MCA572– Neural Networks and Deep Learning

V MCA

27-09-2024

Regular lab Question – 3

Develop a neural network using backpropagation to classify images from the CIFAR-10 dataset. The dataset contains 60,000 32x32 color images divided into 10 classes (airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks).

Your objective is to build a neural network model, train it using backpropagation, and evaluate its performance.

You can download the dataset

https://www.kaggle.com/c/cifar-10/

https://www.cs.toronto.edu/~kriz/cifar.html

Instructions:

1. **Data Preprocessing:**
   - Load the CIFAR-10 dataset.
   - Perform necessary data preprocessing steps:
     - Normalize pixel values to range between 0 and 1.
     - Convert class labels into one-hot encoded format.
     - Split the dataset into training and test sets (e.g., 50,000 images for training and 10,000 for testing).
     - Optionally, apply data augmentation techniques (such as random flips, rotations, or shifts) to improve the generalization of the model.
2. **Network Architecture Design:**
   - Design a feedforward neural network to classify the images.
     - **Input Layer:** The input shape should match the 32x32x3 dimensions of the CIFAR-10 images.
     - **Hidden Layers:** Use appropriate layers.
     - **Output Layer:** The final layer should have 10 output neurons (one for each class) with a softmax activation function for multi-class classification.

Question:

- Justify your choice of network architecture, including the number of layers, types of layers, and the number of neurons/filters in each layer.

3. **Activation Functions:**
    - Choose any two appropriate activation functions for the hidden layers (e.g., ReLU, sigmoid, or tanh).
    - Explain why you selected a particular activation function and its role in the backpropagation process.
4. **Loss Function and Optimizer:**
    - Use any two loss functions and compare with the categorical cross entropy since this is a multi-class classification problem.
    - Select an appropriate optimizer (e.g., SGD, Adam, RMSprop) and explain how the learning rate affects the backpropagation process.

Question:

- How does the choice of optimizer and learning rate influence the convergence of the network? How would you adjust the learning rate if the model is not converging properly?

5. **Training the Model:**
    - Implement backpropagation to update the weights and biases of the network during training.
    - Train the model for a fixed number of epochs (e.g., 50 epochs) and monitor the training and validation accuracy.

Question:

- How does backpropagation update the weights in each layer, and what role does the learning rate play in this process?

6. **Model Evaluation:**
    - After training, evaluate the performance of your model on the test set.
    - Calculate accuracy, precision, recall, F1-score, and the confusion matrix to understand the model’s classification performance.

Question:

- How can you further improve model performance if the accuracy is low?

7. **Optimization Strategies:**
    - Discuss optimization strategies such as:

        - Early stopping to prevent overfitting.
        - Learning rate scheduling for smoother convergence.
        - Weight initialization techniques to ensure efficient learning.

Question:

    - Why is weight initialization important, and how does it impact the convergence of your network?

8. **Report:**
    - Provide a detailed report/documentation that includes (in Colab file):
        - The architecture of the network and justifications for the choices made.
        - Training and test accuracy with relevant plots showing loss and accuracy over the epochs.
        - Hyperparameter values used (learning rate, batch size, number of epochs, etc.).
        - A brief discussion of the challenges you faced and how you addressed them.

Deliverables:

- Submit your code along with documentation (as a Google Colab file and Github link).
- Include comments in your code explaining each step clearly.
- The Colab file should also contain performance metrics and any visualizations (such as training curves or confusion matrices) that demonstrate the model's effectiveness.

Program Evaluation Rubrics

Evaluation Criteria	
5 marks	**C1-Implementation, Correctness and Complexity**
2 marks	**C2-Documentation and Visualization**
3 marks	**C3-Concept Clarity and Explanation**

General Instructions

1. The file you have to save with your name, last 3 digits of register number and program number “Aaron_201_Lab1”.
2. The implemented code you have to upload in Github and in the Google Classroom in the given scheduled time.
3. Failure to upload within the allotted time will result in the loss of all marks for the corresponding lab exercise.