Deep Learning 2020 Assignment 5

Assignment 5a: Implementing deep generative models in PyTorch

Due to: Dec 20, 2020

Goal: Implement a VAE and a GAN in PyTorch and analyze them.

Submission: The assignment consists of two parts: implementation and an analysis. You are supposed to present results for both parts in the following manner:

1. Upload your code.

2. Prepare a report with an analysis of results obtained at home.

The code and the report must be uploaded due to the deadline to Canvas.

UPLOAD A **SINGLE FILE** (a zip file) containing your code and the report. Name your file as follows: [vunetid] [assignment number].

Introduction

In this assignment we are going to implement a Variational Auto-Encoder (VAE) and Generative Adversarial Network (GAN). A VAE is a likelihood-based deep generative model that consists of a stochastic encoder (a variational posterior over latent variables), and a stochastic decoder, and a marginal distribution over latent variables. A GAN models an implicit distribution (i.e., it does not assume any form of the likelihood function) using a generator (a neural network for generating), and a discriminator (a neural network that mimics the likelihood function).

The goal of this assignment is twofold:

1. Implement a VAE and a GAN.

2. Analyze them and compare them.

Since this assignment is an advanced one, please think of appropriate architectures.

Part 1: Descriptions of a VAE and a GAN

Please describe both models in the report:

 Write objective functions for VAEs and GANs. Use standard normal prior p(z) for VAE.

Bonus: Derive VAE objective with the Normalizing Flow as a prior. What are the advantages of such model formulation?

1

Deep Learning 2020 Assignment 5

2. Please write down components of VAEs and GANs, and explain their roles.

3. Please pay attention to the adversarial loss. Please explain which loss you are going to use.

Part 2: Implementation

Implement a VAE and a GAN in PyTorch

- 4. Write a code in Python using PyTorch.
- 5. Please think of appropriate architectures (e.g., architectures that do not take too long to be trained).
- 6. Please use the ADAM optimizer.
- 7. Please implement the FID score using the pre-trained Inception V3 network from torchvision. Since there is no baseline pre-trained Inception V3 network for MNIST, please resize MNIST images (1x28x28) to appropriate sizes (3x32x32).

Bonus: As you know from the lecture, one of the advantages of VAE is that it allows us to approximate the likelihood. Implement a Negative Log Likelihood estimator for VAE. Use Importance Sampling to get the approximation (also referred to as IWAE, see appendix A in this paper).

- 8. Please use the following datasets:
 - a. MNIST
 - b. Imagenette: https://github.com/fastai/imagenette (160px: link)

REMARKS:

- Please do not try to achieve state-of-the-art scores, but still ensure that the models generate reasonable images.
- Please focus rather on reasonable architectures that you can learn on your machine.
- Please try to keep the number of weights in a VAE and a GAN to be comparable.
- You can use Pytorch Model Summary to calculate the number of weights (and summaries) in your models (https://pypi.org/project/pytorch-model-summary/).

Part 3: Analysis

Learn the VAE and the GAN using the ADAM optimizer and analyze their performance:

1. Analyze learning curves during training. If you use a separate validation data (NOT test!), please analyze learning curves on it as well.

Deep Learning 2020 Assignment 5

At the end of training (e.g., after a fixed number of epochs), calculate the test FID scores for the VAE and the GAN.

- 2. Sample from the VAE and the GAN. Analyze and compare samples.
- 3. Sample two points in the latent space and linearly interpolate between them. Then, for 10 points between the two original points, generate images. Analyze and compare the interpolations.
- 4. Perform 2 and 3 on MNIST and Imagenette.

Remarks:

- In order to monitor the progress during training, it is beneficial to observe generations' quality every 10 or so epochs.