MATH7017: Probabilistic Models and Inference, Project 2023 **(total: 40 points)**

Conditional Generation with Variational Autoencoders and Generative Adversarial Networks

Overview

In this project, you will work with Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs). You will build and train conditional versions of these models to generate MNIST digits given a digit class (see also *Labels improve subjective sample quality*, in week 12).

Objectives

1. **Conditional Variational Autoencoder (C-VAE)**: Implement a Conditional VAE that takes an MNIST image *and* its associated label as input. The model should learn to generate new images that resemble the input image given a class label.

2. **Conditional Generative Adversarial Network (C-GAN)**: Implement a Conditional GAN that takes a random noise vector and a class label as input to the generator, and produces a digit of the specified class. The discriminator should also be conditioned on the class label.

3. **Model Comparison**: Compare the performance of the two models. Discuss their strengths and weaknesses, and compare the quality of the generated samples. Use both qualitative (visual) and quantitative measures (if possible) for this comparison.

4. **Extra Challenge (Optional)**: Experiment with different architectures, training strategies, and techniques for improving the quality and diversity of the generated images (such as different types of regularisation, different architectures, etc.). Document your findings and provide explanations for the observed results.

Deliverables

1. **Code**: Well-commented Python scripts or Jupyter notebooks for both the C-VAE and C-GAN implementations. The code should include data loading, model definition, training loop, and a testing routine to generate new samples given class labels.

2. **Report**: A brief report that includes the following:
   - Model descriptions: An overview of the implemented models, including the chosen architectures and specific implementation details. Provide references to external sources and texts.
   - Training details: Information about the training process, such as loss curves, training times, and any issues encountered.
   - Results: Include generated samples from both models, and any quantitative results if computed.
   - Discussion: A comparison of the models, any insights gained from the project, and suggestions for future work or improvements.

The key goal of this project is not only to implement the models, but to gain a deeper understanding of VAEs and GANs, how conditioning on labels affects their performance, and the trade-offs between the two models.

Hints

Encoding the labels

One-hot encoding is a way of representing categorical variables as binary vectors. In the context of the MNIST dataset, the label of an image is an integer from 0 to 9 representing the digit. In one-hot encoding, each of these labels is transformed into a binary vector of size 10 (since we have 10 classes), where the position corresponding to the digit is set to 1, and all other positions are set to 0.

A simple example using PyTorch:

```
def to_one_hot(labels, num_classes):
    return torch.eye(num_classes)[labels]
```

If you run `to_one_hot(torch.tensor([0, 2, 9]), 10)`, it will return a tensor of shape $(3, 10)$ where each row is a one-hot encoded vector representing the respective label:

```
tensor([[1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 1.]])
```

Using One-Hot Encoded Labels in Conditional Models

In a conditional VAE or GAN, you want the model to generate data given a certain condition. In this case, the condition is the class label of the digit to be generated.

In a C-VAE, the one-hot encoded label is typically concatenated with the input image when feeding it into the encoder, and concatenated with the latent vector when feeding it into the decoder. This allows the VAE to learn a latent space that's not just based on the input image, but also the class label.

In a C-GAN, the one-hot encoded label is typically concatenated with the noise vector when feeding it into the generator, and with the image (real or fake) when feeding it into the discriminator. This allows the generator to create images based on the class label, and allows the discriminator to evaluate images based on the class label.

Remember that the purpose of conditioning is to allow the model to generate data that adheres to a specific condition. By providing the class label as an input to the model, the model can learn to generate data that's specific to each class.

Submit all your answers in two files to vUWS: your program code (as `.py` or python notebook), and a report as a PDF (preferably written in LaTeX, word processors like Word/Pages are also acceptable). Please include your name and student ID in all your files. Do not zip/rar/compress the file, and do not send your LaTeX or Word documents, just the PDFs.

Late answers and submissions by email will NOT be accepted.

Make use of multiple submissions to prevent a single late submission. Your last submission before the deadline will be marked. Give yourself time to submit the result – don't wait till last minute.

Marking:

1. **Implementation of Conditional Variational Autoencoder (C-VAE) – 10 points**
   - Correct implementation of the model architecture and conditioning on labels (5 points)
   - Successful training of the model and generating new samples given class labels (5 points)
2. **Implementation of Conditional Generative Adversarial Network (C-GAN) – 10 points**
   - Correct implementation of the model architecture and conditioning on labels (5 points)
   - Successful training of the model and generating new samples given class labels (5 points)
3. **Report - 20 points**
   - Clear and comprehensive description of the implemented models, training details, and results (6 points)
   - Detailed and insightful comparison of the models, including qualitative and quantitative metrics (6 points)
   - Discussion of challenges, interesting observations, and suggestions for future work or improvements (6 points)
   - Correct grammar, spelling, and presentation (2 points)