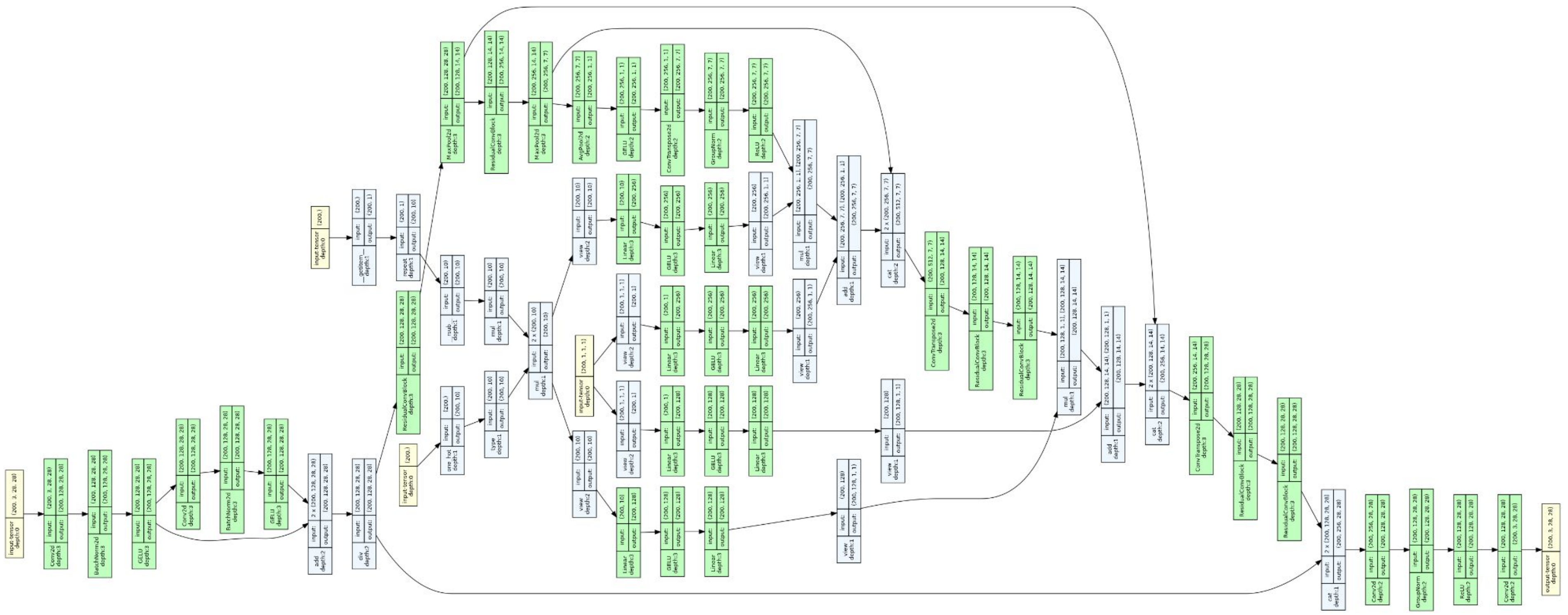
DLCV HW2 Report

NTU CSIE, R12922051
資工碩一 陳韋傑

Q1-1: Model Architecture

Please see the full size of model structure here

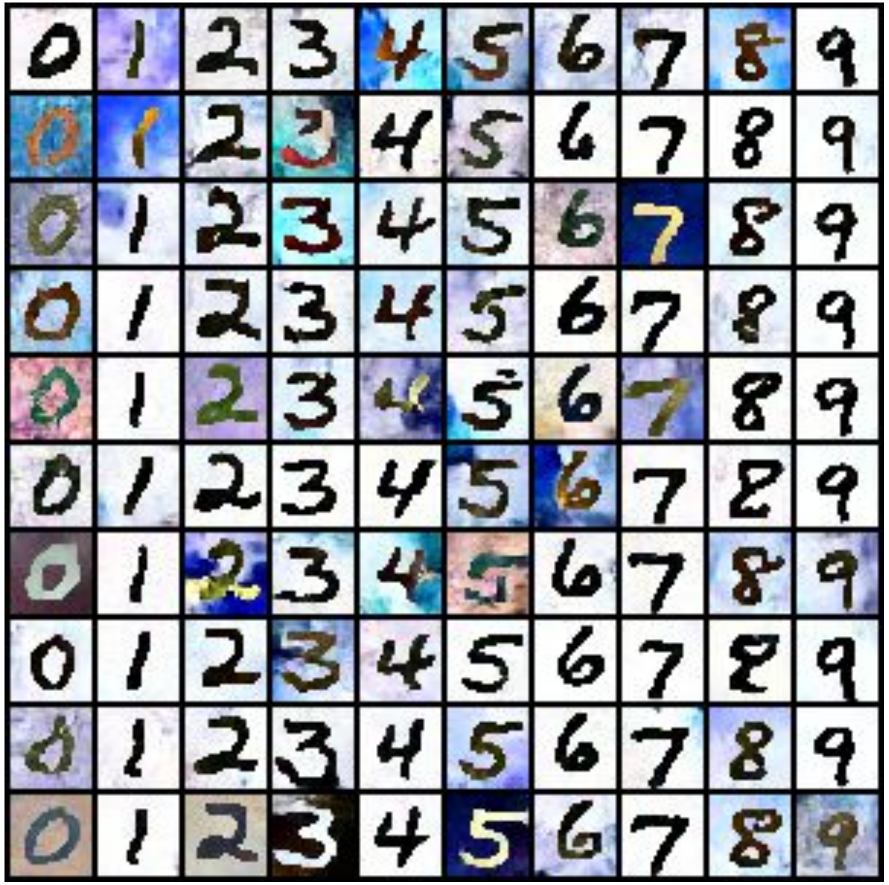


Q1-2: Generated Results

The generated 100 images are showed in the figure. By increasing the guidance coefficient*, we can get more accurate but less diverse results.

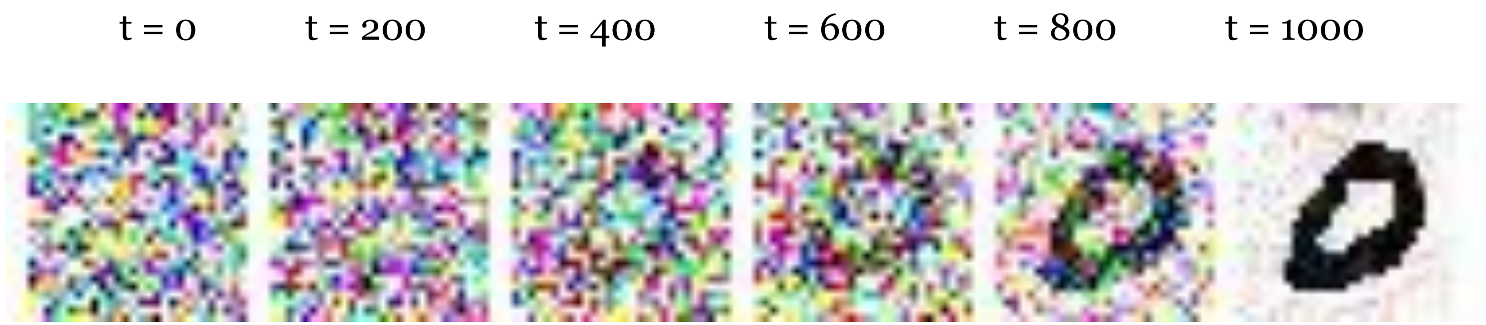
Classifier Accuracy: 0.992 (w = 5)

* Jonathan Ho et al. CLASSIFIER-FREE DIFFUSION GUIDANCE.



Q1-3: Diffusion Process

The reverse process of the first “0” (top-left in Q1-2) is showed in the following figure. We can see that the image is not clear until around 800 timesteps, and becomes much clearer at 1000 timesteps.



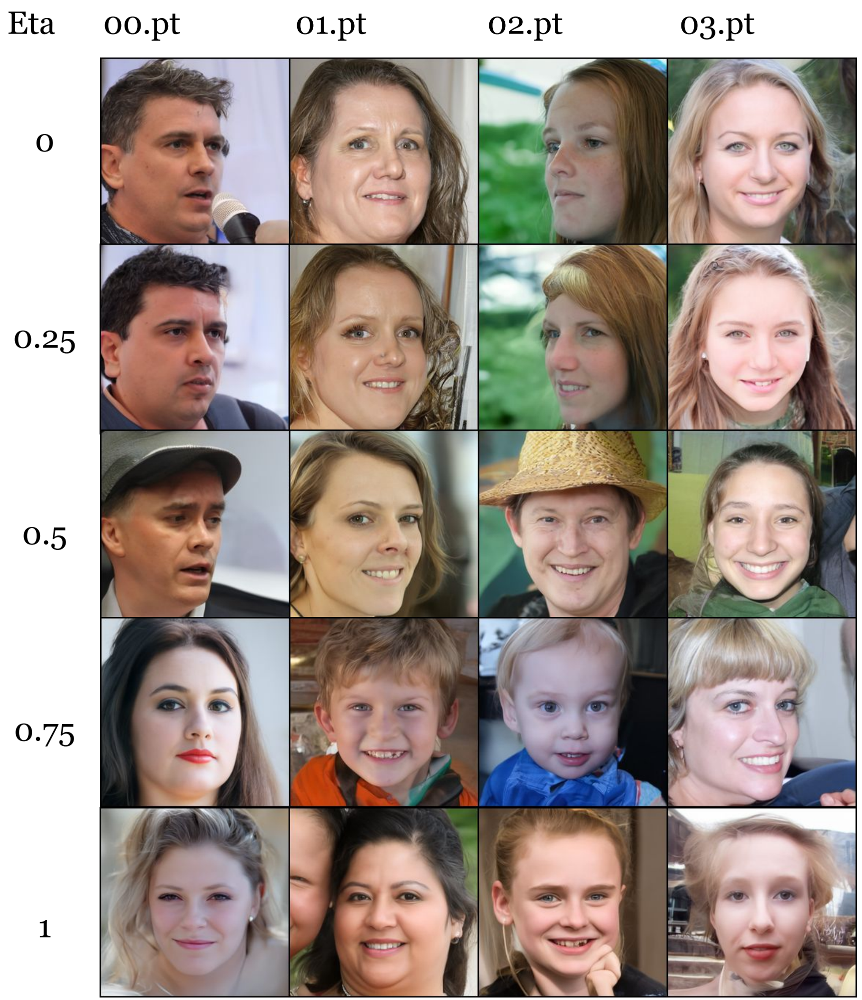
Q1-4: Discussion

- I learned how to add condition into a diffusion model and how to process it.
- The generation process of DDPM is slow. (That’s where DDIM comes in!)
- By adjusting the guidance coefficient (w), we can choose whether to get diverse but less accurate results (lower w) or to get accurate but less diverse results (higher w).

Q2-1: Eta

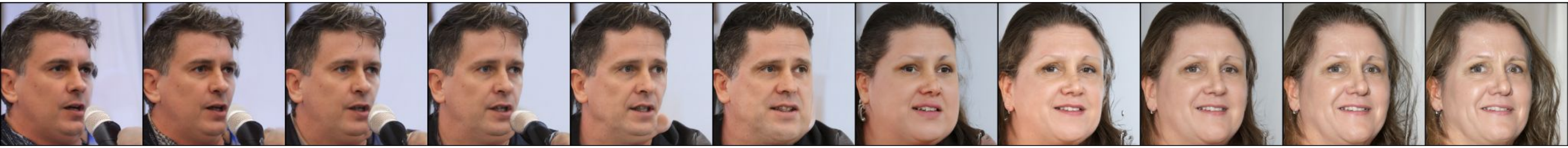
We can see that as eta increase, the generated faces are more random.

- Eta = 0, the generation process is determined
- Eta = 1, DDIM becomes DDPM

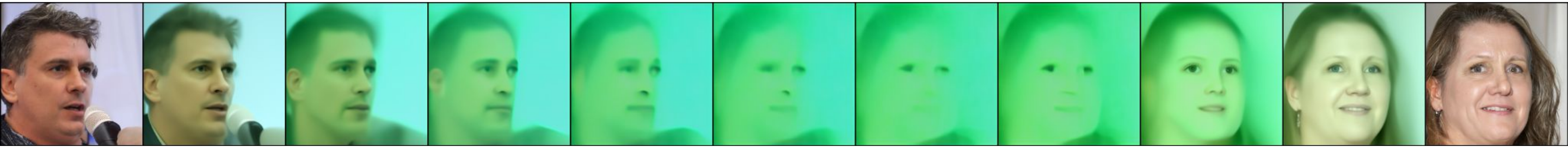


Q2-2: Interpolation

SLERP Interpolation



Linear Interpolation



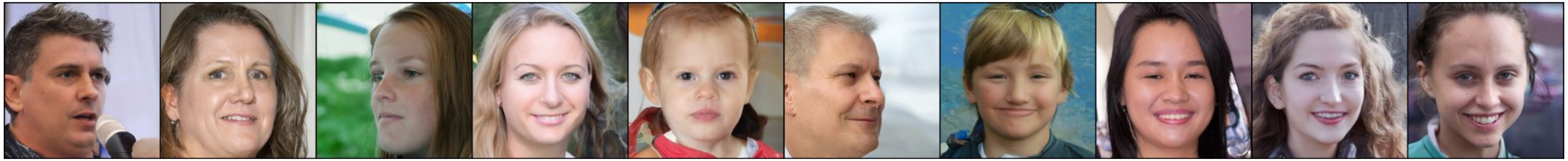
Linear interpolation isn't suitable for high-dimensional spaces with Gaussian or uniform priors. On the other hand, spherical linear interpolation (slerp), treating the interpolation as a circle path on an n-dimensional hypersphere, is commonly used and has shown promise in generative models. (As shown above)*

* Tom White. Sampling Generative Networks.

Q2 Appendix: Generated Faces by DDIM

Provided ground truth

Generated by myself



Can barely see the difference by human eyes.

MSE: 2.4668

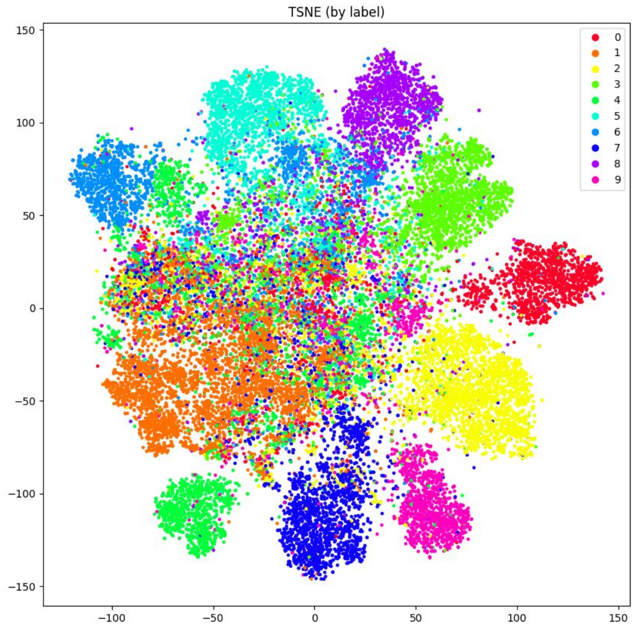
Q3-1: Results

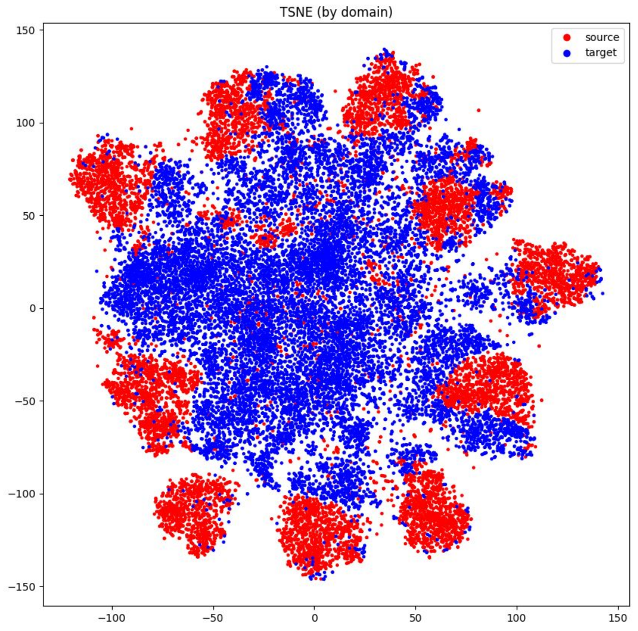
The hyperparameters are fixed through out all the experiments.

Model/Task performance (Validation Accuracy)	MNIST-M → SVHN	MNIST-M → USPS
Trained on source	19.83%	68.08%
Adaptation (DANN)	59.46%	84.81%
Trained on target	93.18%	98.92%

Q3-2: Visualization with t-SNE (MNIST-M → SVHN)

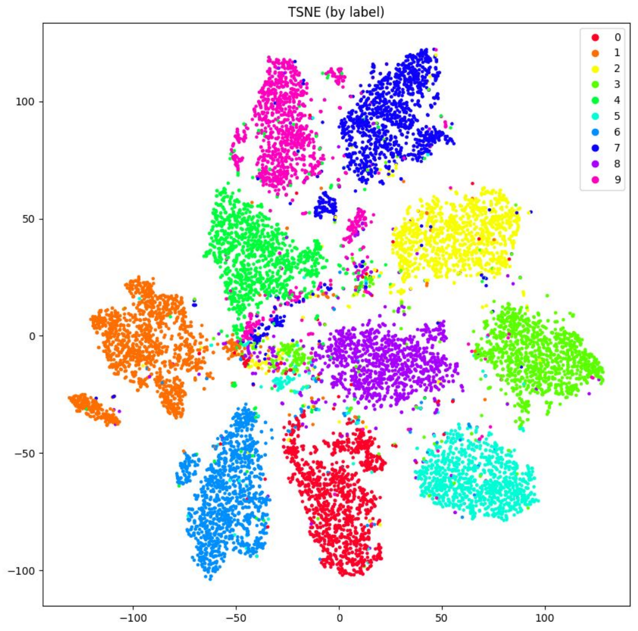
We can see that two domains are partially merged in the right figure (but not so good), also the extractor capture the feature of images successfully, as we can see some clusters starting to form in the left figure.

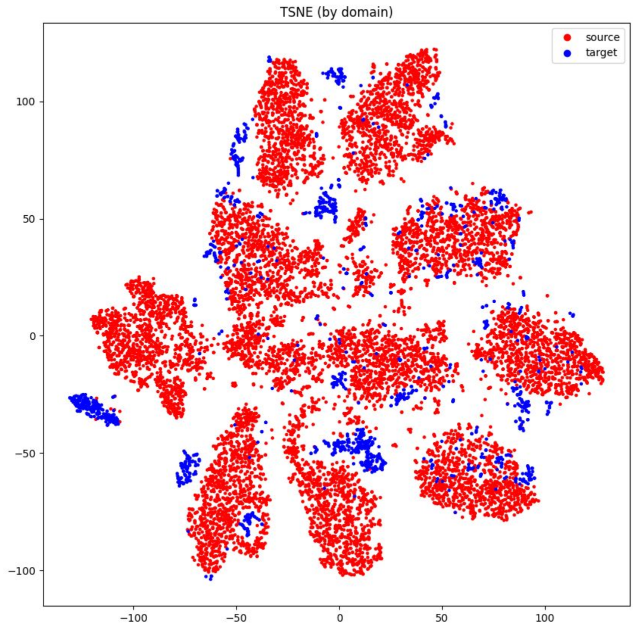




Q3-2: Visualization with t-SNE (MNIST-M → USPS)

We can see that two domains are merged better than previous task in the right figure, also the extractor did a great job on capturing the feature of images and we can see clear clusters in the left figure.





Q3-3: Implementation

- Epoch: 20
- Learning Rate: 5e-3
- Momentum: 0.9
- Batch Size: 16
- Loss: Cross Entropy
- Loss Ratio: 0.5*

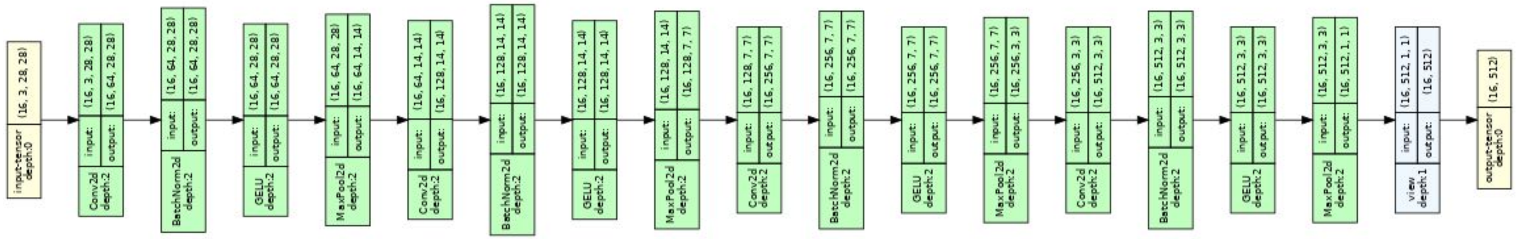
- Optimizer: SGD
- Scheduler: Same as reference code
- Preprocessing: Normalization by mean and standard deviation of each dataset respectively

*The ratio between domain loss and classification loss in DANN, the final loss is calculated by 2 * (loss_ratio * domain_loss + (1 - loss_ratio) * class_loss)

Q3-3: Implementation

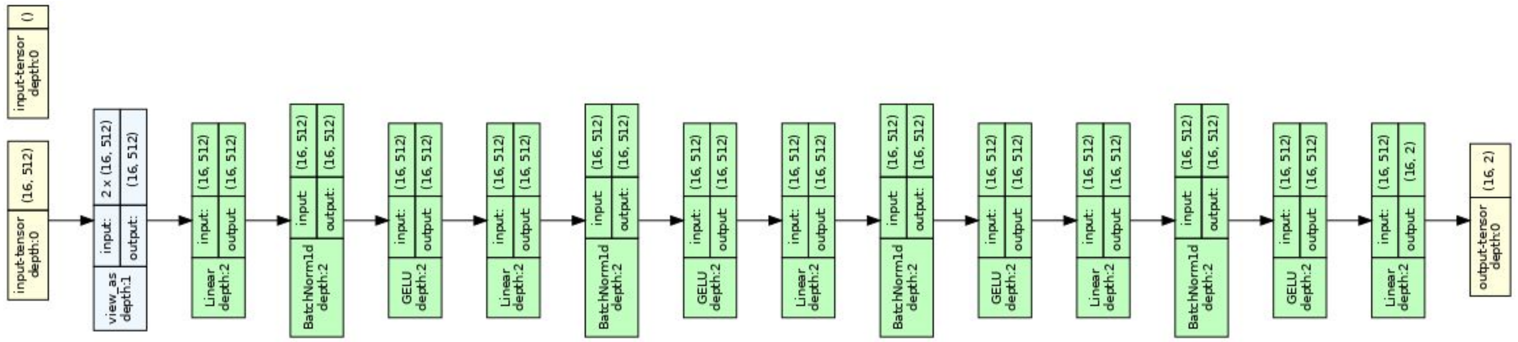
(See the full size of model structure here)

Extractor



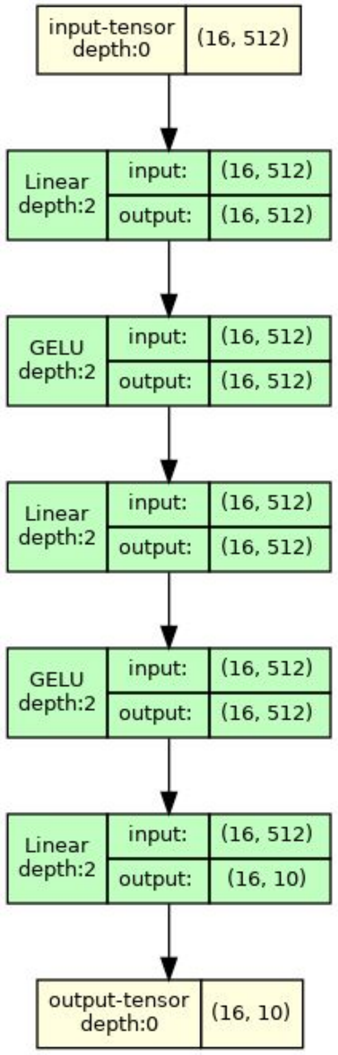
input-tensor depth:0 (16, 3, 28, 28)
Conv2d depth:2 input: (16, 3, 28, 28) output: (16, 64, 28, 28)
BatchNorm2d depth:2 input: (16, 64, 28, 28) output: (16, 64, 28, 28)
GELU depth:2 input: (16, 64, 28, 28) output: (16, 64, 28, 28)
MaxPool2d depth:2 input: (16, 64, 28, 28) output: (16, 64, 14, 14)
Conv2d depth:2 input: (16, 64, 14, 14) output: (16, 128, 14, 14)
BatchNorm2d depth:2 input: (16, 128, 14, 14) output: (16, 128, 14, 14)
GELU depth:2 input: (16, 128, 14, 14) output: (16, 128, 14, 14)
MaxPool2d depth:2 input: (16, 128, 14, 14) output: (16, 128, 7, 7)
Conv2d depth:2 input: (16, 128, 7, 7) output: (16, 256, 7, 7)
BatchNorm2d depth:2 input: (16, 256, 7, 7) output: (16, 256, 7, 7)
GELU depth:2 input: (16, 256, 7, 7) output: (16, 256, 7, 7)
MaxPool2d depth:2 input: (16, 256, 7, 7) output: (16, 256, 3, 3)
Conv2d depth:2 input: (16, 256, 3, 3) output: (16, 512, 3, 3)
BatchNorm2d depth:2 input: (16, 512, 3, 3) output: (16, 512, 3, 3)
GELU depth:2 input: (16, 512, 3, 3) output: (16, 512, 3, 3)
MaxPool2d depth:2 input: (16, 512, 3, 3) output: (16, 512, 1, 1)
view depth:1 input: (16, 512, 1, 1) output: (16, 512)
output-tensor depth:0 (16, 512)

Discriminator



input-tensor depth:0 ()
input-tensor depth:0 (16, 512)
view_as depth:1 input: 2 x (16, 512) output: (16, 512)
Linear depth:2 input: (16, 512) output: (16, 512)
BatchNorm1d depth:2 input: (16, 512) output: (16, 512)
GELU depth:2 input: (16, 512) output: (16, 512)
Linear depth:2 input: (16, 512) output: (16, 512)
BatchNorm1d depth:2 input: (16, 512) output: (16, 512)
GELU depth:2 input: (16, 512) output: (16, 512)
Linear depth:2 input: (16, 512) output: (16, 512)
BatchNorm1d depth:2 input: (16, 512) output: (16, 512)
GELU depth:2 input: (16, 512) output: (16, 512)
Linear depth:2 input: (16, 512) output: (16, 512)
BatchNorm1d depth:2 input: (16, 512) output: (16, 512)
GELU depth:2 input: (16, 512) output: (16, 512)
Linear depth:2 input: (16, 512) output: (16, 2)
output-tensor depth:0 (16, 2)

Classifier



input-tensor depth:0 (16, 512)
Linear depth:2 input: (16, 512) output: (16, 512)
GELU depth:2 input: (16, 512) output: (16, 512)
Linear depth:2 input: (16, 512) output: (16, 512)
GELU depth:2 input: (16, 512) output: (16, 512)
Linear depth:2 input: (16, 512) output: (16, 10)
output-tensor depth:0 (16, 10)

Q3-3: Discussion

- DANN is not easy to train. I’ve run many settings with different hyperparameters and model structures, only a few of them works. I believe not only the capability of extractor, discriminator, classifier needs to be similar or close, but also the hyperparameter setting plays a important role.
- If training failed, consider remove softmax layer after model output (or don’t add it in the first place), it will be easier for model to learn from logits than possibilities.

References

Q1: https://github.com/TeaPearce/Conditional_Diffusion_MNIST/tree/main

Q2: https://github.com/ermongroup/ddim/tree/main

Q3:

- https://github.com/NaJaeMin92/pytorch_DANN/tree/master
- https://colab.research.google.com/drive/1cTdIDT_fsBWGbaljhPSmBI6gwkw-tQ2H