Deep Learning Course

Assignment Four

Assignment Goals:

- Implementing Fully Connected AutoEncoders
- Implement naive generative model
- Understand VAE and GAN, then implement a classical generative model: VAE-GAN.

Please be aware that this assignment must be completed using PyTorch.

DataSet

In this Assignment, you will use the Fashion-MNIST dataset. The dataset is not given in the assignment package, please download/load by yourself. *Hint*: You can use

```
(x_train, _), (x_test, _)  = keras.datasets.fashion_mnist.load_data()
```

to load the dataset.

```
import keras
(x_train, _), (x_test, _)  = keras.datasets.fashion_mnist.load_data()
print(type(x_train))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset
29515/29515 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset
26421880/26421880 [==============================] - 2s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset
5148/5148 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset
4422102/4422102 [==============================] - 1s 0us/step
<class 'numpy.ndarray'>
```

Requirements

1. **(20 points) Implement a Fully Connected AutoEncoder**
    - Your AutoEncoder should have a bottleneck with two neurons and use Mean Squared Error (MSE) as the objective function. Design the model structure by yourself. Notice that

in an AutoEncoder, the layer with the least number of neurons is referred to as a bottleneck.

- Train your model on Fashion-MNIST. Plot the train and test loss.
- Randomly select 10 images from the test set, encode the selected 10 images, visualize the original images and the decoded images.

2. **(30 points) Naive generative model**

   This question is about using an AutoEncoder to generate similar but not identical Fashion-MNIST items. We use a naive approach: Try to see if a trained decoder can map randomly generated inputs (random numbers) to a recognizable Fashion-MNIST item.

   1. Start with your Fully Connected AutoEncoder from part 1. Try to generate new images by inputting some random numbers to the decoder (i.e. the bottleneck layer). Visualize 10 generated images. (10 points)
   2. Now restrict each neuron of the bottleneck layer to have a distribution with mean zeroes and variance one. Retrain the Fully Connected AutoEncoder with the normalized bottleneck. Now randomly generate inputs to the bottleneck layer that are drawn from the multi-variate standard normal distribution, and use the randomly generated inputs to generate new images. Visualize 10 generated images. (15 points)
   3. Are the output images different between A) and B)? If so, why do you think this difference occurs? (5 points)

3. **(50 points + 5 points BONUS 1(optional) + 5 points BONUS 2 (optional)) Advanced generative model**

   In this part, you are asking to implement a VAE-GAN model. A VAE-GAN is a Generative Adversarial Network whose generator is an Variational Autoencoder.Here is the paper which proposed the VAE-GAN: **[PAPER]**. You may need to read this paper before implementing this model.

   1. Implement a Variational Autoencoder based on your Fully Connected AutoEncoder from part 1. Use your VAE to randomly generate 10 images. Does the VAE produce a different quality of output image? (30 points)
   2. Implement a VAE-GAN based on your implemented VAE. Train the VAE-GAN. (20 points)
      - Then use your VAE-GAN to randomly generate 10 images from $p(z)$.
      - Randomly select 10 images from the test dataset and reconstruct them using your model, then visualize the original and reconstructed images.
      - **Loss-function of your VAE-GAN model:**

- The basic assignment will choose the L_{prior} and L_{Gan} loss described in the paper (Eq. 5). So your loss function will be

  $L = L_{prior} + L_{gan}$

  instead of $L = L_{prior} + L^{Dis_l} + L_{gan}$ (Eq. 8).

- **BONUS 1 (5 points)**: For those of you who want to try incorporating the L^{dis_l} term, this would be a bonus question. The basic idea here is to use a discriminator and use one of the hidden layers in the discriminator as a representation of the input. They apply this with a CNN but you can use a hidden discriminator layer with different discriminator architectures. You would be free to use your own creativity in how to select a discriminator layer.
- **BONUS 2 (5 points)**: The VAE-GAN paper uses a convolutional neural network instead of a fully connected architectre. For a bonus, you can replace the fully connected auto-encoder from the previous part with a convolutional neural network. You can use the architecture described in the paper, or adapt the convolutional auto-encoder architecture described in the book to work with the VAE-GAN system.
- *Hint:* (1) For the generation and reconstruction tasks, refer to Section 4.1 in the paper.(2) the authors have posted their code in Github. You should write your own code completely from scratch, but you can look at their code as a resource for clarification.*

Submission Notes:

Please use Jupyter Notebook. The notebook should include the final code, results, and answers. You should submit your Notebook in .pdf and .ipynb format. (penalty 10 points).

Notice that your AutoEncoders should have only one bottleneck.

This assignment must be completed using PyTorch

Instructions:

The university policy on academic dishonesty and plagiarism (cheating) will be taken very seriously in this course. Everything submitted should be your writing or coding. You must not let other students copy your work. Spelling and grammar count.

Your implementation

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.