

Reconstruct

March 4, 2019

```
In [4]: import torch

from torch.autograd import Variable
from torchvision.datasets import MNIST
from torchvision.transforms import ToTensor
from data_loader import ImageFolder
from torchvision import transforms

from net import CapsNetWithReconstruction, CapsNet, ReconstructionNet

%matplotlib inline
import matplotlib.pyplot as plt

# initialize network classes
capsnet = CapsNet(3, 33)
reconstructionnet = ReconstructionNet(16, 33)
model = CapsNetWithReconstruction(capsnet, reconstructionnet)

# Load trained model
MODEL_PATH = '060_model_dict_3routing_reconstructionTrue.pth'
model.load_state_dict(torch.load(MODEL_PATH))

transform = transforms.Compose([transforms.Grayscale(), transforms.ToTensor()])

dataset = ImageFolder("/media/jiashu/Data/Peal_Test/Re/", transform)

# (1x28x28 tensor input)
def get_digit_caps(model, image):
    input_ = Variable(image.unsqueeze(0), volatile=True)
    digit_caps, probs = model.capsnet(input_)
    return digit_caps

# takes digit_caps output and target label
def get_reconstruction(model, digit_caps, label):
    target = Variable(torch.LongTensor([label]), volatile=True)
    reconstruction = model.reconstruction_net(digit_caps, target)
    return reconstruction.data.cpu().numpy()[0].reshape(224, 224)
```

```

# create reconstructions with perturbed digit capsule
def dimension_perturbation_reconstructions(model, digit_caps, label, dimension, dim_val):
    reconstructions = []
    label = label.long()
    for dim_value in dim_values:
        digit_caps_perturbed = digit_caps.clone()
        # print("Orginal :", digit_caps_perturbed)
        # print("Change Value :", dim_value)
        # print("Before :", digit_caps_perturbed[0, label, dimension])
        digit_caps_perturbed[0, label, dimension] = dim_value
        # print("After :", digit_caps_perturbed[0, label, dimension])
        reconstruction = get_reconstruction(model, digit_caps_perturbed, label)
        reconstructions.append(reconstruction)
    return reconstructions

print("OVER")

```

OVER

```

In [5]: # Get reconstructions
images = []
reconstructions = []
for i in range(10):
    image_tensor, label = dataset[i]
    # print(type(image_tensor))
    digit_caps = get_digit_caps(model, image_tensor)
    reconstruction = get_reconstruction(model, digit_caps, label)
    images.append(image_tensor.numpy()[0])
    reconstructions.append(reconstruction)
print("OVER")

```

/home/jiashu/.local/lib/python3.6/site-packages/ipykernel_launcher.py:29: UserWarning: volatile...
/home/jiashu/.local/lib/python3.6/site-packages/ipykernel_launcher.py:35: UserWarning: volatile...

OVER

```

In [6]: # Plot reconstructions

fig, axs = plt.subplots(2, 10, figsize=(16, 4))
axs[0, 0].set_ylabel('Org image', size='large')
axs[1, 0].set_ylabel('Reconstruction', size='large')
for i in range(10):
    axs[0, i].imshow(images[i], cmap='gray')
    axs[1, i].imshow(reconstructions[i], cmap='gray')
    axs[0, i].set_yticks([])

```

```

    axs[0, i].set_xticks([])
    axs[1, i].set_yticks([])
    axs[1, i].set_xticks([])

digit, label = dataset[9]
perturbed_reconstructions = []
perturbation_values = [0.15*i for i in range(-7, 8)]
digit_caps = get_digit_caps(model, digit)

# print("Original Vector ", digit_caps)

for dimension in range(16):
    perturbed_reconstructions.append(
        dimension_perturbation_reconstructions(model, digit_caps, label,
                                                dimension, perturbation_values)
    )

fig, axs = plt.subplots(16, 15, figsize=(11*1.5, 16*1.5))
for i in range(16):
    axs[i, 0].set_ylabel('dim {}'.format(i), size='large')
    for j in range(15):
        axs[i, j].imshow(perturbed_reconstructions[i][j], cmap='gray')
        axs[i, j].set_yticks([])
        axs[i, j].set_xticks([])
        if(j == 7):
            axs[i, 7].set_xlabel('Original', size='small')
        if(j < 7):
            modify_value = -0.15*(7-j)
            modify_value = round(modify_value, 2)
            axs[i, j].set_xlabel(modify_value, size='small')
        if(j > 7):
            modify_value = 0.15*(j-7)
            modify_value = round(modify_value, 2)
            axs[i, j].set_xlabel(modify_value, size='small')

print("OVER")

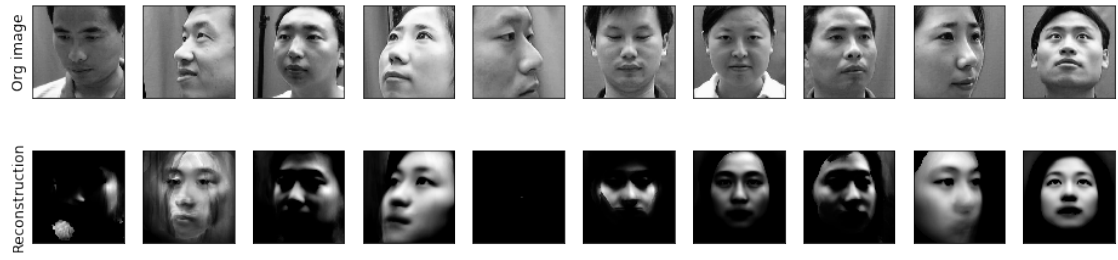
```

```

/home/jiashu/.local/lib/python3.6/site-packages/ipykernel_launcher.py:29: UserWarning: volatil
/home/jiashu/.local/lib/python3.6/site-packages/ipykernel_launcher.py:35: UserWarning: volatil

```

OVER





In []:

```
In [ ]:
```