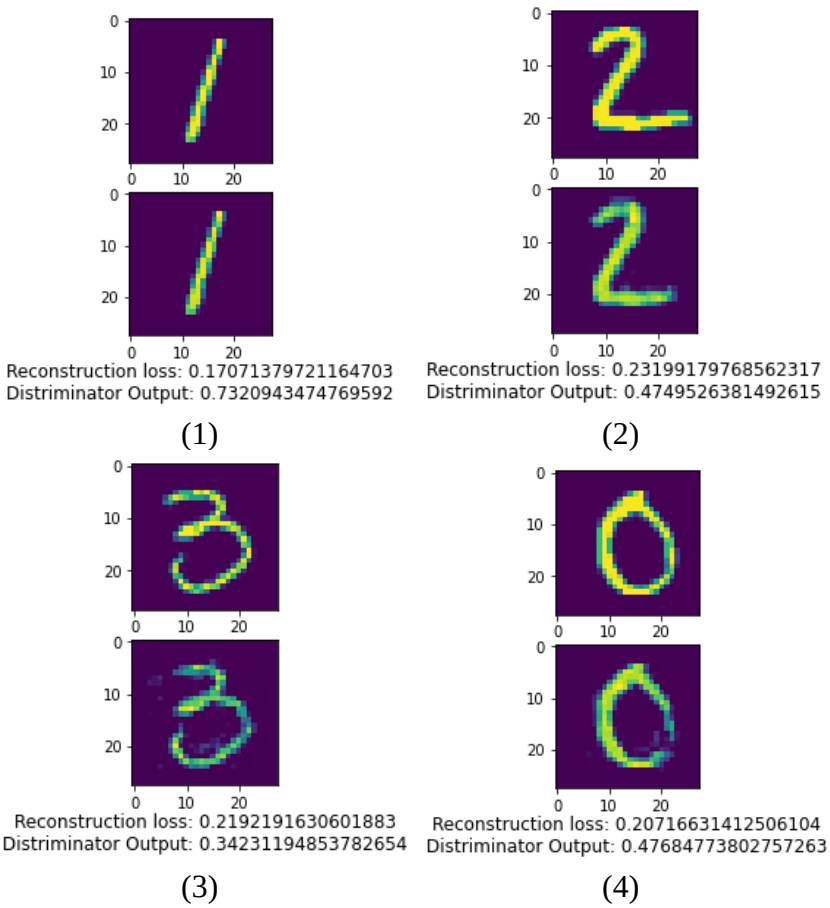
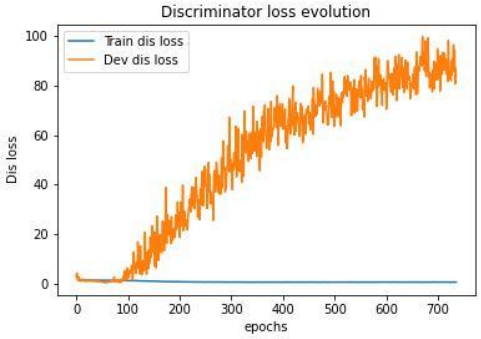


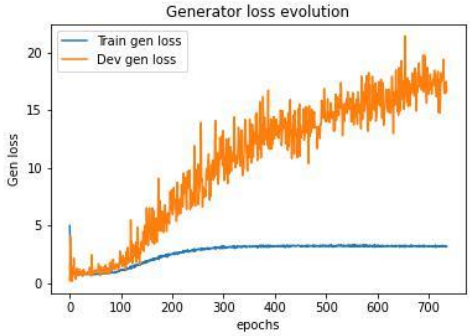
Figure 4.4: The real (above) and fake (below) images, reconstruction losses and

discriminator outputs. Target class is 1 (inlier). We can distinguish outliers either by reconstruction loss or by discriminator output or combine them together.

The results are good for a model that has only be trained for 80 epochs. But we found our model suffer the issue that after the first 60 epochs, the more we trained, the bigger loss comes to D&G networks. And the reconstruction loss does not improve anymore after first 60 epochs as well. The network learns irrelevant feature and lose its novelty detection ability during training. These are shown in **Figure 4.5** and **Figure 4.6**. We believed that the paper model[1] we tried to imitate originally have this issue.
We would try to dug into this issue in week 3 and week 4.



(1)



(2)

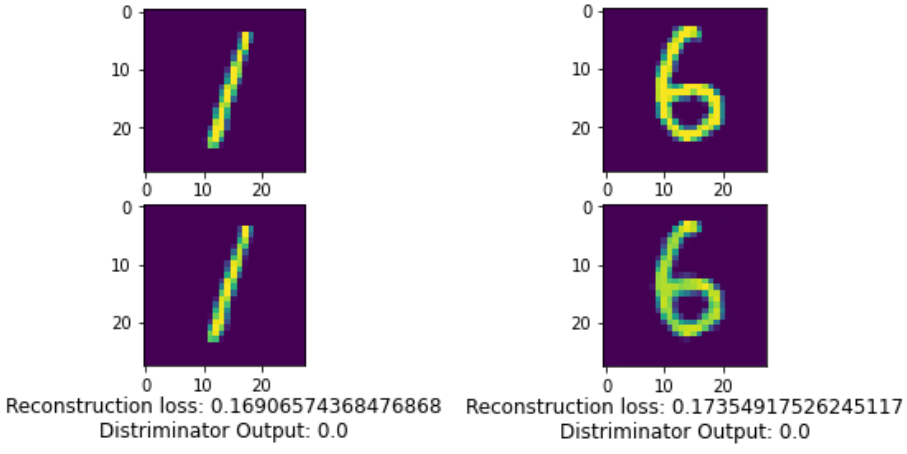
Epoch 81:
TRAIN METRICS: {'rec_loss': 0.16574676975373592, 'gen_loss': 0.8162824053044349, 'dis_loss': 1.363162185275141}
VALID METRICS: {'rec_loss': 0.1664239286851253, 'gen_loss': 1.0094688919672357, 'dis_loss': 1.2585190020993944}

Epoch 353:
TRAIN METRICS: {'rec_loss': 0.16419693667610688, 'gen_loss': 3.1423961964179767, 'dis_loss': 0.6217490405295165}
VALID METRICS: {'rec_loss': 0.16698248102801486, 'gen_loss': 23.262723705947135, 'dis_loss': 81.49931855726872}

Epoch 602:
TRAIN METRICS: {'rec_loss': 0.16398145776939335, 'gen_loss': 3.283472541530703, 'dis_loss': 0.5979073433096633}
VALID METRICS: {'rec_loss': 0.16807397514712968, 'gen_loss': 30.07347879955947, 'dis_loss': 131.61035242290748}

(3)

Figure 4.5: D (1) & G (2) loss vs epochs and some console outputs of D & G loss vs epochs (3).



(1) (2)

Figure 4.6: Novelty detection results after 800 epochs. Novelty detection results for our GAN model. The real (above graph of each subfigure) and fake (below

graph of each subfigure) images, reconstruction losses and discriminator outputs. Target class is 1 (inlier).

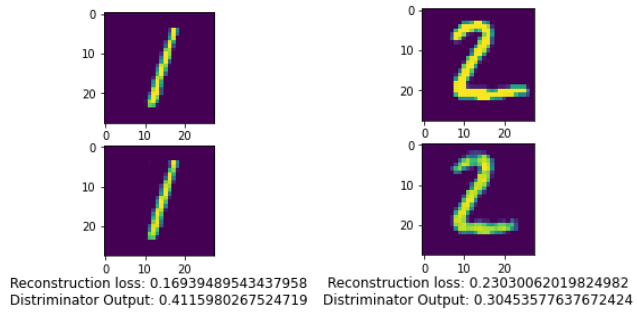
In week three, we began the real 'novel' parts of our research that focused on applying our ideas to develop our method.

Normally people would use either the auto-encoder or the D network for novelty detection and discard the other after training process. But since previous researches[3][4] has suggested that the reconstruction score (G score) and the D score are complementary in different scenarios that D score performs better in high-dimensional datasets while G score is more ideal on normal samples close to zero of low-dimensional datasets, we wanted to combine both of the networks together for our novelty detection task.

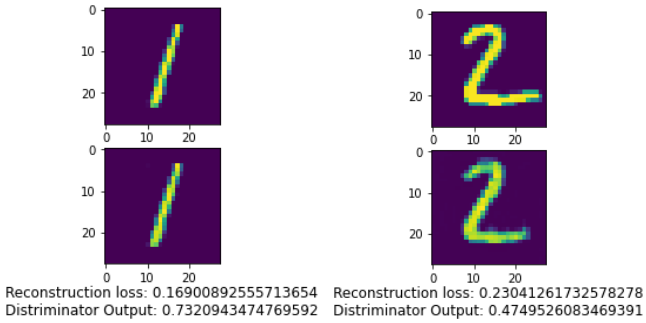
We first tried to use D(R(X)) to combine the two networks together for novelty detection. The intention is to first use the R network to widen the difference between inliers and outliers because **Figure 4.4** has shown that the reconstruction losses are larger for outliers, then use the D network to finally detect the novelty.

But experiment results in **Figure 4.7** show that the differences between inliers and outliers are narrowed after using D(R(X)). This is probably because the performance of R network's reconstruction is not good enough so that D(R(X)) for inliers are reduced too much and it's more difficult to find a suitable threshold comparing to D(X) results.

Another possible explanation is that though the differences are narrowed, they're actually more distinguishable for optimal threshold, which would be tested in week 4.



(1)



(2)

Figure 4.7: Comparisons of the discriminator output results between when discriminator output = D(R(X)) (1) and when discriminator output = D(X) (2).

As for the issue we found in week 2, after referencing papers[10][15] and discussions, we assumed the reason why our previous week model showed unexpected bad

performance in **Figure 4.5** and **Figure 4.6** after 80 epochs is that our network has vanishing gradient problem that the sigmoid function of D network greatly diminished the gradient.

When the reconstructions change rapidly and produce large absolute error of the D network, the sigmoid function narrow the error and thus the gradient was diminished during training. As the diminished gradient accumulated after the network has achieved good performance in the first epochs, the network then become unbalanced and the adversarial training process could not promote both networks together, then losses of both networks increase when being trained more epochs.

We first tried some normal ways to solve vanishing gradient problems. We implemented routine drop-outs and batchnorms with little effect. Then we tried to learn from other networks (DenseNet, ResNet, etc.) and applied residual connections and concatenation on the R network. We believe that this helps to pass features and alleviate vanishing gradients. The architecture is shown in **Figure 4.8**.

The result did be improved slightly. However, applying both methods strongly enhanced G network, so it became more powerful than D network that D loss is at least twice G loss during training when they share the same training rate during the whole training process.

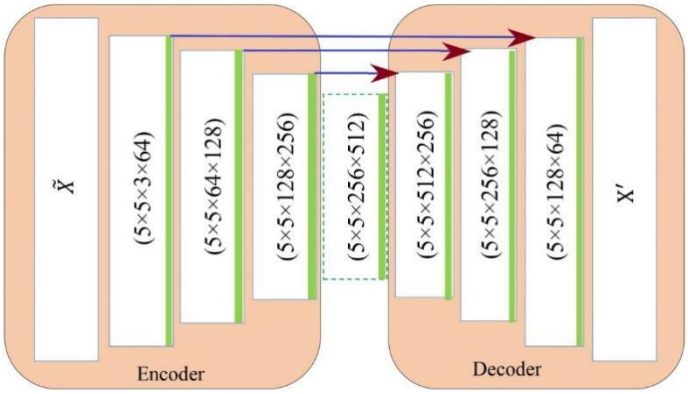


Figure 4.8: Architecture of improved R network. Arrows means concatenation or residual connections (depends on the hyper-parameter we choose). In the case of concatenation, number of input channels in each decoder layer doubled.

Then we want to use different loss functions other than traditional GAN loss, and remove the sigmoid function of D network that leads to vanishing gradients. We found that many state-of-the-art papers in the field did not train many epochs or use OCGAN[6][7][10], but it is suggested recently that Wasserstein GAN solved the problem at bottom while OCGAN not[15]. We didn't find paper that use Wasserstein GAN loss in novelty detection. Therefore, we wanted to do this.

We implement Wasserstein GAN loss[15], RMSProp optimizer and take away sigmoid function from D network to primarily apply a Wasserstein GAN. **Code Block 2** shows the now loss code.

```
def R_WLoss(d_net: torch.nn.Module, x_real: torch.Tensor, x_fake: torch
    .Tensor, lambd: float) -> dict:

    pred = torch.sigmoid(d_net(x_fake))

    rec_loss = F.mse_loss(x_fake, x_real)
```

```
    gen_loss = -torch.mean(pred)
    L_r = gen_loss + lambd * rec_loss

    return {'rec_loss' : rec_loss, 'gen_loss' : gen_loss, 'L_r' : L_r}


def D_WLoss(d_net: torch.nn.Module, x_real: torch.Tensor, x_fake: tor
ch.Tensor) -> torch.Tensor:

    pred_real = torch.sigmoid(d_net(x_real))
    pred_fake = torch.sigmoid(d_net(x_fake.detach()))
    dis_loss = -torch.mean(pred_real) + torch.mean(pred_fake)

    return dis_loss
```

Code Block 2

Figure 4.9 show that D&G loss did not go up and reconstruction loss was still decreasing after 300 epochs.

```
Epoch 326:
TRAIN METRICS: {'rec_loss': 0.1450514882511217, 'gen_loss': -1.006229605458321, 'dis_loss': 0.0}
VALID METRICS: {'rec_loss': 0.14598996334664097, 'gen_loss': -1.014977973568282, 'dis_loss': -4.705353455396476e-08}
TIME: 23.77 s
Epoch 327:
TRAIN METRICS: {'rec_loss': 0.14498245026512904, 'gen_loss': -1.006229605458321, 'dis_loss': 0.0}
VALID METRICS: {'rec_loss': 0.14609423397921256, 'gen_loss': -1.014977973568282, 'dis_loss': -3.360966753854626e-08}
TIME: 23.61 s
Epoch 328:
TRAIN METRICS: {'rec_loss': 0.14489825759928435, 'gen_loss': -1.006229605458321, 'dis_loss': 0.0}
VALID METRICS: {'rec_loss': 0.1460507968448857, 'gen_loss': -1.014977973568282, 'dis_loss': -5.377546806167401e-08}
TIME: 23.68 s
Epoch 329:
TRAIN METRICS: {'rec_loss': 0.14484602193456134, 'gen_loss': -1.006229605458321, 'dis_loss': 0.0}
VALID METRICS: {'rec_loss': 0.14606327275347605, 'gen_loss': -1.014977973568282, 'dis_loss': -2.6887734030837006e-08}
TIME: 23.71 s
Epoch 330:
TRAIN METRICS: {'rec_loss': 0.14482574327096004, 'gen_loss': -1.006229605458321, 'dis_loss': 0.0}
VALID METRICS: {'rec_loss': 0.14579631788615088, 'gen_loss': -1.014977973568282, 'dis_loss': -4.705353455396476e-08}
TIME: 23.61 s
Saving model on epoch 330
Epoch 331:
TRAIN METRICS: {'rec_loss': 0.1448207188699338, 'gen_loss': -1.006229605458321, 'dis_loss': 0.0}
VALID METRICS: {'rec_loss': 0.14592117107912309, 'gen_loss': -1.0149778660173459, 'dis_loss': -8.738513560022026e-08}
TIME: 23.63 s
```

Figure 4.9: Some console outputs of D & G loss vs epochs.

But **Figure 4.10** shows that the discriminator outputs for all inputs were 1 for our initial Wasserstein GAN model, which proves that the network is not 'trained' and out model can't be used.

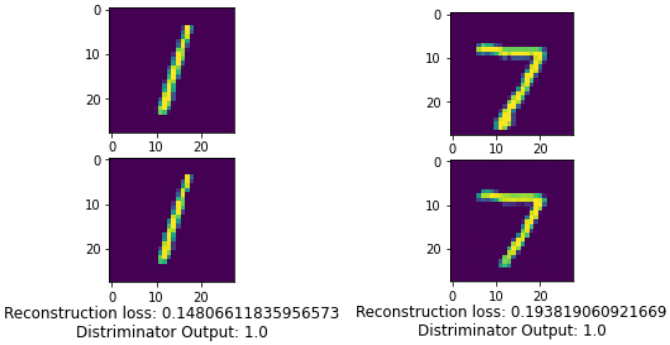
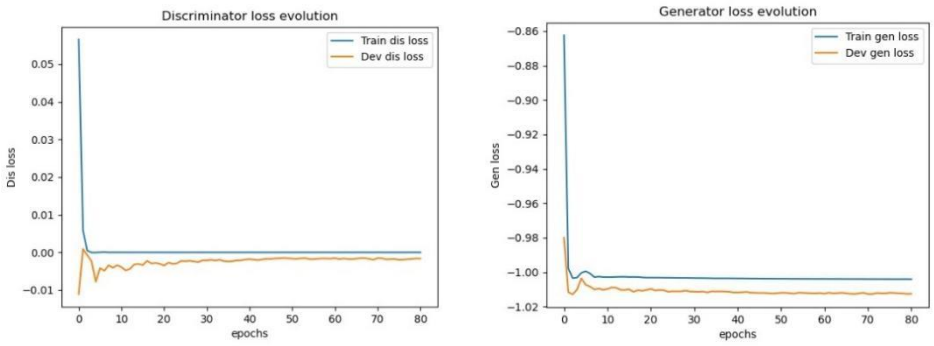


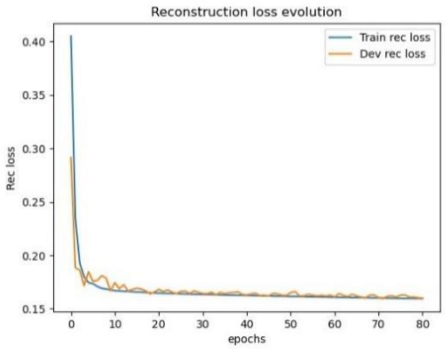
Figure 4.10: Novelty detection results for our GAN model. The real (above graph of each subfigure) and fake (below graph of each subfigure) images, reconstruction losses and discriminator outputs. Target class is 1 (inlier).

This maybe because the network was greatly unbalanced that the D loss in the training set would come to 0 after first a few epochs, and the G loss (with an absolute

value of 1) could not be improved. Only the reconstruction loss went down then, but make no sense. This is shown in **Figure 4.11**.



(1) (2)



(3)

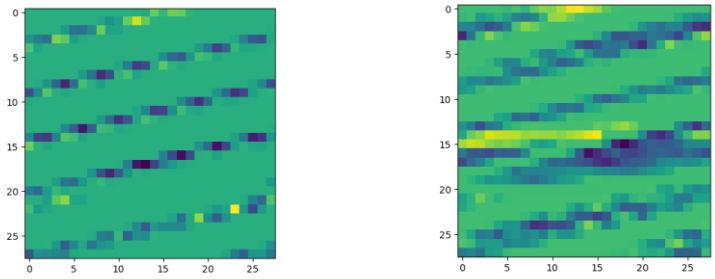
Figure 4.11: D (1) & G (2) network loss vs epochs and reconstruction loss vs epochs (3)

We tried many approaches in week 3 to solve the imbalance:

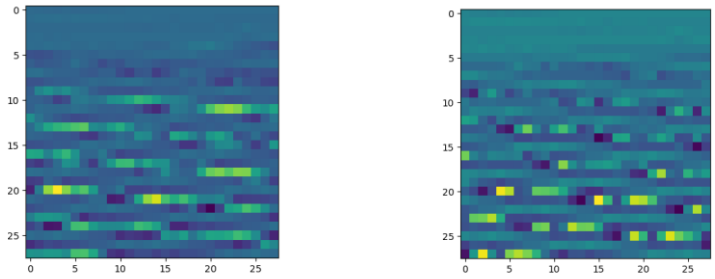
1. Let D optimizer only step when D_loss > 0.4.
2. For each iteration train the G network many times then train D network once.
3. Provide different learning rate for G and D. Use smaller learning rate for D.

But these didn't solve that problem in week 3 because there's some bug in the G network training process.

We also tried to print out some intermediate layer features to help us understand the network, they're printed in **Figure 4.12**. But they didn't seem to make sense for us.



(1) (2)



(3) (4)

Figure 4.12: Intermediate layer features of our model.

In week 4 which is the final week, we first solved some bugs in our R network training process. Then after we adding absolute number clip of D network parameters for each update as required for Wasserstein GAN[15] and doing hyper-parameter adjusting, the imbalance of R network and D network was eventually solved.

Then we applied early-stop to shorten training time and prevent over-fitting by stopping training after getting to local minimum in validation set. After that, we turned to testing stage then because of time limitation.

We first tested our model on MNIST dataset using D(X) as discriminator and the best threshold in validation set and compared our result with other methods.

We trained a simple linear model to combine reconstruction loss with D(X) together. We found optimal linear combination and threshold in the validation set and test it in the testing set. The result was compared with D(X) result, and we found that the linear combination was better.

We also tried to use D(R(X)) as output function again to see if an optimal threshold would help. We found an optimal threshold for it in the validation set and compare the output function with D(X), but we found that the result is not as competitive as our D(X) model, and the reason for this has been discussed **above**. We tried some tricks to solve the problem but gained little effect.

Finally, we tested our model in a specially designed dataset[9] built from ImageNet, but we found that our result is poor, which indicated that our model is not robust enough for different datasets. This made us think of whether a general auto-encoder structure for all datasets could exist or not.

The testing results and discussions are in **Question 6**.

Question 5

The works are distributed mainly according to our weekly meetings' discussion results. We met for about 4 hours in average each week through Zoom, where we discussed papers, the progress and problems, shared our insights and ideas, and decided our work for that week. Then we took the part we're interested respectively concerning our available time. Another concern is computation resource that Ruofan didn't have a good GPU, so he didn't do training tasks.

As a result, slightly more than half (50%) of the workload and ideas were done by Keyang Qian. Ruofan Yang and Ke Ran did about 25% of the works, respectively.

The detailed distribution of our works is shown in the tabular below.

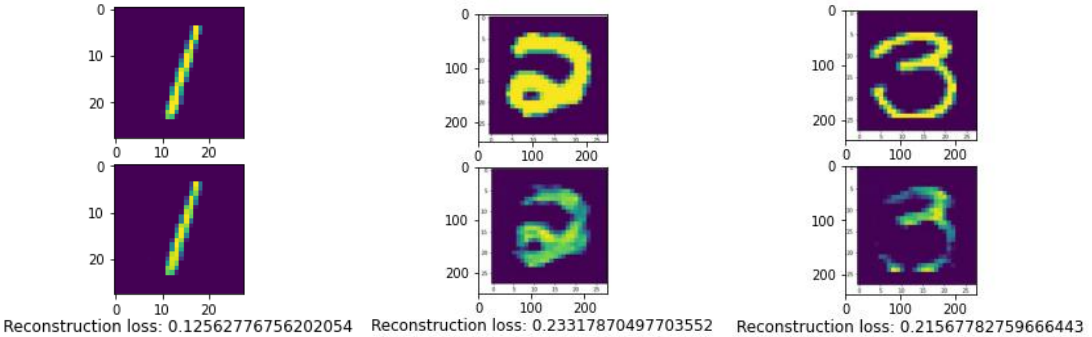
Names	Keyang Qian	Ruofan Yang	Ke Ran
Before week 1	Found the topic; searched for and discussed related papers in the field; learnt about GANs and pytorch; organized meetings.	Discussed the topic; read and discussed related papers in the field; learnt about GANs and pytorch; attended meetings.	Discussed the topic; read and discussed related papers in the field; learnt about GANs and pytorch; attended meetings.
Week 1	Organized meetings; searched for and discussed related papers in the field; came up with research ideas; discussed about model implementation.	Discussed problems in the meetings; searched for and discussed related papers in the field; came up with research ideas; discussed about model implementation.	Discussed problems in the meetings; searched for and discussed related papers in the field; came up with research ideas; discussed about model implementation.
Week 2	Organized meetings; wrote the model and model training part; wrote sample testing visualization and output functions part; did some training tasks; discussed the possible problems and came up with some improvement insight ideas.	Discussed problems in the meetings; wrote and improved some loss functions; extracted and selected some test sample images; discussed the possible problems;	Discussed problems in the meetings; wrote some test functions; improved the model and the model training part; did some training tasks; discussed the possible problems.
Week 3	Organized meetings; looked for insights into failures and applied residual connections and concatenation on the R network, then	Discussed problems in the meetings; did network combination attempts; printed out some intermediate layer	Discussed problems in the meetings; did many training tasks; implemented drop-outs and

	applied Wasserstein GAN primarily; tried many approaches to solve the imbalance problem.	features and improved them.	batchnorms.
Week 4	Organized meetings; solved the existing problems and applied early-stop; tested the model in MNIST; tested the model on selected samples; visualized the results; evaluated our work with baselines and objectives; came up with further ideas;	Discussed problems in the meetings; did further attempts to combine networks together as output and tested them.	Discussed problems in the meetings; did many training tasks; tested the model in ImageNet dataset.
Report writing	Question 4 & 5 & 6 and parts of question 1 & 3 & 7.	Question 1 & 2 and part of question 3.	Question 8, majority part of question 3 and part of question 7.

Question 6

The main outcome is that we tried and developed a series of methods on our GAN model for novelty detection. We found that our initial model imitating a recent paper[1] in 2018 suffered extreme problem when training too many epochs, and we dug into the problem, found the intrinsic issue of the network and tried different ideas to solve it, including that we didn't find a paper using Wasserstein GAN which that may solve the problem in the root in novelty detection, then we applied it and found other problems, then kept on solving them, and found that this method did improve the results. We noticed that a paper has suggested that only use one of G&D networks as novelty detection method has its different pros and cons respectively, so we tried different ways to combine them together as the novelty detection output.

Figure 6.1 shows sample outputs in MNIST.



(1) (2) (3)

Figure 6.1: Novelty detection results of our model. Real (above graph of each subfigure) and fake (below graph of each subfigure) images, reconstruction losses and discriminator outputs. Target class is 1 (inlier).

We tested our model on MNIST dataset and compared our result with other's results in **Figure 6.2**. The comparison shows that our model is better than a traditional method[16] and slightly better than the recent paper[1] we imitate at the beginning (initially our result is worse than the paper results) using the same output function. This mainly proves that Wasserstein GAN is effective for novelty detector than traditional GANs, as discussed in **Question 4**.

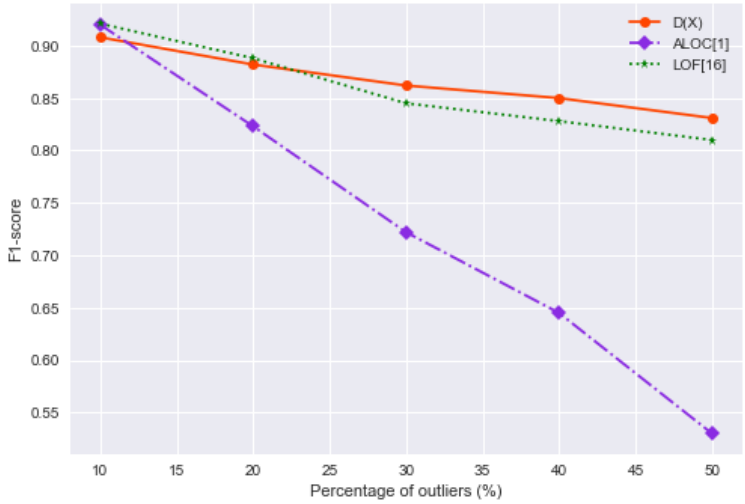


Figure 6.2: Comparisons of F1-scores on MNIST dataset for different percentage of outliers involved in the experiment. LOF[16] is a traditional method and ALOC[1] is a recent method in 2018 we initially imitated.

As we tried to combine both of the R & D networks together for our novelty detection task, we found optimal linear combination of their losses and optimal threshold in the validation set. The testing result was shown in **Figure 6.3** where we found that the linear combination result was better, but the difference became insignificant when the proportion of outlier is large. And the linear combination model requires a labelled open-set validation set, but usually for open-world learning tasks we don't have suitably labelled open-set. Therefore, such degree of improvement may not be very useful.

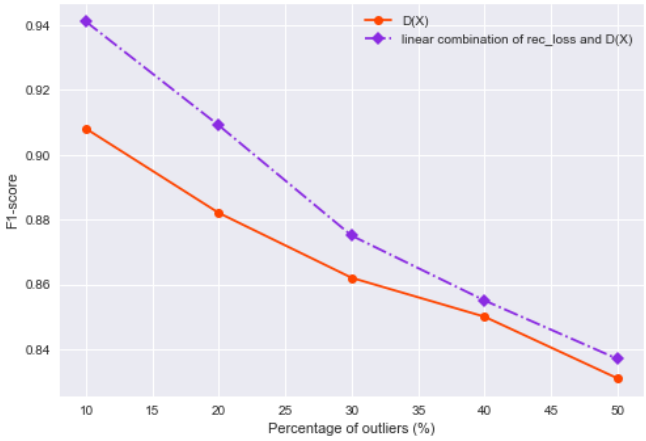


Figure 6.3: Comparisons of F1-scores on MNIST dataset in models using different output functions.

We also tried to use D(R(X)) as output function, the idea and failure reason was discussed in **Question 4** while failure was confirmed in **Figure 6.4**.

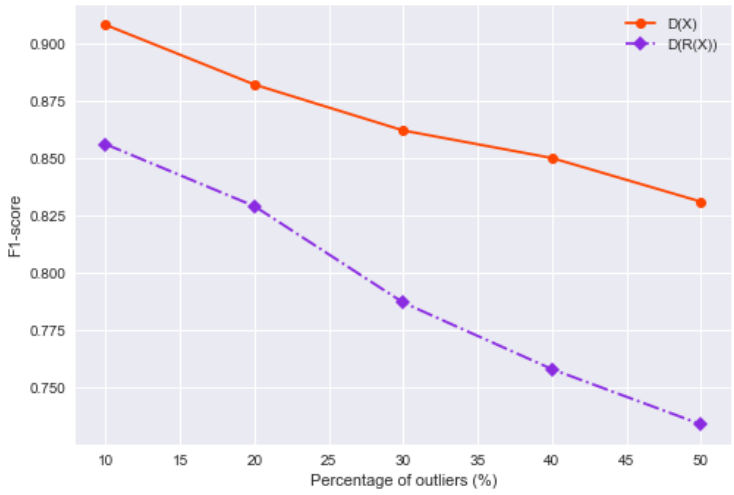


Figure 6.4: Comparisons of F1-scores on MNIST dataset in models using different output functions.

In addition, we tried to apply our model in a specially designed dataset[9] built from ImageNet (IN-125), but the result was really poor that F1-score was only 0.63 when the percentage of outliers is 50%, which indicates that our model did not work in this dataset and that our model is not robust enough for different datasets especially for complex novelty detection tasks. A scenario-based GAN network for specific dataset

task may be required.

Question 7

The poor test result for ImageNet (IN-125) indicates that our model is not robust enough for different datasets especially for complex novelty detection tasks. A scenario-based GAN network for specific dataset task may be required. It seems that a general auto-encoder structure for all datasets is not likely to exist. We wonder if we could let GAN learn feature from qualified k-class classifier's feature space that the classifier is designed for specific scenario and has collected useful features.

Although we finally get a well-trained GAN model, but the instability still exists in GAN network training process. But it would take too much time for training to validate a slight change or whole model estimation. Could we improve it? There is an inspiration from *P-KDGAN: Progressive Knowledge Distillation with GANs for One-class Novelty Detection*[21] that has the potential for improvement. The knowledge distillation is a reduction method that can transfer the generalization a bility of a large network to a light-weight network. To reduce the network training cost, applying the knowledge distillation on two GAN, the teacher GAN and the stud ent GAN would be helpful. The teacher GAN is the pre-trained model which could be the model well-trained before the slight change, and student GAN is trying to imitate the teacher GAN and improve each other.

With new loss function consist of weighted distillation loss and two GANs'own loss, there are two steps to train the simple structure of student GAN. First, use the pre-trai ned teacher GAN to teach the student GAN using the student loss and distillation loss. The second step is having two GANs trained together use both their own loss functio n and the distillation loss to reach the better stage. The first step could reduce the training effort for a student GAN which similar to teacher GAN model. In the second step we can compare the two improved GAN networks to check whether the slight change make sense or not easily and wisely.

Question 8

Through the works of our research, we gained much knowledge not only on novelty detection and GAN, but also learned how to do the research and represent it. We learned how the close-world environment used to train the agents is different from the real world. We knew that our the reason we do the research and the benefit them. Like novelty detection, only if we keep digging into it, more intelligent agents can be used in real life, thus fewer harm things will happen.

We also looked through many papers in this area, we learned that some papers use the latent representation to help characterize the data, some specified to detect one-class novelty and others even use another machine learning method. We also knew about the adversarial networks GAN, the component of it, the way it is used in novelty detection, and its benefit of it compare to other networks. We also use PyTorch to achieve a real GAN network, learn to build the generator and discriminator network, use the train data to train them, calculate different scores to compare our solution with the state of art, and use different images to show the outputs of our model. We also improve our models using many ways, including adding the Wasserstein loss function, changing the training rate of the two networks, setting thresholds to control the training progress, and so on. We even tried to draw the middle layers of the networks to gain understanding. Some of them succeed but surely some failed.

At last, we learned that only if we research the topic that we are interested in, we can have the enthusiasm to take time on it. We think that GAN is an interesting learning method, and it makes us insist study those knowledge that we haven't heard about before.

For this time, we think that our topic is too large and is such a mainstream direction, we should do more research and read more papers before we actually figure out the background knowledge we required and then determine the topic. Our research needs a deep understanding of machine learning and neural network knowledge and needs us to be familiar with Pytorch. This makes our team spend lots of time learning this background knowledge and even needed to learn the languages from start. As a result, we may choose the topic that the requirement conforms to our current level and is funnier and informal next time like other groups take focus on the video games and board games. Moreover, our initial plan is to improve the model we chose as the basic, but it evolved into improving our current model as there are many problems with the code construction the model and the result is much worse than how it is described. Next time, we may directly change a new model that is relatively good and then adds our thoughts on it.

References

[1] Sabokrou, M., Khalooei, M., Fathy, M., & Adeli, E. (2018). Adversarially learned one-class classifier for novelty detection. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3379-3388).

[2] Y. Xia, X. Cao, F. Wen, G. Hua, and J. Sun. Learning discriminative reconstructions for unsupervised outlier removal.In *ICCV*, pages 1511–1519, 2015.

[3] Kliger, M., & Fleishman, S. (2018). Novelty detection with gan. *arXivpreprint arXiv:1802.10560*.

[4] Kong, S., & Ramanan, D. (2021). Opengan: Open-set recognition via open data generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 813-822).

[5] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.

[6] Perera, P., Nallapati, R., & Xiang, B. (2019). Ocgan: One-class novelty detection using gans with constrained latent representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2898-2906).

[7] Zhang, Z., Chen, S., & Sun, L. (2020). P-kdgan: Progressive knowledge distillation with gans for one-class novelty detection. *arXiv preprint arXiv:2007.06963*.

[8] Kerner, H. R., Wagstaff, K. L., Bue, B. D., Wellington, D. F., Jacob, S., Horton, P., ... & Ben Amor, H. (2020). Comparison of novelty detection methods for multispectral images in rover-based planetary exploration missions. *Data Mining and Knowledge Discovery*, 34(6), 1642-1675.

[9] Burlina, P., Joshi, N., & Wang, I. (2019). Where's Wally now? Deep generative and discriminative embeddings for novelty detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 11507-11516).

[10] Zhang, Z., Dong, Y., Peng, H., & Chen, S. (2021). New Perspective on Progressive GANs Distillation for One-class Novelty Detection. *arXiv preprint arXiv:2109.07295*.

[11] Salehi, M., Arya, A., Pajoum, B., Otoofi, M., Shaeiri, A., Rohban, M. H., & Rabiee, H. R. (2021). Arae: Adversarially robust training of autoencoders improves novelty detection. *Neural Networks, 144*, 726-736.

[12] Toron, N., Mourao-Miranda, J., & Shawe-Taylor, J. (2022). TransductGAN: a Transductive Adversarial Model for Novelty Detection. *arXiv e-prints*, arXiv-2203.

[13] Kerner, H. R., Wagstaff, K. L., Bue, B. D., Wellington, D. F., Jacob, S., Horton, P., ... & Ben Amor, H. (2020). Comparison of novelty detection methods for multispectral images in rover-based planetary exploration missions. *Data Mining and Knowledge Discovery*, 34(6), 1642-1675.

[14] Pidhorskyi, S., Almohsen, R., & Doretto, G. (2018). Generative probabilistic novelty detection with adversarial autoencoders. *Advances in neural information processing systems, 31*.

[15] Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In *International conference on machine learning* (pp. 214-223). PMLR.

[16] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof:identifying density-based local outliers. In *ACM sigmodrecord*, volume 29, pages 93–104. ACM, 2000.

[17] Y.Xia, X. Cao, F.Wen, G.Hua, and J.Sun. Learning discriminative reconstructions for unsupervised outlier removal. In *ICCV*, pages 1511–1519, 2015.

[18] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. *Advances in neural information processing systems*, *30*.

[19] Salimans, T., Zhang, H., Radford, A., & Metaxas, D. (2018). Improving GANs using optimal transport. *arXiv preprint arXiv:1803.05573*.

[20] Wang, H. G., Li, X., & Zhang, T. (2018). Generative adversarial network based novelty detection using minimized reconstruction error. *Frontiers of Information Technology & Electronic Engineering*, *19*(1), 116-125.