# CPEN 455: Deep Learning

## Course Project

Qihang Zhang

University of British Columbia

Winter, Term 2, 2024

# Use lightweight model for debugging (locally)

```
python pcnn_train.py \
--batch_size 16 \
--sample_batch_size 16 \
--sampling_interval 25 \
--save_interval 25 \
--dataset cpen455 \
--nr_resnet 1 \
--nr_filters 10 \
--nr_logistic_mix 2 \
--lr_decay 0.999995 \
--max_epochs 100 (not necessary for debug)
```

```
git add .
git commit –m “”
git push
```

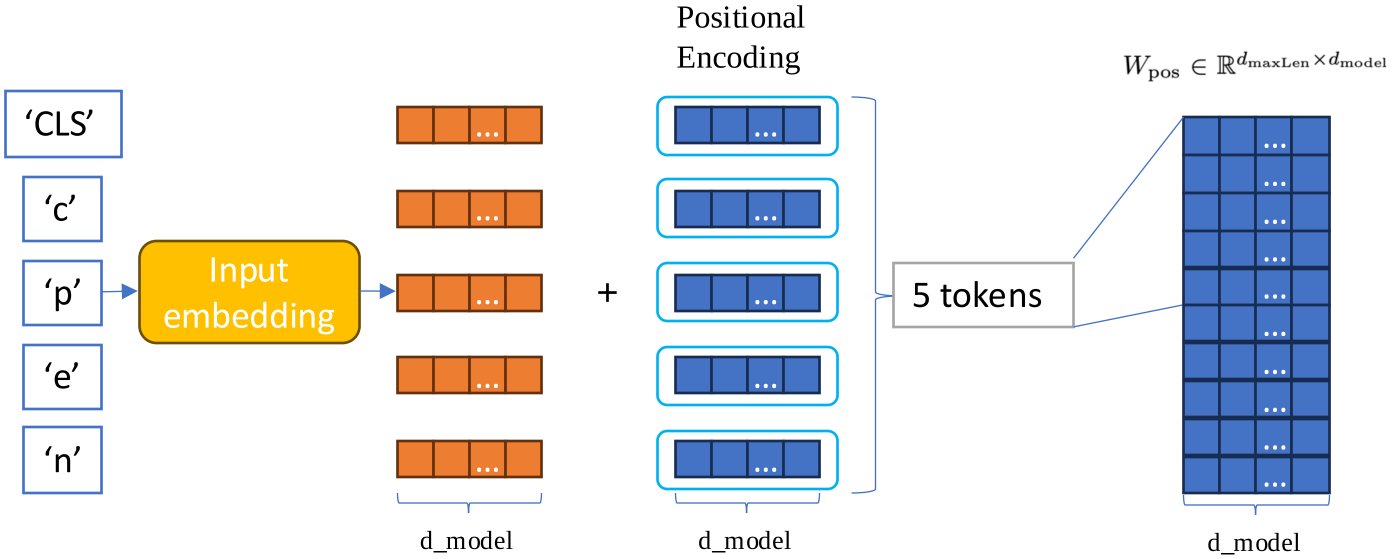
# Use bigger model for training (on cloud)

```
git pull
```

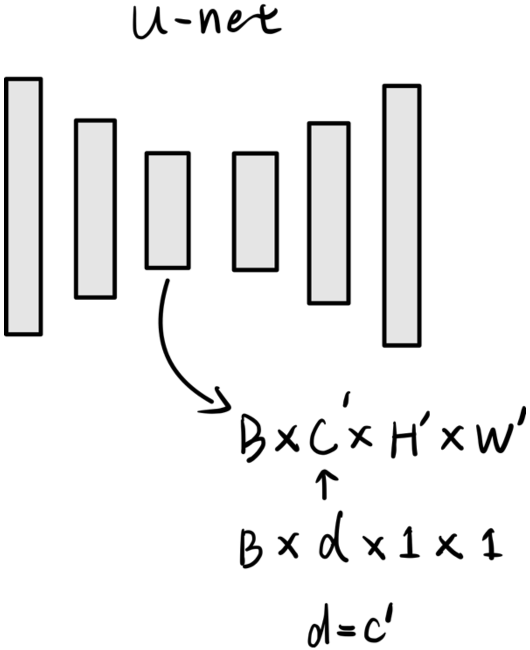
```
python pcnn_train.py \
--batch_size 16 \
--sample_batch_size 16 \
--sampling_interval 25 \
--save_interval 25 \
--dataset cpen455 \
--nr_resnet 5 \
--nr_filters 160 \
--nr_logistic_mix 10 \
--lr_decay 0.999995 \
--max_epochs 100 (until bpd converage)
```

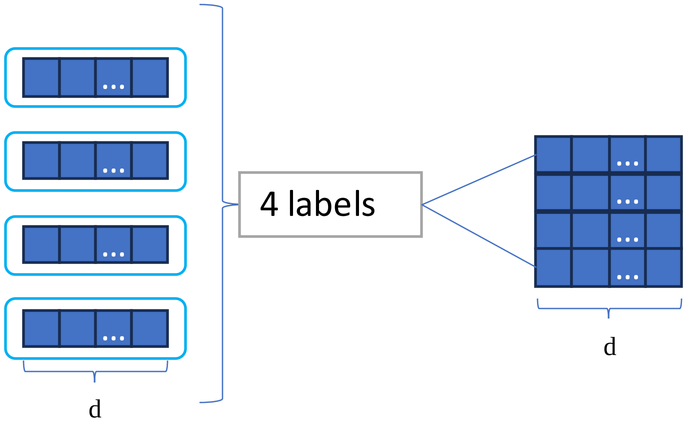
# Positional Encoding

**1.2 [5pts]** Implement the (absolute) learnable positional encoding module. This module has a learnable weight matrix $W_{\text{pos}} \in \mathbb{R}^{d_{\text{maxLen}} \times d_{\text{model}}}$. The $i$-th row $(1 \leq i \leq d_{\text{maxLen}})$ of this matrix is a learnable vector corresponding to the $i$-th position in a sequence. This module applies positional encoding to a sequence by element-wise adding rows of this matrix to their corresponding position in the input.



# Label info Fusing


U-net
$B \times C' \times H' \times W'$
$B \times d \times 1 \times 1$
$d = c'$



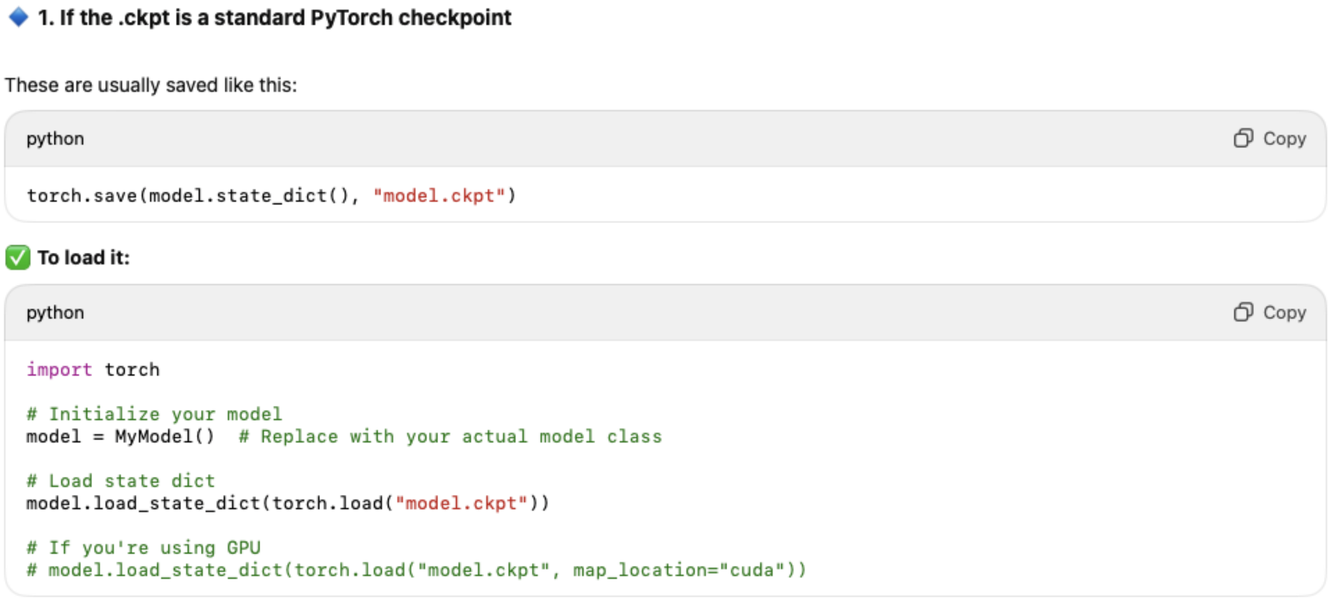
# Evaluate Classicifation when training

```
for epoch in tqdm(range(args.max_epochs)):
    train_or_test(..., mode = ‘train‘)
    train_or_test(..., mode = 'val’)
    eval_classification()
```

# Questions asked

How many epochs should I train? It depends on bpd, FID, and acc.

Colab is not stable? The code we provide can save ckpt every few epochs, just load the saved ckpt next time you train your model.



🔷 **1. If the .ckpt is a standard PyTorch checkpoint**

These are usually saved like this:

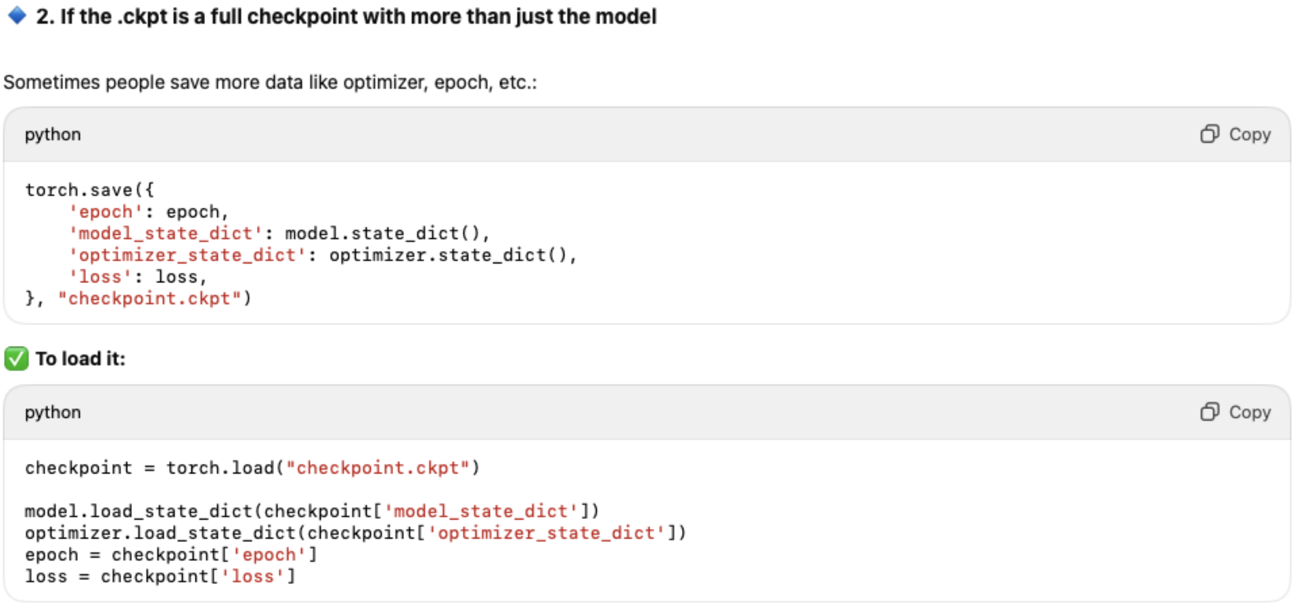
```python
torch.save(model.state_dict(), "model.ckpt")
```

✅ **To load it:**

```python
import torch

# Initialize your model
model = MyModel()  # Replace with your actual model class

# Load state dict
model.load_state_dict(torch.load("model.ckpt"))

# If you're using GPU
# model.load_state_dict(torch.load("model.ckpt", map_location="cuda"))
```

# Questions asked

Colab is not stable? The code we provide can save ckpt every few epochs, just load the saved ckpt next time you train your model.



### 🔷 2. If the .ckpt is a full checkpoint with more than just the model

Sometimes people save more data like optimizer, epoch, etc.:

```python
torch.save({
    'epoch': epoch,
    'model_state_dict': model.state_dict(),
    'optimizer_state_dict': optimizer.state_dict(),
    'loss': loss,
}, "checkpoint.ckpt")
```

**✅ To load it:**

```python
checkpoint = torch.load("checkpoint.ckpt")

model.load_state_dict(checkpoint['model_state_dict'])
optimizer.load_state_dict(checkpoint['optimizer_state_dict'])
epoch = checkpoint['epoch']
loss = checkpoint['loss']
```

# Questions asked

Colab is not stable? Kaggle gives 30 hours/week of using GPU's for free, and in the background, so you dont have to be active at all on screen when running it. you can run it and go to sleep. (https://piazza.com/class/m5lcfy9klal5kc/post/155)

# Grading

Make sure generation_evaluation.py and classification_evaluation.py can run locally.