Statistical Methods in AI

Assignment 3

Question 1

The CIFAR-10 and CIFAR-100 datasets are benchmarks for image classification tasks in computer vision. In this exercise, you will explore how different neural network architectures perform on these image datasets.

Tasks

1. **Multilayer Perceptron (MLP):**
   - Develop a Multilayer Perceptron (MLP) model with an appropriate architecture (number of layers, hidden units, etc.).
   - Train the MLP model on the CIFAR-10 dataset.
   - Evaluate its performance on the test set using metrics like accuracy and loss. Analyze the results.
2. **Convolutional Neural Network (CNN):**
   - Design a simple Convolutional Neural Network (CNN) architecture with convolutional layers, pooling layers, and fully connected layers.
   - Train the CNN model on the CIFAR-10 dataset.
   - Evaluate its performance on the test set using the same metrics as the MLP. Critically compare CNN's performance to the MLP's.
3. **Transfer Learning with VGG:**
   - Utilize a pre-trained VGG model (such as VGG16 or VGG19) available in common deep learning libraries.
   - Adapt the pre-trained VGG model for the CIFAR-10 or CIFAR-100 classification task.
   - Evaluate its performance on the test set. Analyze the benefits of transfer learning compared to your MLP and CNN models trained from scratch.

Analysis and Discussion

- Compare the test set accuracy and loss across the MLP, CNN, and the VGG-based model.
- Discuss the reasons behind the differences in performance. Explain how CNNs leverage the spatial structure of images for better feature extraction compared to MLPs.
- If you used the VGG model, elaborate on how transfer learning helped improve performance or reduce training time.

Tips (Not Graded):

- **Data Preprocessing:** Consider normalizing the data and employing data augmentation techniques if needed.
- **Hyperparameter Tuning:** Experiment with different learning rates, optimizers, and network architectures to fine-tune your models.
- **Visualization:** Visualize the filters learned by the CNN to gain insights into what features are being detected

Question 2

You are required to use the MNIST dataset for this question. Implement the Models using PyTorch. For the accuracy part you can use Structural Similarity (SSIM) from scikit-learn to compare 2 images.

Tasks

1. Implement the Encoder - Decoders Models

2. Add Gaussian Noise to the images and train the model. Use various values of sigma (3 different values) in the Gaussian and plot the average SSIM test score for these 3 varying sigma values. Explain your observations.

3. Now keep the Sigma constant and change the bottleneck dimensionality (3 different values) of the Model and plot the average SSIM test score for these 3 values. Explain your observations.

Question 3

Given the IMDB Movie Review Dataset, create an RNN model that predicts whether the given review is negative or positive.

You need to create your Dataset, Dataloader and Model. Keep your code modular and avoid hardcoding any parameters. This will allow you to experiment more easily.

Plot graphs for loss and accuracy for each epoch of a training loop. Try using wandb for logging training and validation losses, accuracies (especially for hyperparameter tuning)

Tasks

1. **RNN Model:**
   - Build a Dataset from the IMDB Movie Review Dataset by taking reviews with word count between 100 and 500. Perform text processing on the movie reviews and create a word to index mapping for representing any review as a list of numbers.

     Create Dataloaders for the train, test and validation datasets with appropriate batch sizes. Create the Model class for the RNN Model. Create functions for running model training and testing
   - Incorporate stemming/lemmatization when doing text preprocessing using the NLTK library. What changes do you observe?
   - In the Model class, experiment with only picking the last output and mean of all outputs in the RNN layer. What changes do you observe?
2. **Hyperparameter Tuning:**
   - Starting with the best configurations based on the above experiments, experiment with 5 different hyperparameter configurations. You can change the size of the embedding layer, hidden state, batch size in the dataloader.

     Evaluate the performance of the configurations on the validation sets using metrics like accuracy and loss. Analyze the results.
3. **After RNNs:**
   - Keeping all the parameters same, replace the RNN layer with the LSTM layer using nn.LSTM. What changes do you observe ? Explain why LSTM layer would affect performance..