Deep Learning Lab (AI29204)
Mid Sem Test --- 17/02/2026
Marks --- 100

Instructions:

- **Create a folder with the section number (e.g., Section_A) and add all the relevant files there**
- **Finally, create a folder with your roll number, copy individual section folders there, compress it and upload roll_number.zip**
- **Answer Section B, and any one from Section A.**

Section A

Question 1: Your task is to build an Artificial Neural Network (ANN) to predict hourly bike rental demand. You will implement a baseline model and then systematically apply regularization techniques to improve generalization. [35]

Dataset : UCI Bike Sharing Dataset (Hourly Data)

1. **[Task 1] Data Loading and Preprocessing**
   i. Load the dataset (hour.csv)
   ii. Drop unnecessary columns: Remove ‘instant’, ‘dteday’, ‘casual’, ‘registered’
   iii. Encoding categorical features : One Hot Encoding for ‘ season’, ‘weathersit’
   iv. Split the data into train(70%), test(15%) and validation(15%) sets
   v. Create PyTorch Dataloaders with your choice of batch size
   vi. Print shapes of train, test and validation sets

2. **[Task 2] Baseline Model Training**
   i. Implement a Feed Forward Neural Network with 2 hidden layers. Choose the number of units per layer.
   ii. Choose appropriate hidden layer activation and output activation
   iii. Perform training under two settings and compare results
   Loss Function : MSE, Optimizer: SGD, LR=0.001
      - Implement the loss function and optimizer from scratch
      - Use Pytorch’s inbuilt loss and optimizer

   iv. For each setting:

      - Track training and validation losses for each epoch, plot loss curves
      - Report final Test MSE, Test MAE and Test R^2 score
      - Create a scatter plot comparing the predicted value to the ground truth

3. **[Task 3] Elastic Net Regularization**
   Elastic Net combines L1 (Lasso) and L2 (Ridge) regularization:

$$\mathcal{L}_{total} = \mathcal{L}_{MSE} + \lambda_1 \sum |w| + \lambda_2 \sum w^2$$

   Implementation:
   i. Use weight_decay parameter in the optimizer for L2 regularization.
   ii. Manually compute and add L1 regularization to the loss function.
   iii. Choose suitable values of (λ1, λ2). Explain your choice
   iv. Report Test MSE, Test MAE and Test R^2 score

4. **[Task 4] Dropout**
   i. Modify the baseline to include dropout layers after each hidden layer. Experiment with dropout rate and choose the best one based on performance on the validation set.
   ii. Report Test MSE, Test MAE and Test R^2 score
   iii. Compare training vs validation loss. Did dropout help reduce overfitting? Explain.

5. **[Task 5] Batch Normalization**
   i. Modify the baseline architecture to include Batch Normalization layers. Apply BatchNorm before the activation function.
   ii. Monitor how quickly the loss decreases compared to the baseline. Compare the stability of training (variance in loss values).
   iii. Report Test MSE, Test MAE and Test R^2 score

6. **[Task 6] Comparison and Analysis**
   i. Loss Curves Plot: A single figure with training loss curves for all 4 model variants: Include a legend identifying each curve.
   - Baseline
   - Elastic Net
   - Dropout
   - Batch Normalization

   ii. Summary Table: Create a table with the following format:

Model	Test MSE	Test MAE	Test R^2	Training Time
Baseline				
Elastic Net (λ1=?, λ2=?)				
Dropout (p=?)				
Batch Normalization				

Question 2: Use transfer learning to adapt a pretrained convolutional neural network to a new task. You will also use the learned representations from the pretrained model for a downstream classification task. Use the CIFAR-10 dataset available in torchvision.datasets. [35]

Task 1:

a. Load the CIFAR-10 training and test datasets.
b. Apply appropriate transformations:
    i. Resize images to 224 × 224
    ii. Convert images to tensors
    iii. Normalize using ImageNet mean and standard deviation
c. Create PyTorch DataLoader objects for training and testing.

Task 2:

a. Load a pretrained ResNet-18 model from torchvision.models.
b. Print the model architecture.

Task 3:

a. Freeze all layers of ResNet-18.
b. Replace the final fully connected layer so that the model outputs 10 classes.

Task 4: You will now train only the newly added classification layer, while keeping the pretrained layers frozen. Train only the final layer

a. Define a suitable loss function and optimizer.
b. Write a training loop for 2 epochs.
c. Print the training loss after each epoch.

Task 5: Instead of directly using the model for classification, you will extract the learned representations from ResNet-18.

a. Write a function that extracts the 512 dimensional feature vector before the final fully connected layer.
b. The function should take a batch of images as input and return feature vectors.

Task 6: You will now train a classifier using the 512 dimensional representations extracted from ResNet-18. Make sure to use the same training conditions used in Task 4 for training.

Task 7: Evaluate the performance of your trained model on the CIFAR-10 test set and compare two approaches. Use suitable metrics for evaluating.

Section B

Question 3: This question checks adversarial attacks on CNN [35]

Task 1 - 2 Marks
Load the FER-2013 dataset from torchvision (train and test split). If torchvision does not contain the complete dataset, download it from Kaggle in .csv format: FER-2013 Dataset. Ensure proper preprocessing and normalization of the dataset.

Task 2 - 2 Marks
Divide the train split into training and validation partitions. Ensure randomness in the split to avoid bias. Define corresponding dataloaders for each partition to facilitate batch-wise data loading.

Task 3 - 1
Marks Visualize the dataset by displaying 3 randomly selected samples from each class. Ensure labels are correctly assigned.

Task 4 - 7 Marks
Design a custom Convolutional Neural Network (CNN) model with the following properties:
● Use 2 convolutional layers. Customize the architecture design to make sure the architecture is flexible with any input image dimensions
● Implement residual connections between successive layers, i.e., 2 residual connections, one for each layer.
● Ensure the architecture is fully convolutional with no fully connected (FC) layer.
● The number of channels for the 2 layers should be [32,64].
● Use Relu as an intermediate activation function and an appropriate output activation function.

Task 5 - 5 Marks
Train the custom CNN model using a suitable loss function and the Adam optimizer with a learning rate of 1x 10 -3 (1M). Track the performance by:
● Plotting the training and validation losses against the number of epochs.
● Ensuring appropriate hyperparameter tuning for optimal performance.

Task 6 - 2 Marks
Evaluate the trained model on the test set and report the following metrics:
● Accuracy
● Precision
● Recall
● Confusion Matrix

Task 7 - 6 Marks
Generate Class Activation Maps (CAMs) using Grad-CAM for 1 randomly selected sample from each class. Visualize and analyze the activation regions.

Task 8 - 10 Marks
Implement a targeted Fast Gradient Sign Method (FGSM) adversarial attack to modify test images originally classified as “sad” so that they are misclassified as “happy” by the trained model.

Report the success rate of the attack, which is defined as the fraction of samples that were initially correctly classified as "sad" but misclassified as "happy" after the adversarial perturbation.