NYCU 2024 Spring DL Lab6
Generative Models

TA 楊賀弼

Aug 13, 2024

Outline

- Rule
- Lab description
- Model design guide
- Scoring criteria

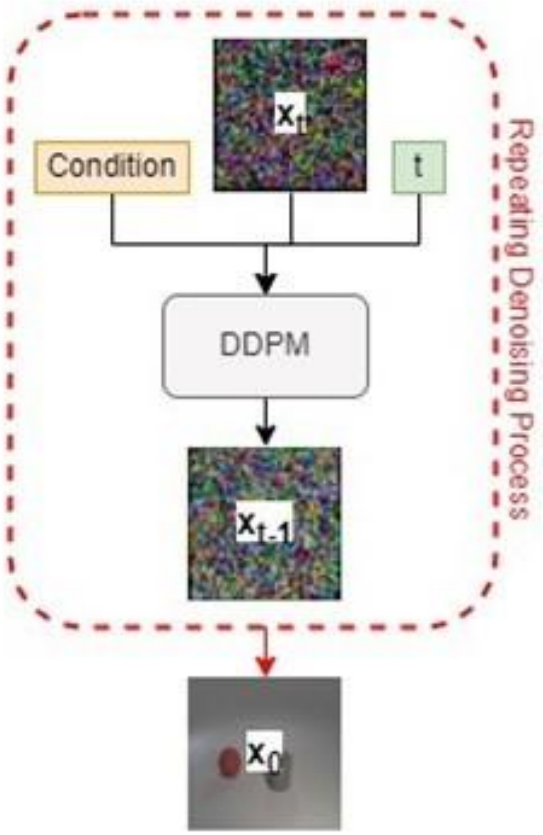
Rule

- Report submission deadline: Aug 23, 2024, 11:59 p.m.
- No need to demo this lab
- Zip all files in one file
  - Report (.pdf)
  - Source code
- Name it "DL_LAB6_YourStudentID_YourName.zip”
  - Example: “DL_LAB6_313553051_楊賀弼.zip”
- -5% to your score if you do not follow the format
- Upload the model weight

Lab description

Lab objective

- You need to implement a DDPM to generate synthetic images according to multi-label conditions

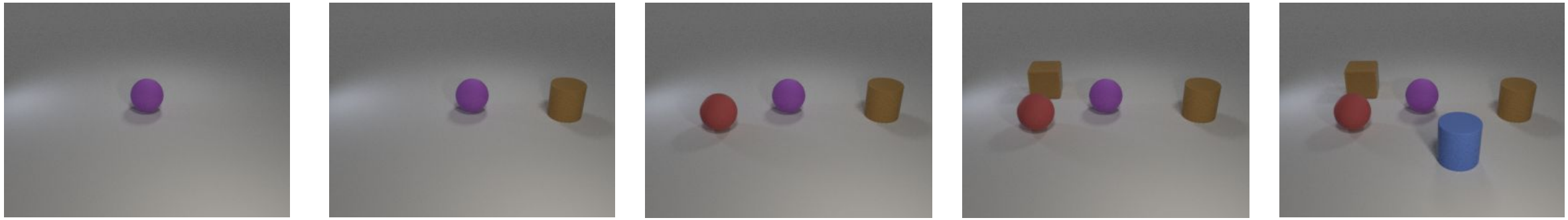


Requirements

- Design and train your models
  - Implement a conditional DDPM
  - You can use any architecture you like
- Synthesis image and evaluate the results
  - Show the synthetic images in grids (for test.json, new_test.json)
  - Show the denoising process in a grid for sampling from your DDPM (with the label set ["red sphere", "cyan cylinder", "cyan cube"])
  - Evaluate your model with the classification accuracies from the provided evaluator (for test.json, new_test.json)

Dataset

- Provided files
  - readme.txt, train.json, test.json, new_test.json, object.json, iclevr.zip
- object.json
  - Dictionary of objects
  - 24 classes
- Example of labels:
  - [“cyan cylinder”, “red cube”], [“green sphere”], …
  - The same object will not appear twice in an image



Pretrained evaluator

- Provided files: evaluator.py, checkpoint.pth
  - DO NOT modify any of them
- Use **eval**(images, labels) to compute accuracy of your synthetic images
  - Labels should be one-hot vector. E.g. [[1,1,0,0,…],[0,1,0,0,…],…]
  - Images should be all generated images. E.g. (batch size, 3, 64, 64)
  - Images should be normalized with transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
- You can involve this evaluator in training or sampling for better result
  - E.g. classifier guidance for DDPM
  - Inherit the class if you need extra functionalities
  - Again, DO NOT modify the weight and class script

Implement a conditional DDPM

Design of DDPM

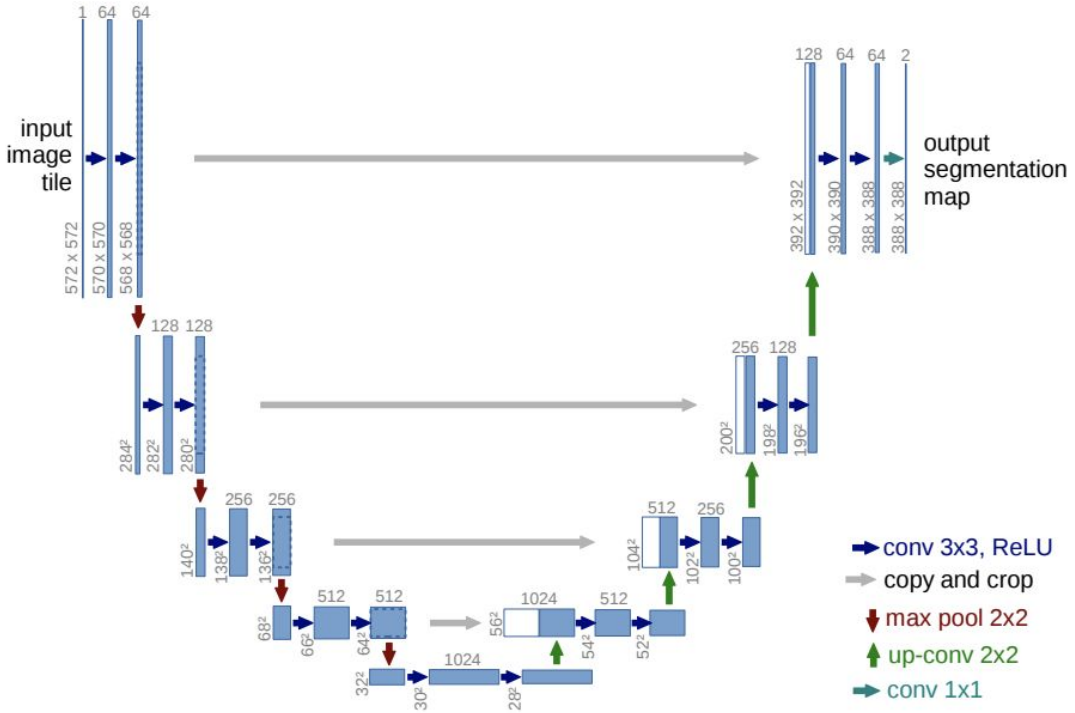
- Denoising diffusion model
- Latent diffusion model
  - encoder/decoder design

- Condition embedding
  - label embedding
  - Time embedding

- Noise schedule
  - linear
  - cosine
  - other

Design of UNet

- Number of blocks
- Number of layers, channels in each blocks
- Down/upsampling blocks design
  - Architecture for the blocks. e.g. Resnet block
  - Self/cross attention
  - Where to add condition
  - other



Choice of sampling method

- DDPM, DDIM, etc.
- With classifier guidance (Diffusion models beat gans on image synthesis)

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
  $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
  $x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$
end for
return x_0

You can include the pretrained evaluator (classifier) here

Algorithm 2 Classifier guided DDIM sampling, given a diffusion model $\epsilon_\theta(x_t)$, classifier $p_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
  $\hat{\epsilon} \leftarrow \epsilon_\theta(x_t) - \sqrt{1-\bar{\alpha}_t}\, \nabla_{x_t} \log p_\phi(y|x_t)$
  $x_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}} \left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\hat{\epsilon}}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1-\bar{\alpha}_{t-1}}\hat{\epsilon}$
end for
return x_0

You can include the pretrained evaluator (classifier) here

Simple suggestions (in the other PDF)

3.2 DDPM

- Denoising Diffusion Probabilistic Models
- Hugging Face Diffusion Models Course

The Hugging Face Diffuser library offers comprehensive functionality to help you implement various diffusion model architectures, scheduling methods, sampling types, reparameterizations, etc. You can refer to the tutorial linked above to design your diffusion model for this lab.

Scoring criteria

Scoring criteria

- Report (60%)
  - Introduction (5%)
  - Implementation details (25%)
    - Describe how you implement your model, including your choice of DDPM, noise schedule
      (There is no demo in this lab, so please write in detail.)
  - Results and discussion (30%)
    - Show your synthetic image grids (16%: 8% * 2 testing data) and a denoising process image (4%)
    - Discussion of your extra implementations or experiments (10%)

Scoring criteria

- Result (40%) (based on results shown in your report)
  - Classification accuracy on test.json and new_test.json.
  - Show your accuracy screenshots
  - The command for inference process for both testing data
    (Please make sure TA can understand how to run your inference code and have your synthetic images)
  - 20% * 2 testing data

Accuracy	Grade
score ≥ 0.8	100%
$0.8 >$ score ≥ 0.7	90%
$0.7 >$ score ≥ 0.6	80%
$0.6 >$ score ≥ 0.5	70%
$0.5 >$ score ≥ 0.4	60%
score < 0.4	0%

Model Weight Upload

- Weight upload link : Link

- Notice : Please name your model weight as
  “DL_lab6_YourStudentID_Name.pth”
  e.g.”DL_lab6_ 313553051_楊賀弼.pth”
- The deadline for uploading model weight is the same as the lab.

Output examples

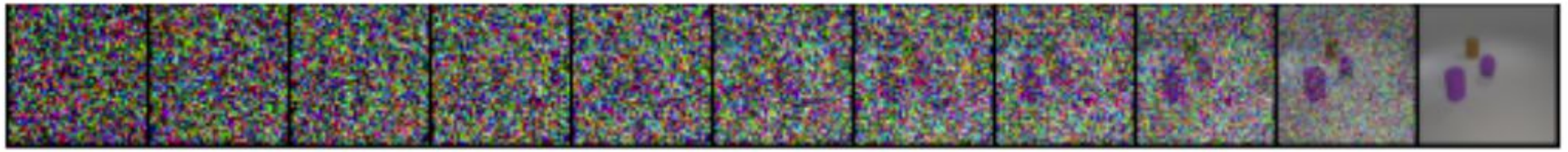


Fig. 2: Example of the denoising process image.

Fig. 3: The synthetic image grid on new_test.json. (F1-score 0.821)