**Name: Ankit Kumar Gupta**

**M.Tech SP-I**

**18145**

**Assignment-2.**

**DLCV 2021**

**Problem-1(RNN)**

**C**

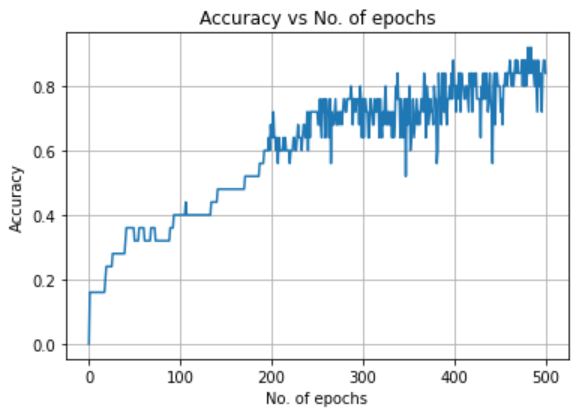
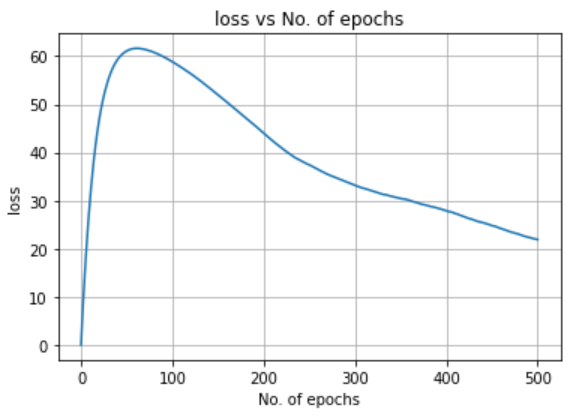
No. of recurrent layers used = 1.

No. of hidden neurons used (with bias) (initially) = 128.

Time Steps used for BPTT = 25.

Gradient Clipping threshold = 5.

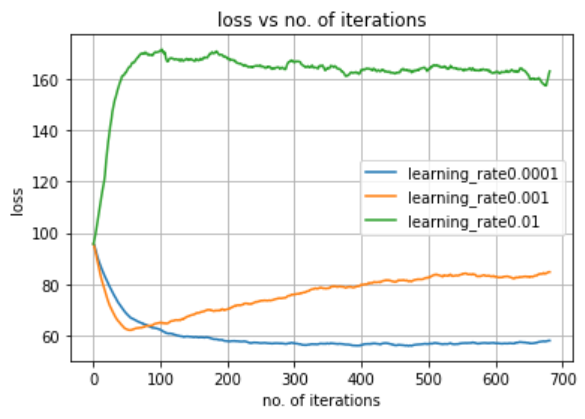
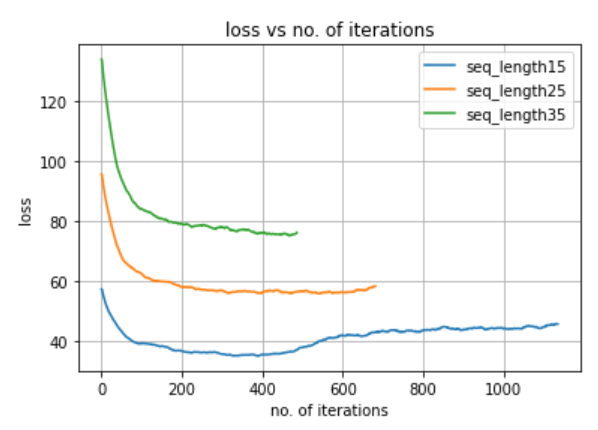
**D**

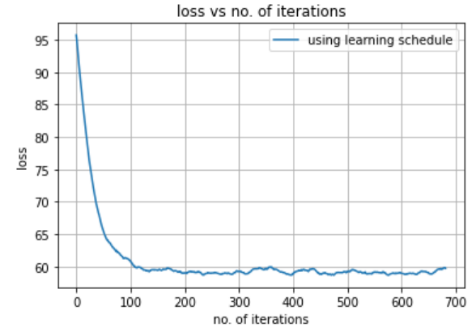
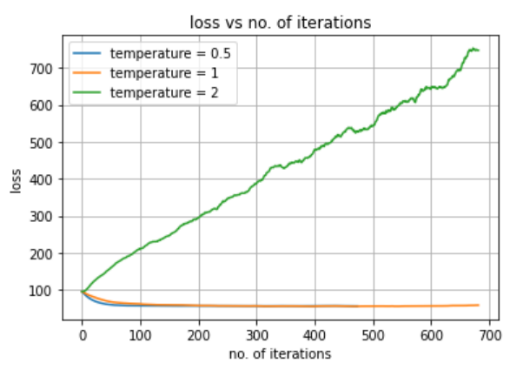
Plot of training accuracy and loss for small corpus:

Observation:

For small corpus, accuracy achieved is approximately 85%, loss is approximately 25. It is also seen that loss increases initially and then decreases.

**G**

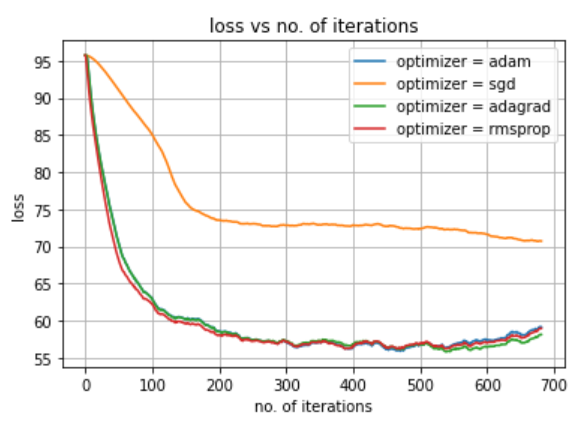
Effect of hyperparameters:



Observation:

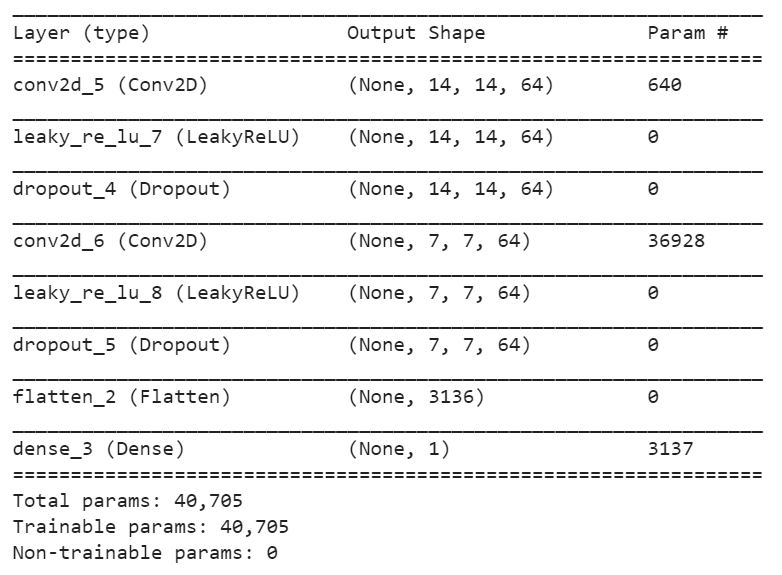
Learning rate 0.0001 performs well compared to others. For other learning rates, loss is exploding sooner or later. For lower length of sequence to be ravelled, the model learns better compared to others hence loss is less. For higher temperature loss keeps on increasing, suggesting that model is trying to diversify too much ending up with a huge loss, but other temperature loss is less. For learning schedule, in every 100 iterations learning rate is decreased by half, which results in a smoothened decrease in loss. For each experiment losses saturated after 3-4 epochs.

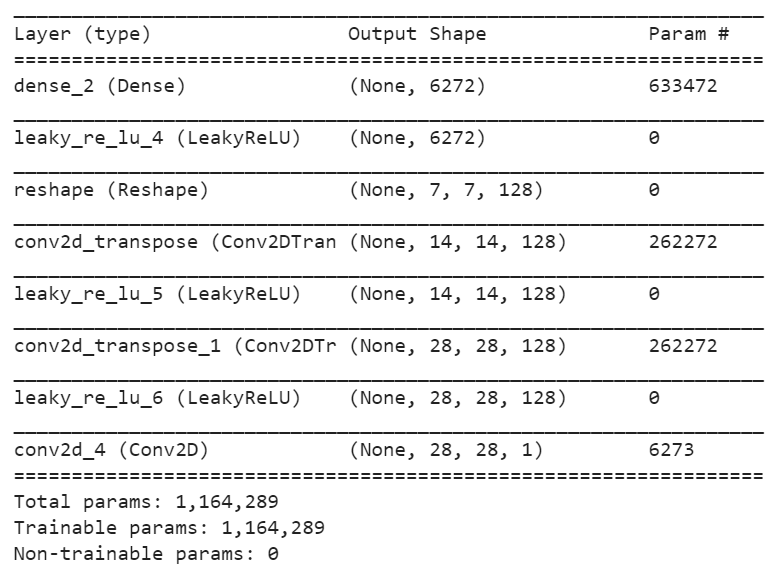
**I**

Here experiments with different optimizers is done. We can see, SGD performs poorer than others.

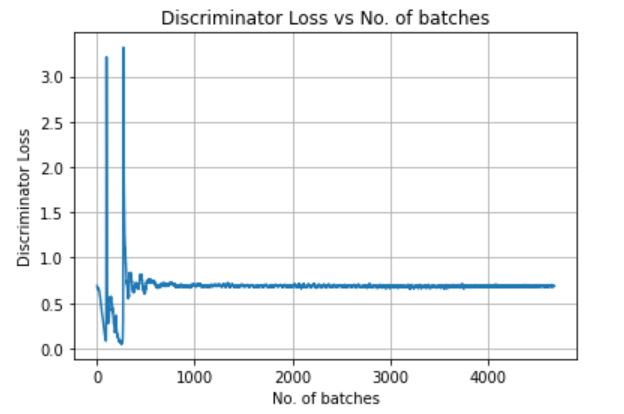
**Problem-2 (Generative Adversarial Networks)**

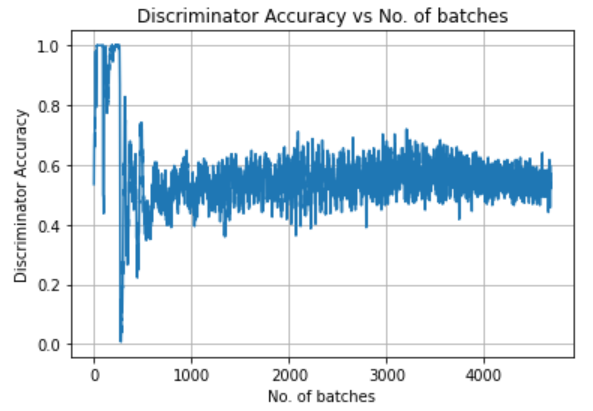
**A** Architecture of Generator Architecture of Discriminator





**B**

Plot of generator loss Plot of Discriminator loss

Plot of Discriminator Accuracy Observation:

It is observed that, discriminator accuracy at first is high, because images generated by generator is not of good quality initially and discriminator can easily classify it as a fake images. However, after generator has been trained, discriminator finds it difficult to classify fake & real images and it’s accuracy oscillates in the region [0.4,0.6].

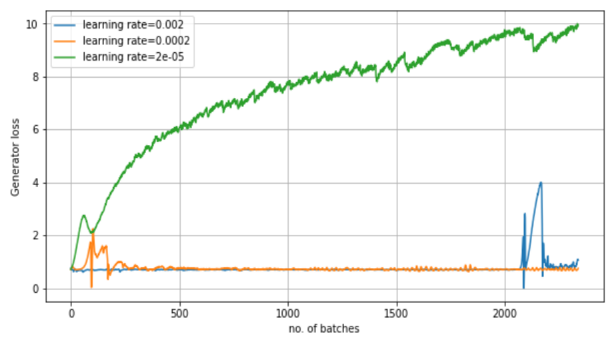
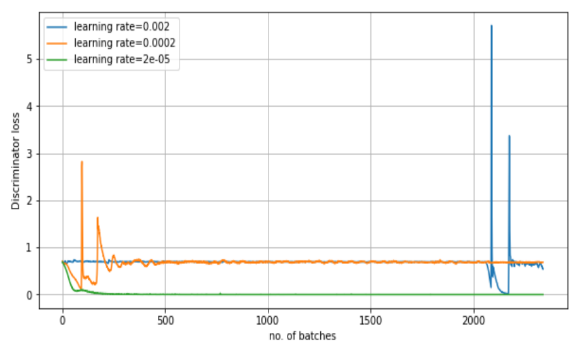
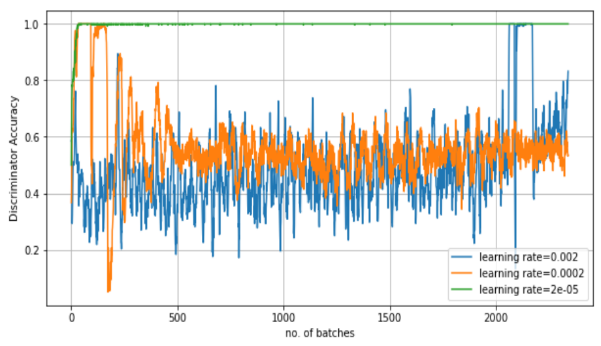
Also, losses of both generator and discriminator revolves around, 0.6 suggesting that a balance has been reached.

Generator has been trained, and discriminator finds it difficult to identify fake vs real images, hence low accuracy

Initially, generator is not trained, hence discriminator can easily identify fake images

**C**

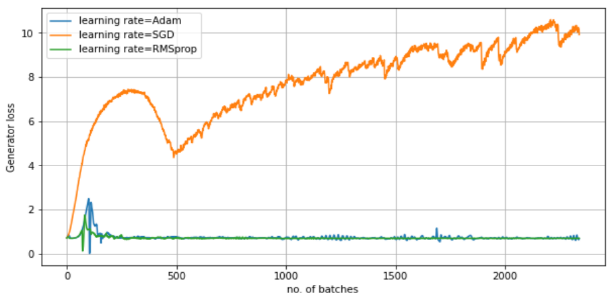
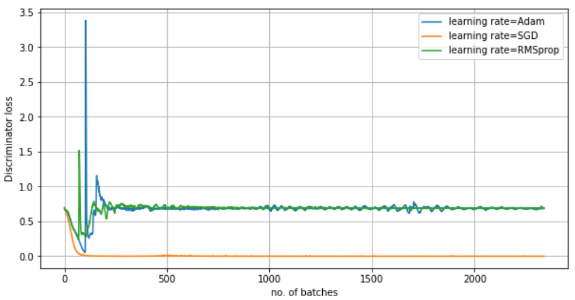
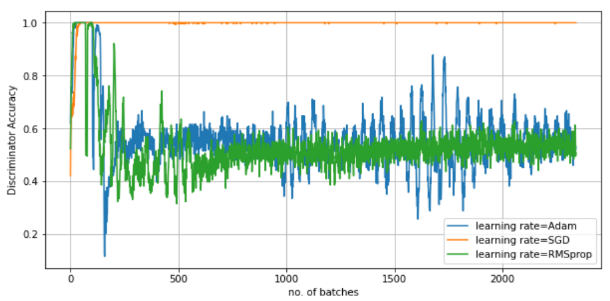
1.Learning Rate.

Generator Loss Discriminator Loss Discriminator Accuracy

Observation:

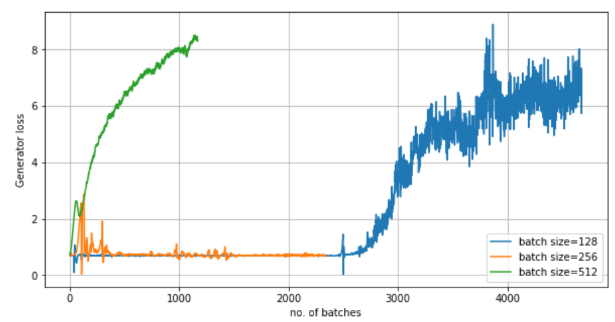
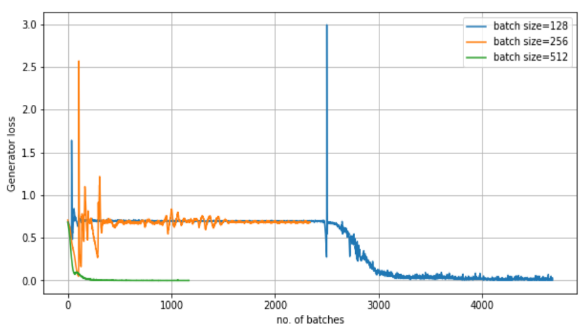
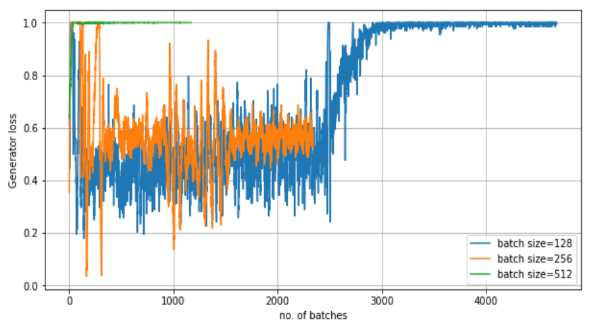
We can see that, for learning rate = 2e-05, the generator never actually learns which means it becomes easy for the discriminator to identify fake samples, hence we can see that discriminator loss decreases to 0 and accuracy spikes up to 1 for discriminator. For learning rate = 0.002, model works fine initially but at the later phase it behaves absurd suggesting that this learning rate is quite high, and the model bounces around local minima. Learning rate = 0.0002 acts as the best learning rate. (For all this learning rate Adam optimizer is used).

2.Optimizer.

Generator Loss Discriminator Loss Discriminator Accuracy

Observation:

As we know in general, Adam & RMSprop performs better than SGD, it is also evident in this model. The event which occurred in previous observation, happens in this as well i.e, when generator loss is high, discriminator’s loss is low and it’s accuracy is high. Also we can see that both RMSprop and Adam has a bit of similar performance in Gen. loss and Disc. Loss but the disc. Accuracy varies within a lesser bound than Adam’s suggesting that RMSprop optimizer yields stable results.

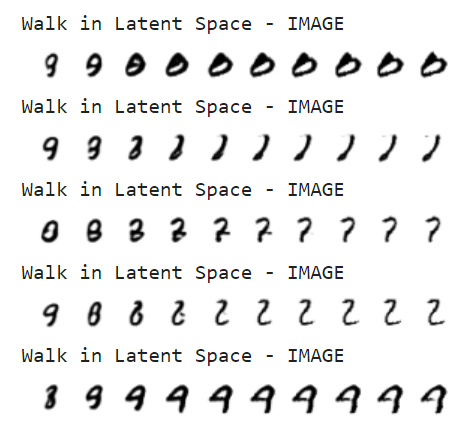
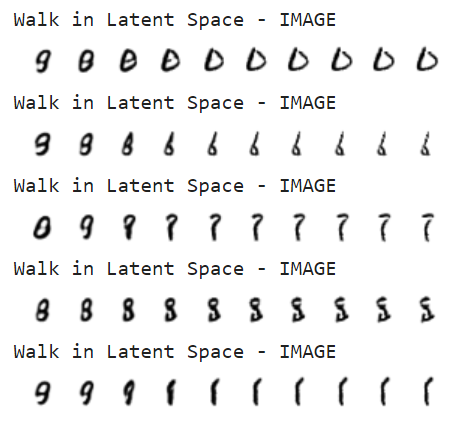
3. No. of Batches in an epoch

Observation:

Note: Here no. of epochs has been kept fixed (=10), so as we vary batch size, no. of iterations will also change. So for low batch size, no. of iterations will be more and vice versa.

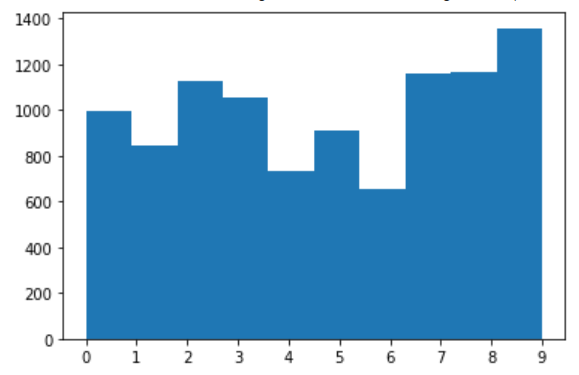
For batch size = 512, generator loss explodes, and similar event happens in disc. Loss and disc. acc as it happened in previous cases. For batch size =128, the model appears stable initially but later just explodes. Batch size = 256 performs far better than others and is stable in nature.

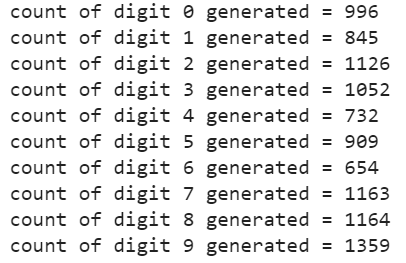
**D**

Traversing in latent space:

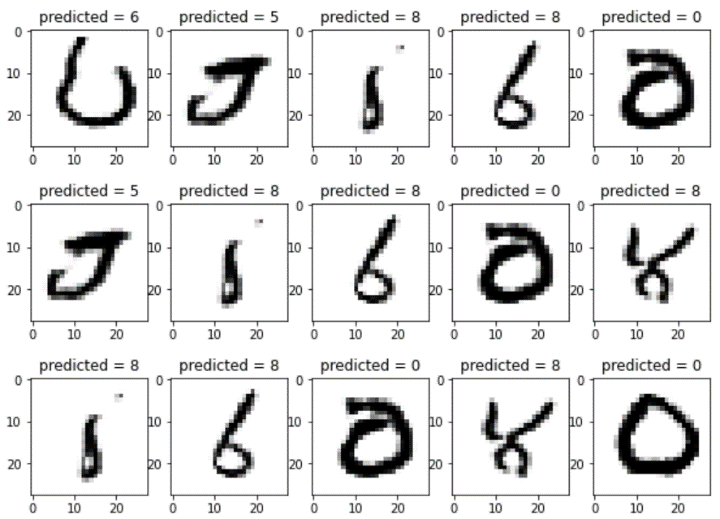
**E**

MNIST classifier has been built according to given configuration. Testing accuracy = 99.07%

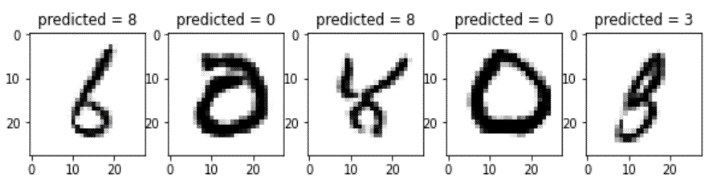
Distribution of 10,000 numbers generated by GAN.:



We can see that, model is producing more “9” digit and less “6” digit. It might have been possible that while training GAN model was not able to sufficiently distinguish this two digits.

Results on 20 random generated images:

Observation: We can see that, few of the images generated by GAN are not accurate. The MNIST classifier classifies them to closest matching digit.



**G**

GANs generally suffer from overconfidence, For example it uses very few features to classify an object. And, also in our case, GAN was producing more “digit 9”.To avoid the problem, we penalize the discriminator when the prediction for any real images go beyond 0.9 (*D(real image)>0.9*). This is done by setting our target label value to be 0.9 instead of 1.0. The images then obtained are presented as below.

Sampled Generated image.

