Lab 4 - Conditional VAE for Video Prediction

TA 林于翔

- Assignment release: 2025/4/1 18:30
- Assignment announce: 2025/4/1 18:30
- E3 Deadline: 2025/4/15 23:59
- Kaggle Deadline: 2025/4/15 18:30
- Format:
    - Zip whole source code directory and named it in LAB4_{studentID}_{YOUR_NAME}.zip
    - Save your report as pdf file and named it in LAB4_{studentID}_{YOUR_NAME}.pdf
- About Kaggle
    - Link: https://www.kaggle.com/t/6efdaf4f77284d35ae5e1e5928745f6c
    - Team name: {student id}_{your name}
    - 1 person 1 team
    - 5 submission per day

I. Introduction

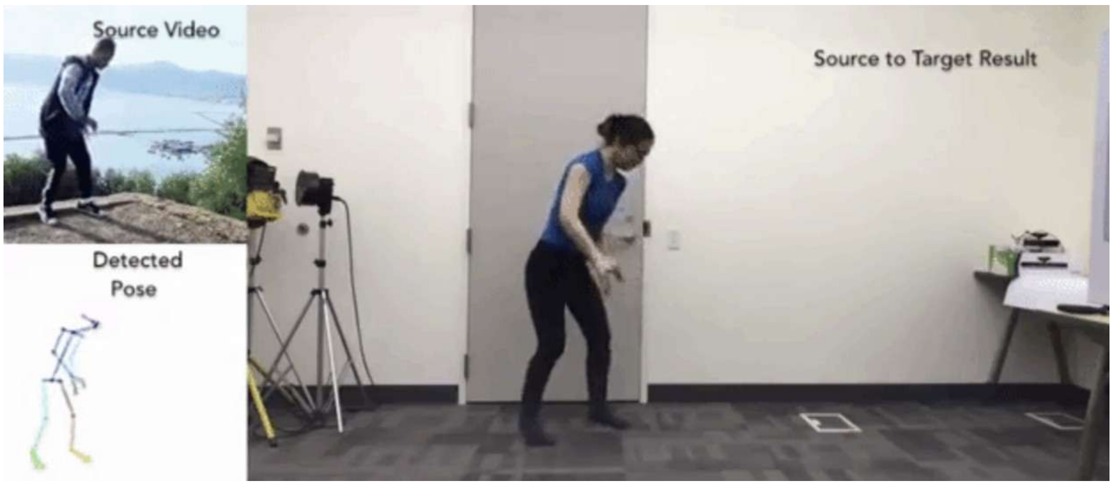


Figure 1. Taking the pose image (left down) that is generated by an off-the shelf pose estimation network as an input to generate the next consecutive frame (right image).

In this assignment, we need to implement conditional video prediction in a VAE-based model. Before we go through this LAB, I want to mention that the topic about this LAB is highly correlated to the ICCV paper [1] which makes the prediction in a GAN-based model and ICML paper [2] which makes the video prediction in the VAE-based model. It might be helpful to study these papers [1], [2] before doing your work.

In [1], the author posts an interesting approach that predicts the video clips in GAN model structure. By taking a pose image generated by a pre-trained pose estimation network and a previous frame, the model can generate the next consecutive frame with a comparable subjective quality. Further details can be found in [1].

In [2] , the author used a VAE-based model which combined the LSTM module to predict future frames in RNN manner. By taking two reference frames, the model has the ability to predict the following future frames. If you have NO ideas about how to perform video generation, reading this paper [2] will provide you great insights.

II. Lab Description

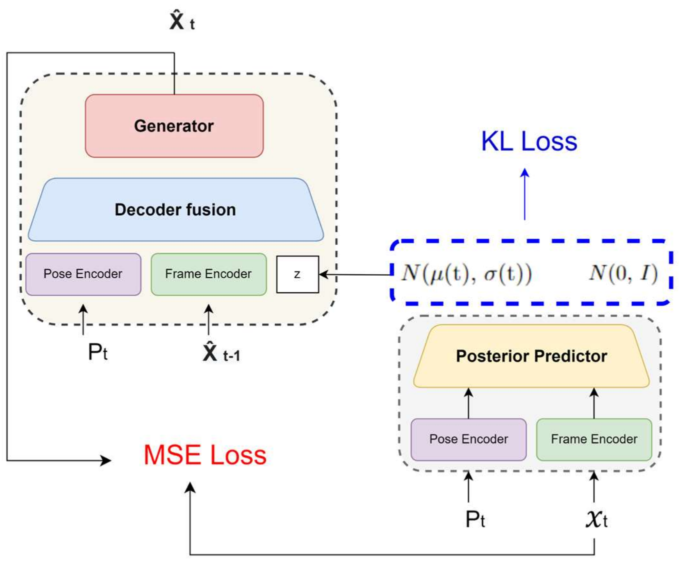


Figure 2. (a)

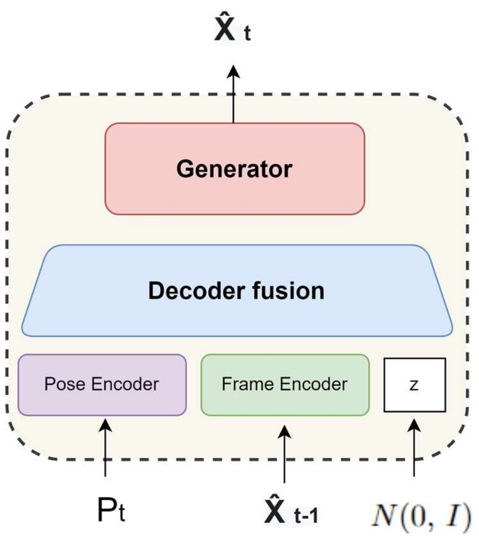


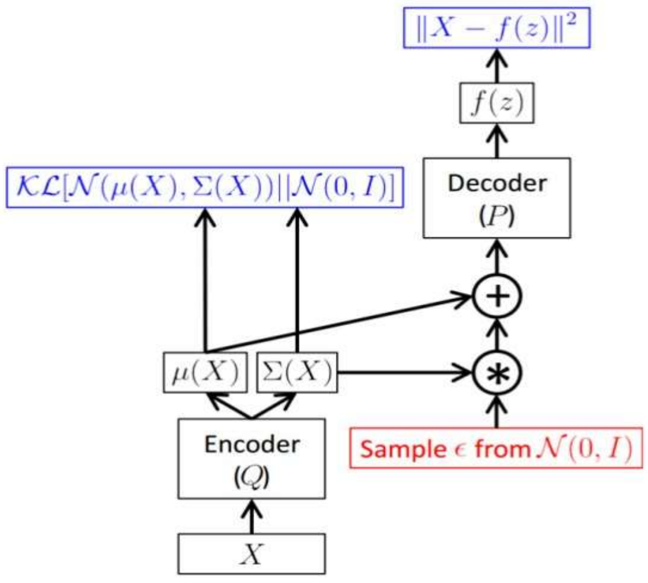
Figure 2. (b)

Fig. 2. (a) give you a coarse version of the training protocol. Posterior Predictor which takes the current frame and current label as input to generate the distribution. The generator which takes the current label, last generated frame, noise sampled by the distribution predicted by Posterior Predictor as inputs to generate the current frame. Figure 2. (b) is the video generation protocol in inference time that the noise will directly sample from the prior distribution.

III. Implementation Details

1. VAE

In the DL lecture, VAE has been explained thoroughly. While training VAE N2N, we need to adopt a reparameterization trick. For the purpose of stable training, the output of the reparameterization trick should be log variance instead of variance directly.


$\|X - f(z)\|^2$
$f(z)$
Decoder (P)
$\mathcal{KL}[\mathcal{N}(\mu(X), \Sigma(X))||\mathcal{N}(0, I)]$
$\mu(X)$
$\Sigma(X)$
Encoder (Q)
Sample ϵ from $\mathcal{N}(0, I)$
X

$$\underbrace{E_{Z\sim q(Z|X;\theta')}\log p(X|Z;\theta)}_{\text{Re-parameterization for end-to-end training}} \quad -\mathrm{KL}(q(Z|X;\theta')||p(Z))$$

Figure 3. The illustration of the reparameterization trick.

2. Training

There will be k (default=16) frames as a training video sequence {x1, x2, …… , xk}, and 16 label frames as conditional signals {P1, P2, …… , Pk}. First frame is the past frame which is provided to predict the consecutive k-1 future frames. Example is provided in Figure 4.

Figure 4. (a), and (b) give you some examples of how to generate the future frames. In (a), the task is to generate frame X2. However, we have no last frame to be taken as Decoder fusion input in scenario (a). Hence, X1 will be provided in the dataset to be the first input of the generative system. In (b), the task is to generate x3. Decoder fusion takes the frame generated by the last step (red dash box in fig. 4(a)) as input to make the prediction.

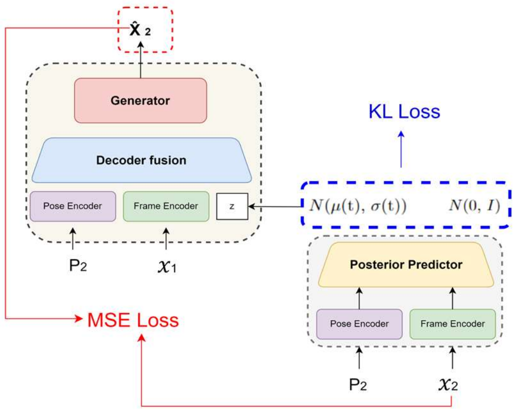


Figure 4. (a)

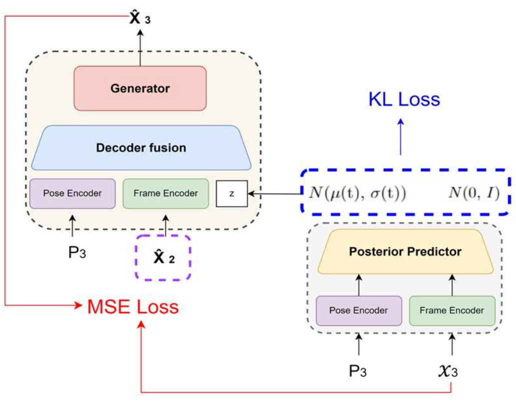


Figure 4. (b)

3. Inference

{ X1, P2, Z } will first be taken as input to generate the second frame. After the second frame is generated, it will be taken as an input to generate the next consecutive frame. Further detail can be found in Figure 4.

There would be 630 consecutive frames in the validation dataset. By taking one past frame, your model should predict the remaining 629 frames. After the prediction, it is suggested that you convert these 629 frames into a gif file, and see the quality of your prediction.

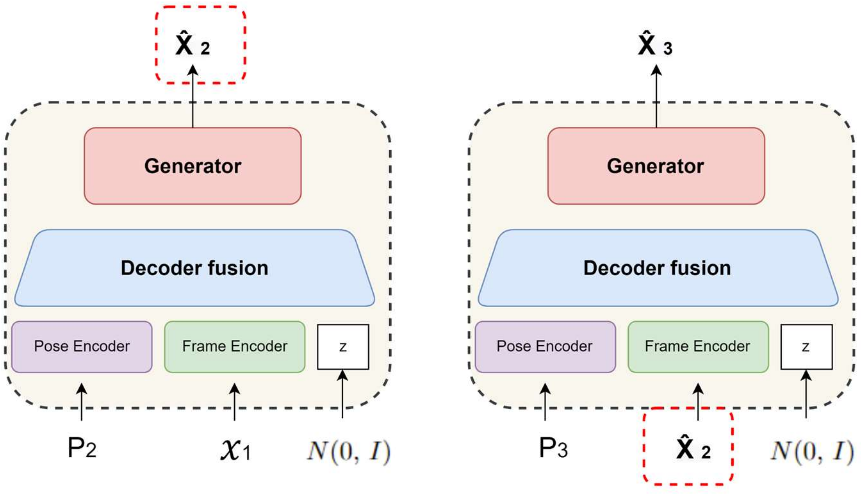


Figure 4. Example of how inference works.

4. KL annealing strategy

A variable weight is added to the KL term in the loss function. The following experiment results should be conducted and make the comparison in your report. It is suggested that you plot the loss curve in training while applying different strategies. Further details can be found in [3].

- Cyclical
- Monotonic
- w/o KL annealing

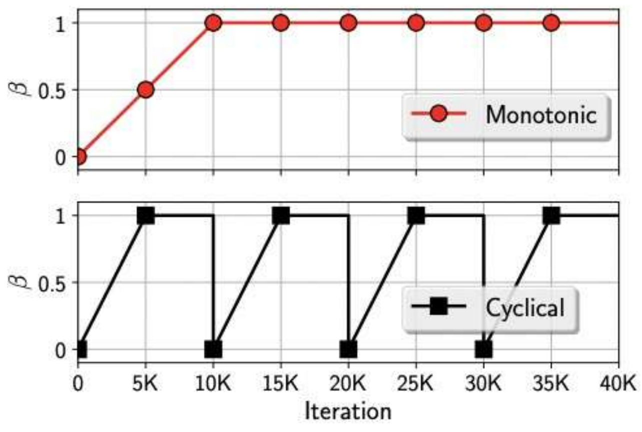


Figure 5. weight in different KL annealing strategies.

IV. Dataset

1. Training dataset
   - train_img: 23410 png files
   - train_label: 23410 png files
2. Validation dataset
   - val_img: 630 png files
   - val_label: 630 png files
3. Testing dataset
   - 5 video sequences are given. Each video sequence contains one first frame and 630 label frames.

V. Model Configurations

1. Input images and labels will be resized to (32, 64) due to memory limitation.
2. All modules e.g.
   - frame encoder
   - pose encoder
   - posterior generator
   - decoder fusion
   - generator

   are provided in directory named modules. Please DO NOT change the structure of modules.

3. Recommended command
   - Training command
     ```
     python Trainer.py --DR {YOUR_DATASET_PATH}
         --save_root {PATH_TO_SAVE_YOUR_CHECKPOINT}
         --fast_train
     ```
     (`--fast_train`: uses fewer dataset and large learning rate to speed up your training)
   - Testing command
     ```
     python Tester.py --DR {YOUR_DATASET_PATH}
         --save_root {PATH_TO_SAVE_YOUR_CHECKPOINT}
         --ckpt_path {PATH_TO_YOUR_CHECKPOINT}
     ```

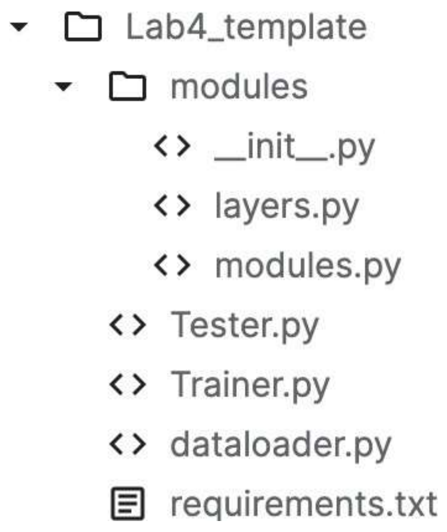
VI. Scoring Criteria

1. Report (60%)
   A. Introduction (5%)
   B. Implementation details (30%)
      You need to explain your codes briefly.
      i. How do you write your training/testing protocol (15%)
      ii. How do you implement reparameterization tricks (5%)
      iii. How do you set your teacher forcing strategy (5%)
      iv. How do you set your kl annealing ratio (5%)
   C. Analysis & Discussion (25%)
      i. Plot Teacher forcing ratio (5%)
         - Analysis & Compare with the loss curve
      ii. Plot the loss curve while training with different settings. Analyze the difference between them (15%)
         - With KL annealing (Monotonic)
         - With KL annealing (Cyclical)
         - Without KL annealing
      iii. Plot the PSNR-per-frame diagram in the validation dataset (5%)
      iv. Other training strategy analysis (Bonus 10%)
2. Kaggle competition (40%)
   - Pass baseline (Public) (20%)
   - Top 40 (Private) (10%)
   - Top 10 (Private) (10%)

VII. Files provided

Lab4 template is provided on Kaggle.

The structure would look like the picture below:



```
Lab4_template
    modules
        __init__.py
        layers.py
        modules.py
    Tester.py
    Trainer.py
    dataloader.py
    requirements.txt
```

VIII. Hints

Friendly notification

1. Do your work early
2. Read the reference papers
3. Dataloader is provided

IX. Upload file format

1. Lab 4- Conditional VAE (code)

   File name: LAB4_{studentID}_{YOUR_NAME}.zip

   example: LAB4_313553009_林于翔.zip

   Note: -5 for wrong file name

   Please DO NOT upload:

   - Dataset
   - PNG/GIF files you generate
   - Model weights, checkpoints

   Note: -5 if you uploaded them

2. Lab 4- Conditional VAE (report)

   File name: LAB4_{studentID}_{YOUR_NAME}.pdf

   example: LAB4_313553009_林于翔.pdf

   Note: -5 for wrong file name

X. References

[1] C. Chan, et al., “Everybody Dance Now,” ICCV, 2019
[2] E. Denton, et al., “Stochastic Video Generation with a Learned Prior,” ICML, 2018
[3] H. Fu, et al., “Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing,” NAACL 201