DLP HW4

多工所碩一 楊奕儒
ID：312553038

i. Introduction

In this lab, I implemented conditional video prediction in a VAE-based model. By taking pose images and source video as input, the model has the ability to predict the following future frames.

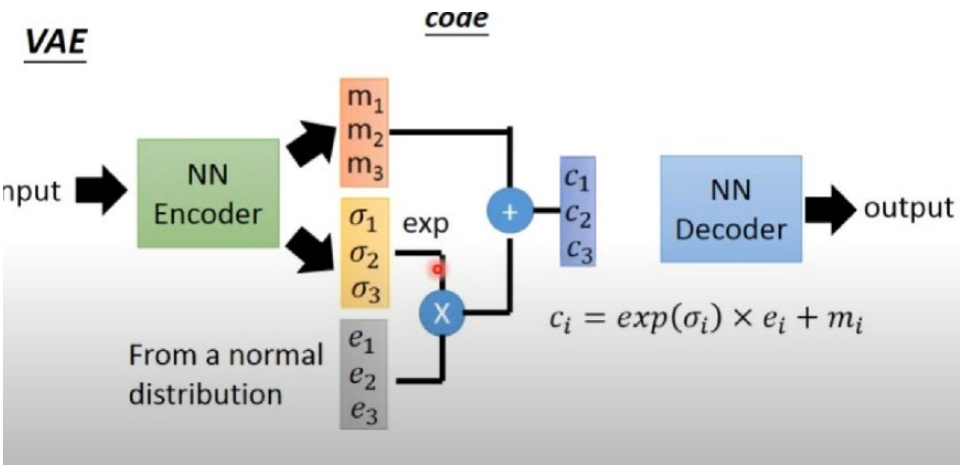
ii. Implementation details

1. How do you write your training protocol

To predict the succeeding frame, I initiated the prediction process by utilizing the present video frame and the concurrent pose image as inputs. These inputs were fed into the Gaussian_Predictor, resulting in the creation of a distribution denoted as 'z'. Subsequently, I harnessed the current pose image, the previously generated frame, and 'z' as inputs to facilitate the generation of the current frame in question.

Following the generation of the subsequent frame, a comparative analysis with the ground truth counterpart was conducted, thereby enabling the calculation of the reconstruction loss through employment of the mean squared error (MSE) criterion. In parallel, the Kullback-Leibler (KL) divergence was computed through the utilization of the kl_criterion. The total loss for this step was determined as the combined sum of the reconstruction loss and the KL loss.

2. How do you implement reparameterization tricks



```python
def reparameterize(self, mu, logvar):
    # TODO
    var = torch.exp(logvar)
    std = var**0.5
    eps = torch.randn_like(std)
    return mu+eps*std
```

Base on the method in the graph, I got “std” by taking the exponential of half the “logvar”, Generate random noise eps from N(0, 1), and return mu + eps * std.

3. How do you set your teacher forcing strategy

```python
def teacher_forcing_ratio_update(self):
    # TODO
    if(self.current_epoch >= args.tfr_sde):
        slope = 1.0 / (args.num_epoch - args.tfr_sde)
        tfr = 1.0 - (self.current_epoch - args.tfr_sde)*slope
        self.tfr = min(1, max(0, tfr))
```

First, I checked whether the current epoch is greater than or equal to the epoch at which the teacher forcing ratio should start to decay. If the condition is met, the new teacher forcing ratio is calculated using the linear decay formula and remains within the range [0, 1].

4. How do you set your kl annealing ratio

```python
if args.kl_anneal_type == "Cyclical":
    self.L = np.ones(args.num_epoch) * 1.0
    self.i = 0
    period = args.num_epoch / args.kl_anneal_cycle
    step = (1.0 - 0) / (period * 0.5)
    for c in range(args.kl_anneal_cycle):
        v = 0.0
        i = 0
        while(v<=1.0 and int(i+c*period)<args.num_epoch):
            self.L[int(i+c*period)] = v
            v += step
            i += 1
```

If type is set to "Cyclical", I calculate the period of each cycle first. The ratio in front half of each cycle increases linearly form 0 to 1, and the other ratio remain to 1.

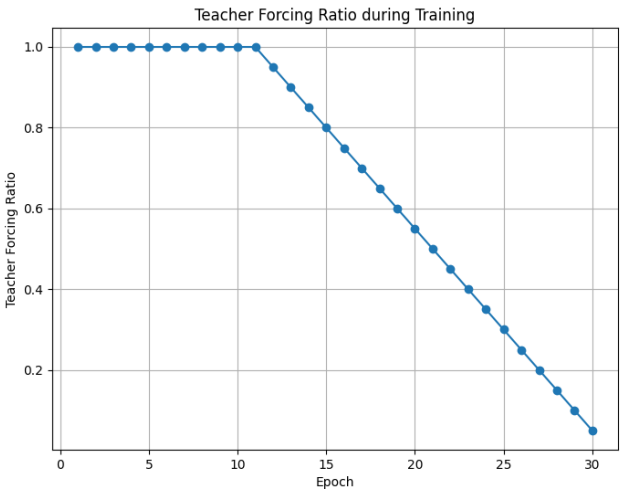
```
elif args.kl_anneal_type == "Monotonic":
    self.L = np.ones(args.num_epoch) * 1.0
    self.i = 0
    period = args.num_epoch / 1
    step = (1.0 - 0) / (period * 0.25)
    for c in range(1):
        v = 0.0
        i = 0
        while(v<=1.0 and int(i+c*period)<args.num_epoch):
            self.L[int(i+c*period)] = v
            v += step
            i += 1
else:
    self.L = np.ones(args.num_epoch) * 1.0
```

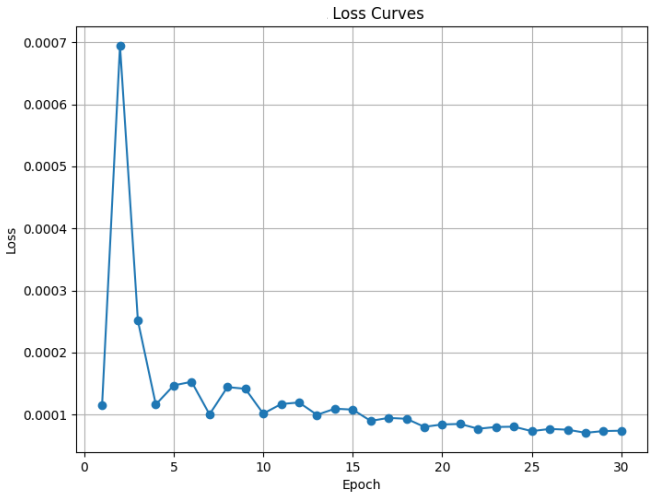
If type is set to "Monotonic", I set the ratio in front quarter of epoch numbers increases linearly form 0 to 1, and the other ratio remain to 1.

iii. Analysis & Discussion

1. Plot Teacher forcing ratio

a. Analysis & compare with the loss curve



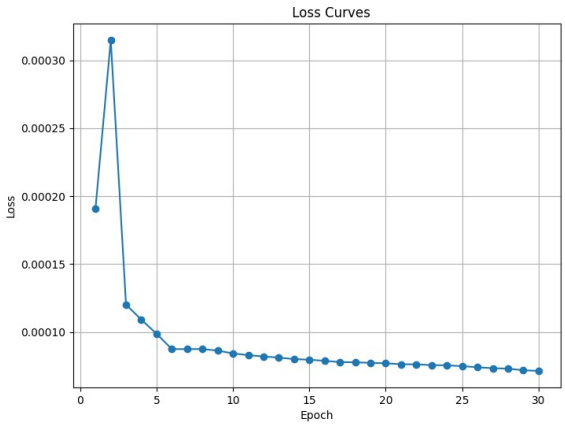


From these two plots, we can observe that as the teacher forcing ratio decreases, the loss also tends to become more stabilized.

2. Plot the loss curve while training with different settings. Analyze the difference between them

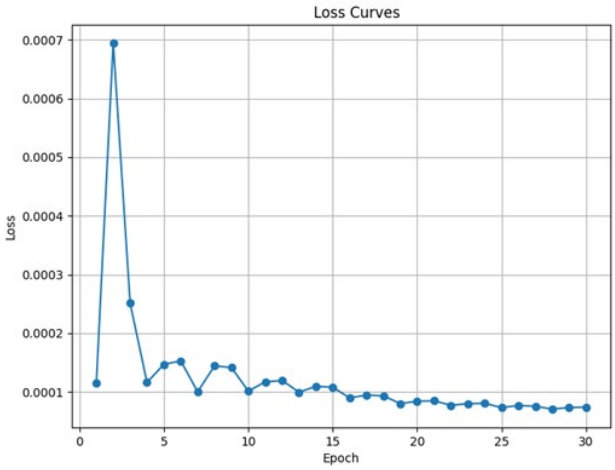
a. With KL annealing (Monotonic)

The training process appears to converge steadily, with a relatively smooth decrease in the loss function over iterations. This suggests a stable training regimen, potentially leading to improved convergence towards the optimal solution.



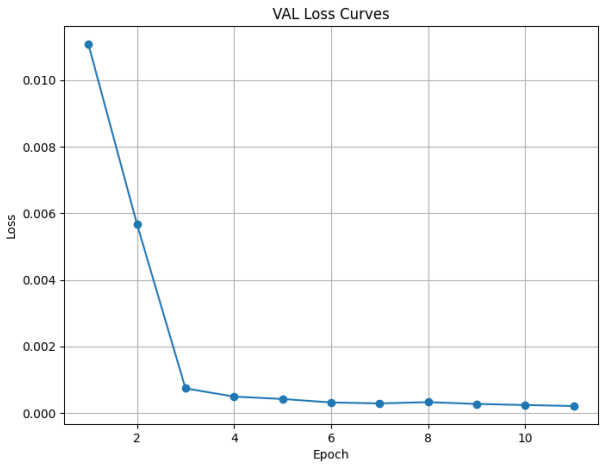
b. With KL annealing (Cyclical)

The training process seems to exhibit periodic oscillations in the loss function, indicating distinct cycles. This could be influenced by factors such as learning rate adjustments or variations in data batches. While this approach may introduce oscillations during the search for the optimal solution, it could also extend the training time.



c. Without KL annealing

It doesn't appear to converge as tightly as in the case of both approaches, and the convergence rate also seems to be slower.



3. Plot the PSNR-per frame diagram in validation dataset