Machine Intelligence and Learning

Indian Institute of Technology Delhi

COURSE ASSIGNMENT 3

Mayank Mishra 2016EE30506 Chahat Chawla 2016MT10492

Problem 1: GAN for Sampling from the Distribution of Handwritten Digits Data

We have used Generative Adversarial networks. Variational autoencoders have not been used because unlike generative adversarial networks, the second network in a VAE is a recognition model that performs approximate inference (which is not the case in GANs). GANs require differentiation through the visible units, and thus cannot model discrete data, while VAEs require differentiation through the hidden units, and thus cannot have discrete latent variables.

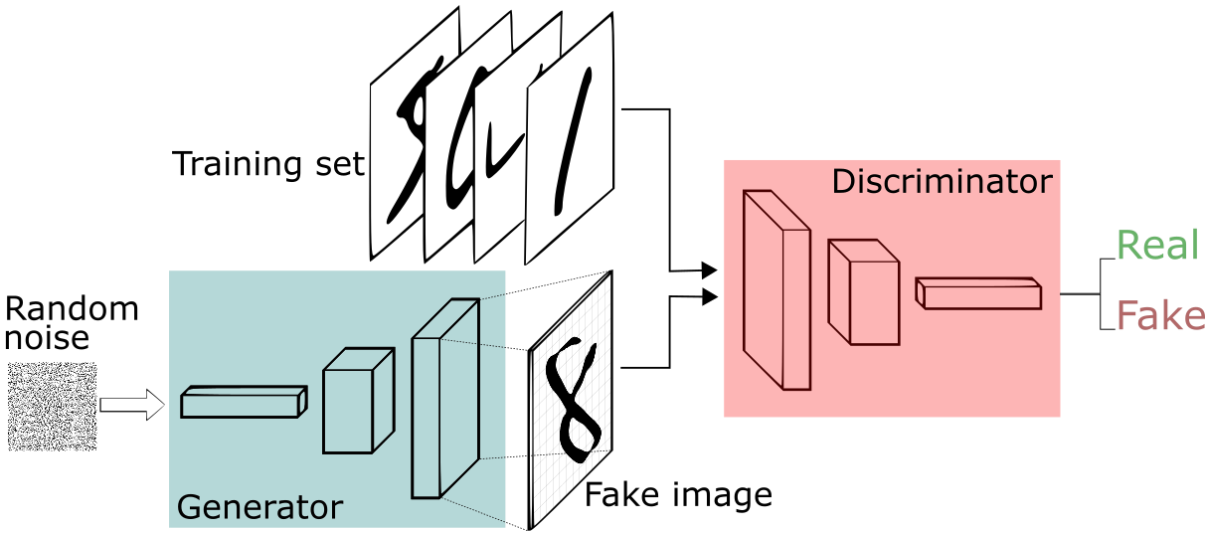
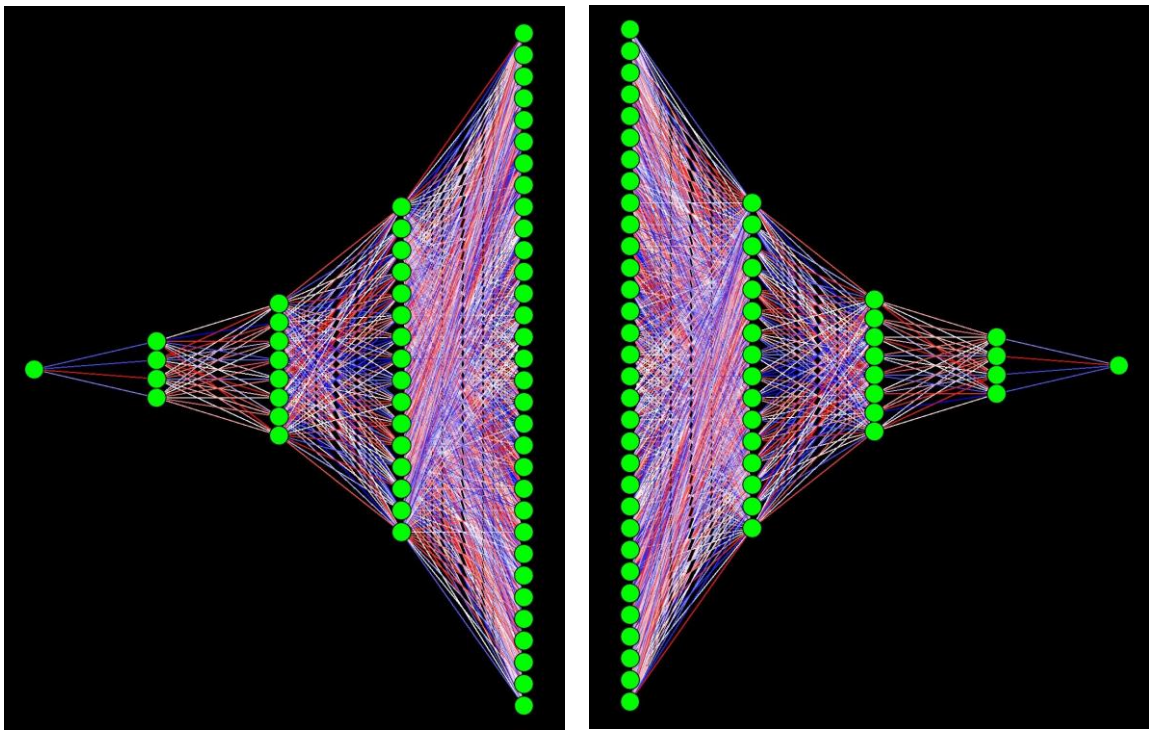


Fig. 1: GAN Architecture

Training in this case is a two-player minimax game with the Value Function V (D, G):

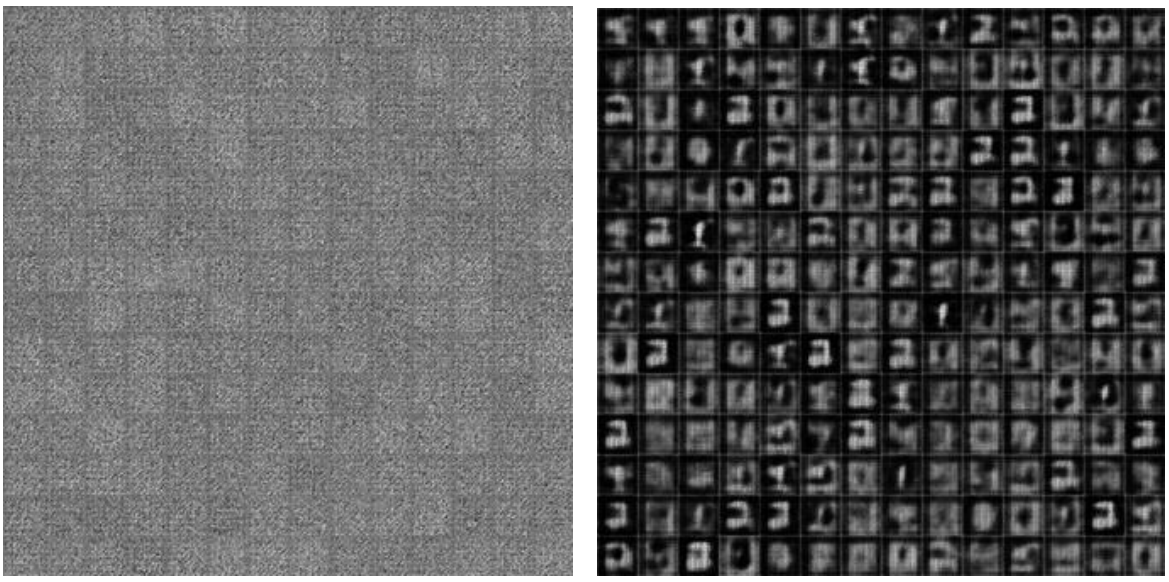
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$



Generator Discriminator

Fig. 2: Generator Discriminator Model employed for GAN for FC network

Both fully connected and ConvNets were tried for the discriminator and generator network. The images shown below have been trained using ConvNets for both Discriminator and Generator network, however.



Random Initialization Epoch 1

Epoch 5

Epoch 10

Epoch 50

Epoch 100

Fig. 3: Sampled Data at different epochs

Problem 2: Nearly MNIST

Objective: To train a model using the original MNIST dataset and achieve good accuracy on the handwritten dataset recorded in the classroom.

We use a ConvNet to solve the problem. We use the data sampled from different epochs of training the GAN, to train our Convnet. This ensures enough variability in the data, so that our ConvNet doesn't overfit to a particular dataset.

Though simple data augmentation can also achieve better accuracy but our model is more robust and noise tolerant due to being trained with examples generated from GAN. The data 'quality' is not good, as we conclude from a manual scanning of the data and hence Data Augmentation is unlikely to perform that well.

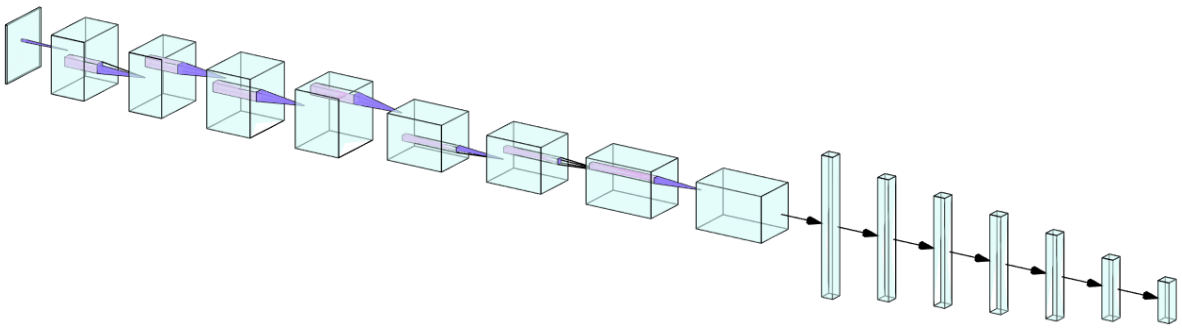


Fig. 4: ConvNet with 8 hidden Conv. Layers and 6 hidden FC layers

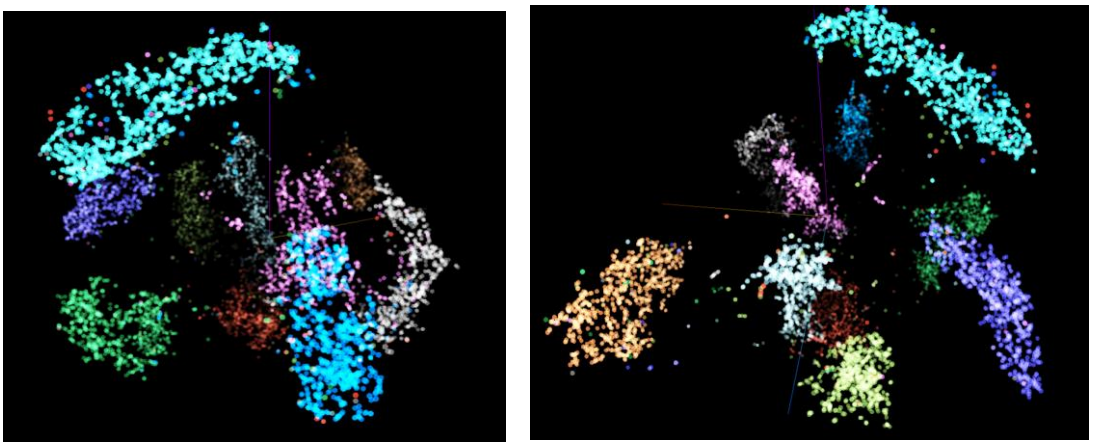


Fig. 5: TensorBoard Visualization for Original MNIST Dataset (using tSNE)

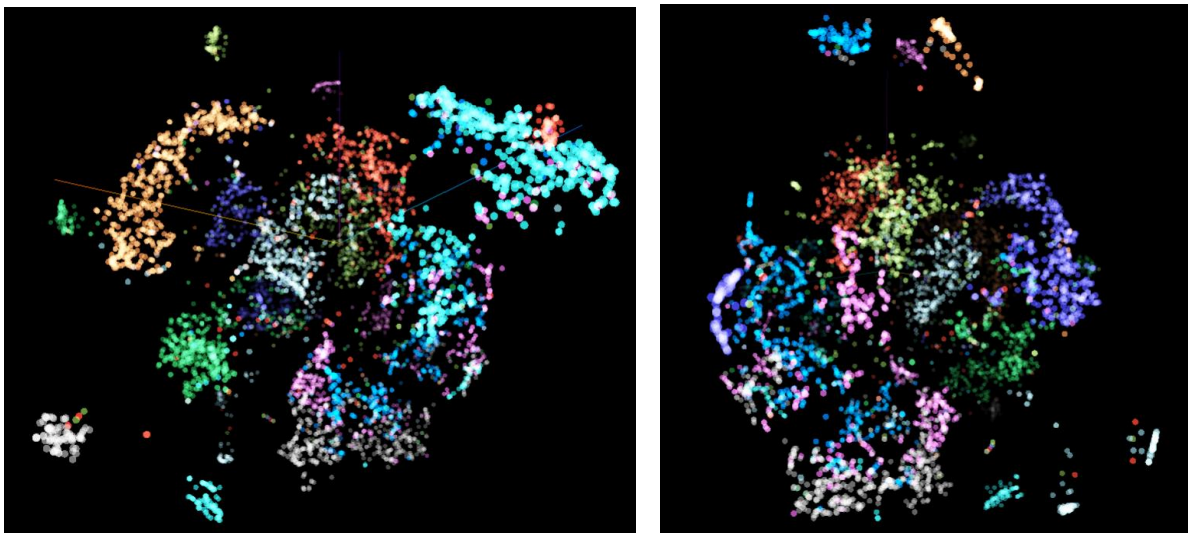


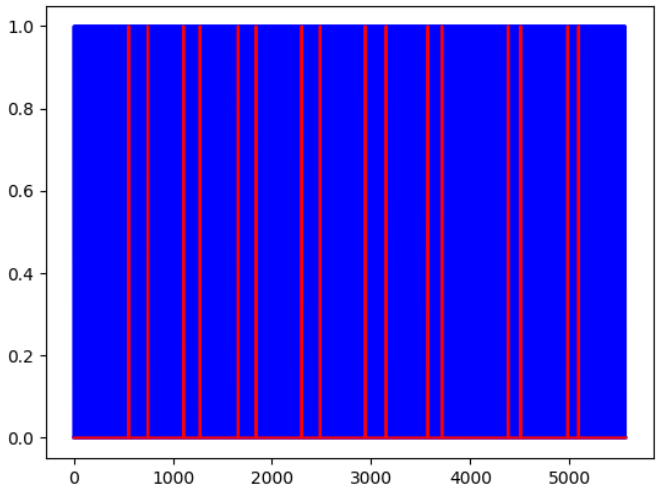
Fig. 6: TensorBoard Visualization for Kaggle Dataset (using tSNE)

Experimentation	Accuracy
ConvNet on original MNIST	89%
Ensemble methods on ResNet164, WideResNet28-10, VGG16 and MobileNet ***(*code used from internet)***	91.34%
ConvNet on data sampled from GAN	97.39%

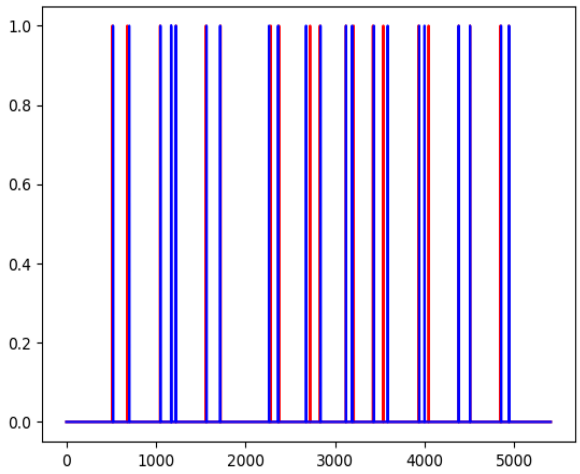
Table 1: Accuracy with different Algorithms

Problem 3: Instant Detection from Signal Data

We are not able to train neural networks since we have very less data (tried with windows of 100, 50 and even 10 with strides 10, 3 and 1). The neural network only learns to output 1 all the time.



What the neural network learns



Results via signal processing

Experiments	Identification rate	Miss Rate	False Alarm Rate	Variance
Signal processing	73.47%	7.11%	19.42%	23

Table 2: Various Accuracy Metrics with different Algorithms

Learnings from the Assignment

- We learnt that applied Deep Learning is a highly empirical process and requires wide experimentation.
- Data is the hunger for Deep Learning, and it is very much evident from the Instant Detection Problem. It could not perform well, and the algorithm was stuck at by-heartening a file in one of the experimentations.
- We also tried Traditional Signal Processing methods owing to the less amount of data, and they could perform comparatively well. This again remarks the high dependence of deep learning on the amount of data.
- Due to difference in the distributions of the learning data and the test data, we project the problem as Generative Adversarial Learning to counter this problem. It indeed gives remarkable improvement over typical Discriminative Models.

- We use generation using GAN to mimic the Data Augmentation, and this serves as Regularization. This further makes it noise and distribution insensitive and thereby generalizing even better.
- We use TensorBoard (with tSNE) Visualization, which builds comparative intuition for different distributions of data for the same problem.
- Difference in the distribution is evident from the TensorBoard Visualization.

References

1. Ian J. Goodfellow , Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio: **Generative Adversarial Nets**