Deep Learning
Applicative Questionnaire

Q1.

Implement an MLP for the classification of the Dry Bean dataset.
Dataset link: https://archive.ics.uci.edu/dataset/602/dry+bean+dataset

Q2.

Design and implement a fully connected autoencoder using PyTorch with three encoder and three decoder layers. Utilize the Street View House Numbers (SVHN) dataset for training and evaluate the autoencoder's performance in terms of reconstruction accuracy (SSIM, PSNR). Further, fine-tune the autoencoder to function as a denoising autoencoder by introducing random noise to the input data during training.

Train the model under this denoising condition and thoroughly assess its capability to reconstruct clean samples from noisy inputs. Provide insights into the impact of denoising on the model's overall performance and reconstruction quality.

Q3.

In this assignment, you are to perform greedy-layer wise pretraining(ref link: https://proceedings.neurips.cc/paper/3048-greedy-layer-wise-training-of-deep-networks.pdf). You have two options of implementing it, both of which have the same marks. Use the CIFAR 10 dataset for both the cases.

Option 1:
Create a deep neural network which consists of atmost 2 convolutional blocks. Each block must have a convolutional layer, activation function(Leaky-ReLU,ELU) a and pooling layer(Adaptive Average Pooling). You need to train this network in a greedy-layer wise

manner. Appropriate plots must be made comparing the train loss obtained by successive addition of blocks. You must also use appropriate metrics for comparison.

Option 2:
Create a convolutional autoencoder whose training needs to be done in a greedy-layer wse manner. You must add atmost 2 convolutional blocks(Each block is as described in option 1). This needs to be trained on the MSE loss. After training, add an additional linear layer to the architecture and finetune it on the cross-entropy loss for multi-class classification on the dataset. Alter between Leaky-ReLU as activation functions and use Adaptive Average pooling as the pooling layer. Appropriate plots must be made like T-SNE and train loss curves. To check for efficacy, you must also use appropriate metrics like a confusion matrix.

— —

Q4.

In this lab exercise, your task is to employ Long Short-Term Memory (LSTM) networks for time series prediction using the Airplane Passengers dataset available for download. Begin by exploring and preprocessing the dataset, handling missing values, and creating relevant features. Visualize trends and patterns using line plots, seasonal decomposition, and other techniques. Design an LSTM model architecture suitable for time series forecasting, experimenting with layers, hidden units, and activation functions. Tune hyperparameters such as learning rate and batch size for optimization. Compile the model with appropriate loss functions and optimizers, then split the dataset into training, validation, and testing sets. Train the LSTM model and monitor for overfitting using the validation set. Evaluate model performance using Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). Finally, optimize the model based on evaluation results, implementing adjustments for enhanced forecasting accuracy. You can download the dataset using this link:

https://www.kaggle.com/datasets/rakannimer/air-passengers

— —

Q5.
Implement a variational autoencoder with UNet connections architecture using PyTorch and train it on the FaceMask dataset. For enhancing the model's performance implement annealing schedules for the KL divergence term. Evaluate the quality of the generated images using both qualitative and quantitative metrics, and discuss strategies for improving the VAE's ability to generate high-fidelity and diverse samples. Compare the results of VAE with a normal autoencoder. Also prove your results by properly visualizing the outputs.
Dataset Link

- https://www.kaggle.com/datasets/ashishjangra27/face-mask-12k-images-dataset/data

Evaluation Criteria:

Integration of U-Net connections into the VAE architectures.(2)

Plotting the training , validation loss/accuracy vs epoch curve (1)

Comparison of the reconstruction ability of the VAE on input images of test set(1)

Randomly sample a point from the latent space of the VAE and compare the output from the decoder with that of a normal autoencoder (1)

Plotting the t-SNE plot of VAE and comparison with normal autoencoder.(1)
Smoothness of Latent Space

- Interpolate between different samples to evaluate the smoothness of latent space and annealing schedules(1)

Take the latent representation of VAE and evaluate its performance on classification task(with or without mask) and also compare this with a regular autoencoder.(2)

Discuss how you can improve the VAE's ability to generate high fidelity and diverse samples.(1)

— —

Q6.

In this assignment, you need to create a DCGAN. The difference between this type of GAN and
the one introduced by Goodfellow et al. is that we use deep convolutional networks in the generator and discriminator architecture. Your task is to perform the following on the CelebA
dataset:

1) Create a DCGAN based architecture. The generator must have an architecture as described here. The discriminator must be as described here. [3]
2) Implement the min-max loss function for the GAN based on the GAN loss described here. [2]
3) Modify the loss function in the previous step to the one implemented by the Least-Squares GAN paper. [2]
4) Perform the training in both parts 2 and 3. Visualize the images created in both the cases after the training is done. Compare their performance based on the FID score metric. To know more of this metric, refer to this link: FID [3]

— —— — —— — —— — ——

Q7.
Implementation of CycleGAN in PyTorch

In this assignment, you will explore the versatility of CycleGAN, a popular framework for unpaired image-to-image translation. Your objective is to implement and evaluate CycleGAN models using innovative techniques and evaluation metrics.

Part 1: Spatial Transformations (5 marks)

Take a dataset of your choice and develop a CycleGAN architecture augmented with spatial transformation modules for improved image translation. Integrate spatial transformation networks (STNs) or attention mechanisms into the generator and discriminator networks to enable dynamic spatial warping and feature alignment. Train the enhanced CycleGAN model on a dataset containing images from diverse domains, such as day-to-night landscapes or aerial photos to satellite imagery. Analyze the effectiveness of spatial transformations in preserving structural consistency and enhancing the realism of translated images. Visualize the transformation process and compare the results with traditional CycleGAN models.

Part 2: Semi-Supervised Learning (3 marks)

Extend the CycleGAN framework to support semi-supervised learning for domain adaptation tasks. Incorporate semi-supervised discriminator or auxiliary classification modules into the CycleGAN architecture to leverage labeled data during training. Explore strategies such as consistency

regularization or entropy minimization to encourage the model to learn meaningful representations across domains. Train the semi-supervised CycleGAN model on a dataset with limited labeled samples, such as labeled images from the source domain and unlabeled images from the target domain. Evaluate the model's performance on both translation quality and domain adaptation accuracy, utilizing metrics such as classification accuracy and domain confusion matrices.

Part 3: Adversarial Robustness (2 marks)

Investigate techniques to enhance the adversarial robustness of CycleGAN models against adversarial attacks. Implement adversarial training methods or adversarial perturbations to defend against potential attacks targeting the generator or discriminator networks. Evaluate the robustness of the trained CycleGAN model against various adversarial perturbations and compare its performance with standard CycleGAN models. Discuss the implications of adversarial robustness in real-world deployment scenarios and potential applications in security-sensitive domains.

— —— — —— — —— — ——

Q8.

Download the pre-trained StyleGan(v1, v2 or v3).

1. Generate 10 realistic images using the StyleGAN.[4 marks]

2. Take your face image and 5 different face images of your friends (One image per friend). Perform feature disentanglement and linear interpolation between your face and your friend's face. [6 marks]

Q9.

In this assignment, you are to take a pretrained convnet and apply it in the tasks given below. Odd roll numbers must take inceptionnet-v1 and even roll numbers must take inceptionnet-v3.
Remove the last linear layer and replace it with appropriate linear layer(s) to train on the dataset given in this link. Your model must only train the last linear layer, thereby using the pretrained model.
Perform the finetuning and testing by dividing the dataset into train-test.[1+2 marks]
Create a function to output the saliency maps corresponding to any 1 image from each class in the following two cases:
Finetune only the last layer and test it.[2 marks]

Re-train the entire network on the new dataset and test it.[2 marks]

Evaluate the performance of the finetuned and original network based on the recall and accuracy metrics. Plot the training loss curve. Finally, write plausible explanation for the difference in metric values you obtained.[3 marks]

Dataset: https://www.kaggle.com/datasets/mahmoudreda55/satellite-image-classification

— —— — —— — —— — ——

Q10.

1. Train the GNN model for the classification of the MNIST images using the paper: https://iopscience.iop.org/article/10.1088/1742-6596/1871/1/012071/pdf (Dataset: MNIST). Note: Modify the architecture and use 4 gatconv layers and 3 Fc layers

(4 marks)
Github link:
https://github.com/Anton-Cherepkov/gnn-mnist-classification?tab=readme-ov-file
2. Tune the hyperparameters of the GNN model. (At least 2 sets of hyperparameters running logs must be shown) (2 marks)
3. Visualize the images in graph format (All the classes). (2 marks)
4. Compare the accuracy and training time to a typical CNN of your choice. (4 cnn layers). (2 marks)

— —— — —— — —— — ——

Q11.

Implement the Segnet architecture following the guidelines provided in the original paper (https://arxiv.org/pdf/1511.00561.pdf).
Download and preprocess the Pascal VOC dataset (http://host.robots.ox.ac.uk/pascal/VOC/). Set up data loaders for training and validation datasets.
Train the model and monitor the training progress by logging metrics such as loss and pixel accuracy. [4 marks]
Evaluate your trained model on the test dataset. Generate segmentation masks for objects and compute metrics such as pixel accuracy, mean IoU, and F1 score for segmentation. [2 marks]

Change the backbone model from VGG to some lightweight model and compare the performance. [4 marks]

— —— — —— — —— — ——

Thank you!!