Assignment3

Tong Zhang *5/23/2022*

0.1 Part I: PyTorch LSTM

You can get the result by run the part1.ipynb in Part 1.

0.1.1 Task1

I implemented LSTM by nn.Linear.I complete the forward according to the formula.I need to set *os.environ* to avoid dynamic linklib error.

0.1.2 Task2

I train both the network as default parameters for 400 epoches.Input length =5,max_norms=10.0,learning rate =0.001

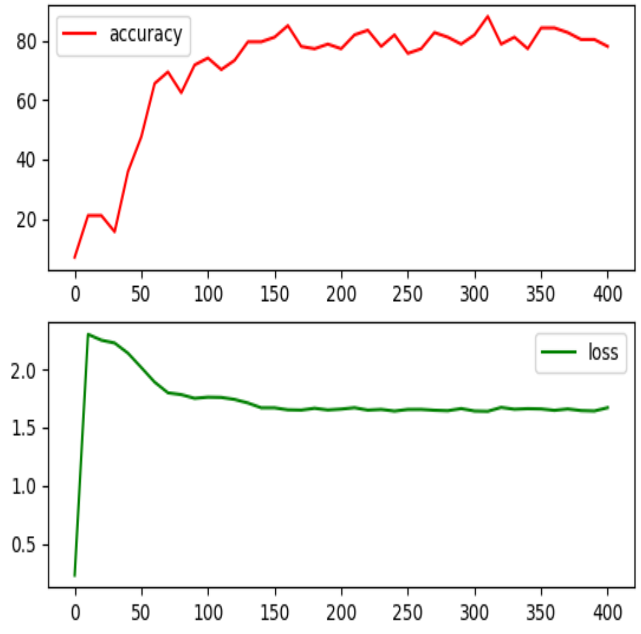


Fig. 1. lstm train for 400 epoch

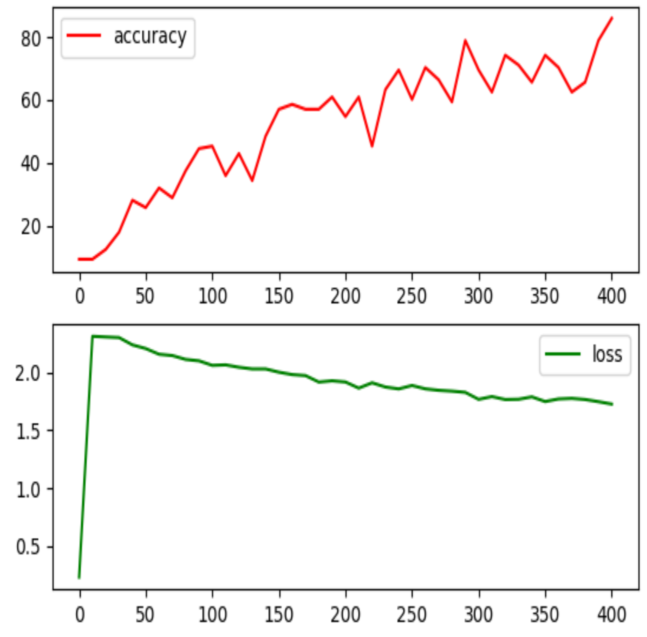


Fig. 2. vrnn train for 400 epoch

As we can see clearly,the lstm's accuracy grows much faster than vrnn and the loss also goes down quicker than vrnn.

0.2 Part II: Generative Adversarial Networks

You can get the result by run the part2.ipynb in Part 2.

0.2.1 Task1

I implement my GAN in my_gan.py.

1. I used torch.nn.Sequential to represent series of layers. The layers are linear layers and leakyrelu layers, where the negative slope is set to 0.2 the output should be non-linear, using tanh activation function.

2. The discriminator also consists of 2 layers: linera layers and leakyrelu layers, where the negative slope is set to 0.2 . Since the discriminator is used to classify if the input is real or fake ,I use sigmoid as output function.

Below is the loss of train my_gan for 200 epoch.

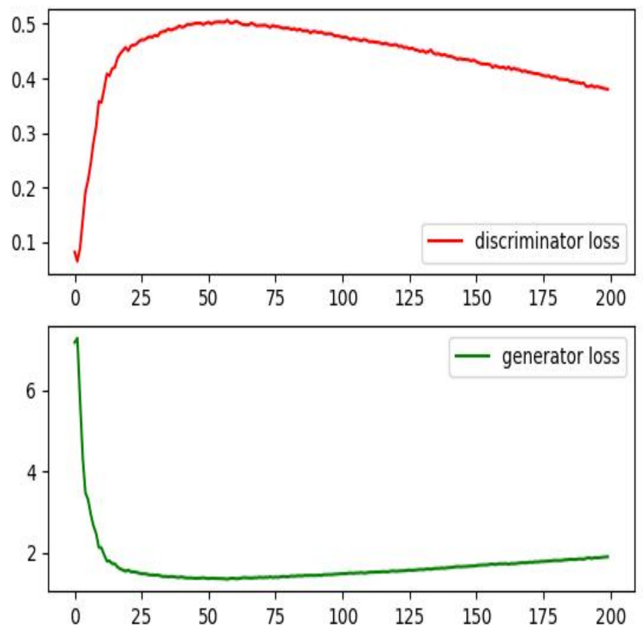
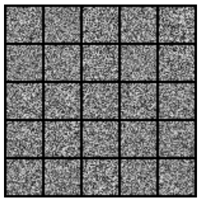


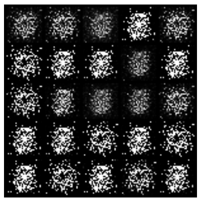
Fig. 3. gan train for 200 eopch

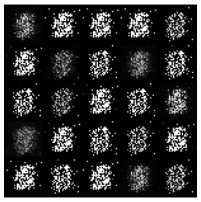
As we can see,the discriminator first go up then go down.Generator loss first go down then go up.

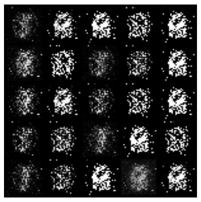
0.2.2 Task2

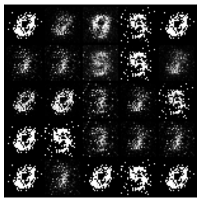
As we can see,In beginning, generated figures are random noise and we cannot tell figure from the images. During the process, we can tell some figures from the generated images but they are not very clear . At 100000 batch, most figures are clear.

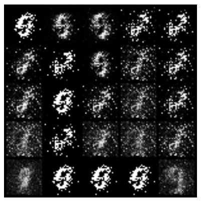

(a) origin

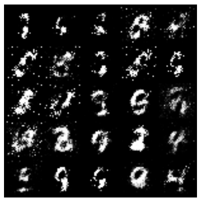

(b) 1000 batch

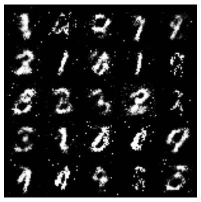

(c) 2000 batch


(d) 3000 batch


(e) 4000 batch


(f) 5000 batch


(g) 10000 batch


(h) 15000 batch

(i) 20000 batch

(j) 40000 batch

(k) 60000 batch

(l) 80000 batch

(m) 100000 batch

(n) 120000 batch

(o) 140000 batch

(p) 160000 batch

(q) 180000 batch

(r) 20000 batch

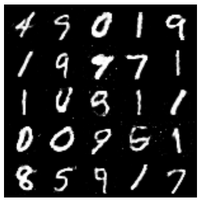
(s) 220000 batch

(t) 240000 batch

(u) 260000 batch

(v) 280000 batch

(w) 300000 batch


(x) 320000 batch

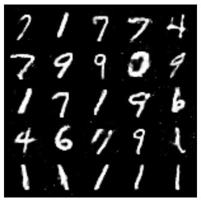
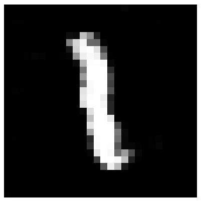
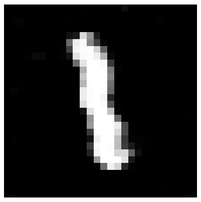

(y) complete train

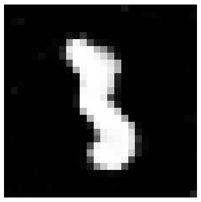
Fig. 4. pics

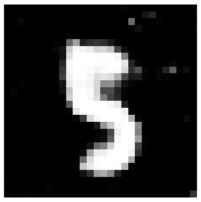
0.2.3 Task3

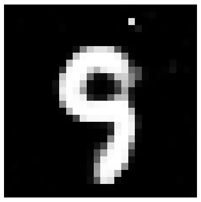
we generate two noises. We can use them as begin and end. To get the interpolate between these noises in latent space, we can gradually add the percentage of end from 0 to 1. At last, there will be nine images. Here are my experiment from 1 to 9:

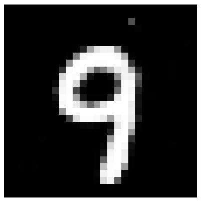

(a)

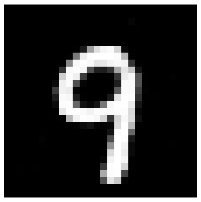

(b)

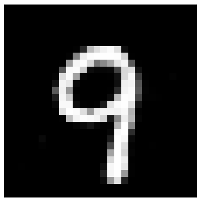

(c)


(d)


(e)


(f)


(g)


(h)

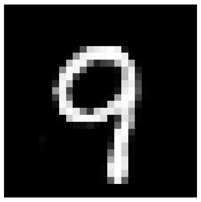

(i)

Fig. 5. pics