ECE 689 Spring 2025 - HW2

The Jupyter Notebook template for this homework is available on Canvas. Please upload your Jupyter Notebook file (.ipynb file) with all outputs and necessary derivations exported as a PDF or HTML to Canvas. All necessary coding documentation can be found in `https://pytorch.org/docs/stable/index.html`. **We highly encourage you to submit your derivation in LaTeX. If you choose to submit a hand-written derivation, please make sure your derivation is legible. If you have hand-written solutions, please scan/take a photo and insert them into their designated space.** You can follow this tutorial to insert images to .ipynb files: `https://www.geeksforgeeks.org/insert-image-in-a-jupyter-notebook/`

1 Problem 1: Score-Based Generative Model with Sliced Score Matching (30 pts)

Implement and train a score-based generative model using the sliced score-matching technique. The objective is to learn the score of the data distribution for generating new samples that closely resemble the training data.

Consider the Swiss Roll dataset, which can be found at `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_swiss_roll.html`. Please follow these steps to design and train the score-based model:

1. Call the Swiss Roll dataset with a noise level of 0.1 and visualize it in 3 dimensions.
2. Construct a simple neural network with residual blocks and Softplus activation function for estimating the data distribution's score. This network should accept an image as input and output a score estimate.
3. Implement the sliced score matching objective. Train the model to approximate the score of the data distribution.
4. Train the model using a straightforward optimizer like Adam, with a conservative learning rate (e.g., 1e-3).
5. Generate samples using the learned score and the unadjusted Langevin algorithm (ULA).

2 Problem 2: Structure Learning using the Chow-Liu Algorithm (30 pts)

Consider the following joint probability:

$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1) \times p(x_2|x_1) \times p(x_3|x_1, x_2) \times p(x_4|x_2) \times p(x_5|x_2, x_3)$$

1. Draw the Bayesian network corresponding to this joint distribution.
2. Which 2 edges can we drop to have a first-order dependency tree?

Consider the following explicit joint distribution:

$$\begin{aligned}
&p(x_1) = \mathcal{N}(0, 1), \\
&p(x_2|x_1) = \mathcal{N}(x_1, 2), \\
&p(x_3|x_1, x_2) = \mathcal{N}\left(x_1 + x_2^2, x_1^2 + 1\right), \\
&p(x_4|x_2) = \mathcal{N}(2x_2, 4), \\
&p(x_5|x_2, x_3) = \mathcal{N}\left(x_3 + x_2^2, x_2^2 + 1\right),
\end{aligned}$$

1. Generate samples from the given distribution.

2. Apply the Chow-Liu algorithm to the generated data, by numerically estimating the mutual information terms.

3. Draw the Tree learned by the Chow-Liu algorithm.

Additional details about Chow-Liu Tree Structure Search algorithm can be found in this lecture: `https://www.cs.cmu.edu/~guestrin/Class/10701/recitations/r10/11152007chowliu.pdf`

3 Problem 3: Estimate Treatment Effects using TARNet (40 points)

Design a TARNet model (`https://proceedings.mlr.press/v70/shalit17a/shalit17a.pdf`) as follows:

1. Consists of 5 shared layers.

2. Each sub-networks h_0 and h_1 has 5 layers.

Use an IPM measurement to penalize the discrepancy between the representations of the different treatment groups (e.g., a Wasserstein distance, Maximum mean discrepancy), with $\alpha = 0.3$ to measure the penalty after each mini-batch.

Train this model to learn the individual treatment effects (ITE) from the IHDP dataset for train and test data. This dataset can be found via `https://causalforge.readthedocs.io/en/latest/user_guide/Loading_Causal_RW_Benchmarking_Datasets.html`. Here, the task is to estimate ITE for all units in a sample for which only the factual outcome of one treatment is observed. Please report the graph of training (factual) loss and the error in estimating the ITE.

Additional details about the Wasserstein distance can be found in the following:

- `https://www.stat.cmu.edu/~larry/=sml/Opt.pdf`
- `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wasserstein_distance.html`