CS597 REPORT 4

Yash Pradeep Gupte
MS Computer Science
A20472798

1] CODE – StyleAttnGAN

Repository link - https://github.com/sidward14/Style-AttnGAN

(A) Datasets

- Bird – CUB-200-2011 Number of categories: 200, Number of images: 11,788
- Coco – MS COCO - 80 object categories, 5 captions per image

(B) Training

(1) Pre training DAMSM model/ encoder
(2) Train Style- AttnGAN models
(3) Train original AttnGAN models

(C) Sampling

Transformer encoder : 'gpt2'

Configuration file : eval_bird_style.yml

Image Transform:[Resize(size=304, interpolation=bilinear, max_size=None, antialias=None), RandomCrop(size=(256, 256), padding=None), RandomHorizontalFlip(p=0.5)]

TextDataset:

Bbox , captions, classids, filenames, imsize (64,128,256), IndextoWords, N_words(5450), WordtoIndex.

Dataloader :

Batchsize(8), Textdataset

Sampling function which generates images from pretrained models:

Split_directory = ‘valid”
Generator model = TRAIN.NET_G
Text_encoder, net_G =

MODEL – Architecture

```
G_NET_STYLED(
  (ca_net): CA_NET(
    (fc): Linear(in_features=256, out_features=400, bias=True)
```

```
  (relu): GLU()
)
(map_net): StyleConditionedMappingNetwork(
  (fc_mapping_model): Sequential(
    (pixelnorm): NormalizeLayer(
      (norm): PixelNorm2d()
    )
    (fc_0): LinearEx(
      (linear): Linear(in_features=200, out_features=100, bias=True)
    )
    (nl_0): LeakyReLU(negative_slope=0.2)
    (fc_1): LinearEx(
      (linear): Linear(in_features=100, out_features=100, bias=True)
    )
    (nl_1): LeakyReLU(negative_slope=0.2)
    (fc_2): LinearEx(
      (linear): Linear(in_features=100, out_features=100, bias=True)
    )
    (nl_2): LeakyReLU(negative_slope=0.2)
    (fc_3): LinearEx(
      (linear): Linear(in_features=100, out_features=100, bias=True)
    )
    (nl_3): LeakyReLU(negative_slope=0.2)
    (fc_4): LinearEx(
      (linear): Linear(in_features=100, out_features=100, bias=True)
    )
    (nl_4): LeakyReLU(negative_slope=0.2)
    (fc_5): LinearEx(
      (linear): Linear(in_features=100, out_features=100, bias=True)
    )
    (nl_5): LeakyReLU(negative_slope=0.2)
    (fc_6): LinearEx(
      (linear): Linear(in_features=100, out_features=100, bias=True)
    )
    (nl_6): LeakyReLU(negative_slope=0.2)
    (fc_7): LinearEx(
      (linear): Linear(in_features=100, out_features=100, bias=True)
    )
    (nl_7): LeakyReLU(negative_slope=0.2)
  )
)
(h_net1): INIT_STAGE_G_STYLED(
  (gen_layers): ModuleList(
    (0): ModuleList(
```

```
      (0): None
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(128, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
      )
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=256, bias=True)
      )
    )
    (1): ModuleList(
      (0): Conv2dEx(
        (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      )
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
      )
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=256, bias=True)
      )
    )
    (2): ModuleList(
      (0): Sequential(
        (0): Upsample(scale_factor=2.0, mode=nearest)
        (1): Conv2dEx(
          (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
        (2): Lambda()
      )
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
```

```
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
       )
       (3): LinearEx(
        (linear): Linear(in_features=100, out_features=256, bias=True)
       )
      )
      (3): ModuleList(
       (0): Sequential(
        (0): Conv2dEx(
         (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
       )
       (1): StyleAddNoise()
       (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
       )
       (3): LinearEx(
        (linear): Linear(in_features=100, out_features=256, bias=True)
       )
      )
      (4): ModuleList(
       (0): Sequential(
        (0): Upsample(scale_factor=2.0, mode=nearest)
        (1): Conv2dEx(
         (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
        (2): Lambda()
       )
       (1): StyleAddNoise()
       (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
       )
```

```
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=256, bias=True)
      )
    )
    (5): ModuleList(
      (0): Sequential(
        (0): Conv2dEx(
          (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
      )
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
      )
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=256, bias=True)
      )
    )
    (6): ModuleList(
      (0): Sequential(
        (0): Upsample(scale_factor=2.0, mode=nearest)
        (1): Conv2dEx(
          (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
        (2): Lambda()
      )
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
      )
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=256, bias=True)
      )
    )
```

```
    (7): ModuleList(
      (0): Sequential(
        (0): Conv2dEx(
          (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
      )
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
      )
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=256, bias=True)
      )
    )
    (8): ModuleList(
      (0): Sequential(
        (0): Upsample(scale_factor=2.0, mode=nearest)
        (1): Conv2dEx(
          (conv2d): Conv2d(128, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
        (2): Lambda()
      )
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
      )
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=128, bias=True)
      )
    )
    (9): ModuleList(
      (0): Sequential(
        (0): Conv2dEx(
          (conv2d): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
```

```
        )
       )
       (1): StyleAddNoise()
       (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
       )
       (3): LinearEx(
        (linear): Linear(in_features=100, out_features=128, bias=True)
       )
     )
   )
   (upsampler): Upsample(scale_factor=2.0, mode=nearest)
   (nl): LeakyReLU(negative_slope=0.2)
   (norm): NormalizeLayer(
     (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
   )
  )
  (img_net1): GET_IMAGE_G_STYLED(
   (torgb): Sequential(
     (0): Conv2dEx(
       (conv2d): Conv2d(64, 3, kernel_size=(1, 1), stride=(1, 1))
     )
     (1): Tanh()
   )
  )
  (h_net2): NEXT_STAGE_G_STYLED(
   (att): GlobalAttentionGeneral(
     (conv_context): Conv2d(256, 64, kernel_size=(1, 1), stride=(1, 1), bias=False)
     (sm): Softmax(dim=1)
   )
   (residual): ResBlock(
     (block): Sequential(
       (0): Conv2dEx(
         (conv2d): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
       )
       (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
       (2): GLU()
```

```
      (3): Conv2dEx(
        (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      )
      (4): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
  )
  (gen_layers): ModuleList(
    (0): ModuleList(
      (0): Sequential(
        (0): Upsample(scale_factor=2.0, mode=nearest)
        (1): Conv2dEx(
          (conv2d): Conv2d(128, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
        (2): Lambda()
      )
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
      )
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=128, bias=True)
      )
    )
    (1): ModuleList(
      (0): Sequential(
        (0): Conv2dEx(
          (conv2d): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
      )
      (1): StyleAddNoise()
      (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
          (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
      )
```

```
      (3): LinearEx(
        (linear): Linear(in_features=100, out_features=128, bias=True)
      )
    )
  )
  (upsampler): Upsample(scale_factor=2.0, mode=nearest)
  (nl): LeakyReLU(negative_slope=0.2)
  (norm): NormalizeLayer(
    (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
  )
 )
 (img_net2): GET_IMAGE_G_STYLED(
  (torgb): Sequential(
    (0): Conv2dEx(
      (conv2d): Conv2d(64, 3, kernel_size=(1, 1), stride=(1, 1))
    )
    (1): Tanh()
  )
 )
 (h_net3): NEXT_STAGE_G_STYLED(
  (att): GlobalAttentionGeneral(
    (conv_context): Conv2d(256, 64, kernel_size=(1, 1), stride=(1, 1), bias=False)
    (sm): Softmax(dim=1)
  )
  (residual): ResBlock(
    (block): Sequential(
      (0): Conv2dEx(
        (conv2d): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      )
      (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (2): GLU()
      (3): Conv2dEx(
        (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      )
      (4): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
  )
  (gen_layers): ModuleList(
    (0): ModuleList(
      (0): Sequential(
        (0): Upsample(scale_factor=2.0, mode=nearest)
```

```
        (1): Conv2dEx(
          (conv2d): Conv2d(128, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
        (2): Lambda()
       )
       (1): StyleAddNoise()
       (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
         (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
       )
       (3): LinearEx(
        (linear): Linear(in_features=100, out_features=128, bias=True)
       )
      )
      (1): ModuleList(
       (0): Sequential(
        (0): Conv2dEx(
          (conv2d): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        )
       )
       (1): StyleAddNoise()
       (2): Sequential(
        (0): Conv2dBias()
        (1): LeakyReLU(negative_slope=0.2)
        (2): NormalizeLayer(
         (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
        )
       )
       (3): LinearEx(
        (linear): Linear(in_features=100, out_features=128, bias=True)
       )
      )
    )
    (upsampler): Upsample(scale_factor=2.0, mode=nearest)
    (nl): LeakyReLU(negative_slope=0.2)
    (norm): NormalizeLayer(
     (norm): InstanceNorm2d(None, eps=1e-08, momentum=0.1, affine=False,
track_running_stats=False)
    )
```

```
  )
  (img_net3): GET_IMAGE_G_STYLED(
    (torgb): Sequential(
      (0): Conv2dEx(
        (conv2d): Conv2d(64, 3, kernel_size=(1, 1), stride=(1, 1))
      )
      (1): Tanh()
    )
  )
)
```

Net_G.eval()

```
Text_encoder : RNN_ENCODER(
    (encoder): Embedding(5450, 300)
    (drop): Dropout(p=0.5, inplace=False)
     (rnn): LSTM(300, 128, batch_first=True, dropout=0.5,
bidirectional=True)
)
```

Batch_size = 8

Noise : tensor dimension = (8,100) -> (batch_size, nz)

Dataloader (for 1st batch)

Keys - ['036.Northern_Flicker/Northern_Flicker_0072_28678',
'004.Groove_billed_Ani/Groove_Billed_Ani_0031_1588',
'112.Great_Grey_Shrike/Great_Grey_Shrike_0086_106533',
'166.Golden_winged_Warbler/Golden_Winged_Warbler_0087_794810',
'098.Scott_Oriole/Scott_Oriole_0084_795860',
'180.Wilson_Warbler/Wilson_Warbler_0050_175573',
'023.Brandt_Cormorant/Brandt_Cormorant_0073_23259',
'029.American_Crow/American_Crow_0130_25163']

Imgs –

0 – (8,3,64,64)
1 – (8,3,128,128)
2 – (8,3,256,256)

Class_ids : [36 4 112 166 98 180 23 29]

Captions : Tensor(8,18)
Cap_lens = 8 captions in one batch – Max length of words in captions = 18

Word_embeddings : size(8,256,18)

Sentence embeddings : size(8,256)

#Generate Fake images
fake_imgs, _, _, _ = netG(noise, sent_emb, words_embs, mask)
Pass the noise, sentence embeddings, word embeddings and mask

(D) Losses ->

Discriminator loss – BCE Binary cross entropy

--- Losses.py file ---

```
if netD.UNCOND_DNET is not None:
        errD = ((real_errD + cond_real_errD) / 2. + (fake_errD + cond_fake_errD +
cond_wrong_errD) / 3.)
else:
        errD = cond_real_errD + (cond_fake_errD + cond_wrong_errD) / 2.
```

Generator Loss – BCE

--- Losses.py file LINE :187 ---

```
errG = nn.BCELoss()(logits, real_labels)
g_loss = errG + cond_errG
```

Rankingloss – word features and sentence code

```
w_loss0, w_loss1, _ = words_loss(region_features, words_embs,
            match_labels, cap_lens,
            class_ids, batch_size)
w_loss = (w_loss0 + w_loss1) * cfg.TRAIN.SMOOTH.LAMBDA
s_loss0, s_loss1 = sent_loss(cnn_code, sent_emb,match_labels, class_ids,
batch_size)
s_loss = (s_loss0 + s_loss1) * cfg.TRAIN.SMOOTH.LAMBDA
```

(E) Training – config file : bird_attn2_style.yml

Command : python main.py --cfg cfg/bird_attn2_style.yml --gpu 0

<class 'model.G_NET_STYLED'>
<class 'model.D_NET_STYLED64'>
<class 'model.D_NET_STYLED128'>
<class 'model.D_NET_STYLED256'>
# of netsD 3
Optimizers – Adam optimizer with LR = 0.002