Second Assignment

Shahid Beheshti University - Bachelor’s Program

Artificial Neural Networks - Winter 2024

Theoretical Exercises

Exercise 1

What are exploding and vanishing gradients in neural networks? How can Adam Optimizer help us with them? Is LeakyReLU better or ReLU in avoiding vanishing gradients?

Exercise 2

How can we avoid overfitting? Name 9 methods and explain them in detail.

Exercise 3

Does dropout slow down training? Does it slow down inference (i.e., making predictions on new instances)? What about MC dropout?

Exercise 4

Compare and contrast L1 regularization and L2 regularization. How do they differ in terms of their effects on model weights? Explore the concept of elastic net regularization and its advantages over L1 and L2 regularization. In what scenarios would elastic net regularization be preferred? (first, describe them in detail and then compare them together with plots and their equations)

Exercise 5

Explain the concept of batch normalization and its role in improving the training of deep neural networks.

Exercise 6

Discuss the benefits of ensemble methods in improving the robustness and accuracy of MLP models.

Exercise 7

Investigate the use of uncertainty estimation techniques, such as Monte Carlo dropout or Bayesian neural networks, for quantifying model uncertainty and improving decision-making in MLPs.

Exercise 8

Discuss the limitations of traditional performance metrics such as accuracy and propose alternative evaluation metrics for assessing model performance in imbalanced datasets or noisy environments.

Exercise 9

Analyze the convergence properties of stochastic optimization algorithms such as stochastic gradient descent (SGD) and Adam. Discuss conditions under which these algorithms converge and potential challenges in practice.

Exercise 10

Explore the challenges associated with non-convex optimization in training deep neural networks. Discuss issues such as saddle points, poor local minima, and plateaus, and strategies to overcome them.

Exercise 11

Discuss second-order optimization methods such as Newton's method and the Quasi-Newton method. Explain how these methods differ from first-order methods like gradient descent and their advantages in terms of convergence speed and robustness.

Practical Exercise

Problem Description

In this practical exercise, you will work on predicting student performance using the Student Performance dataset. The dataset contains information about students' demographics, family background, and academic performance. Your task is to build an MLP model using PyTorch to predict students' performance accurately.

Dataset Description

The Student Performance dataset contains the following columns:

- school
- sex
- age
- address
- famsize
- Pstatus
- Medu
- Fedu
- Mjob
- Fjob
- reason
- guardian
- traveltime
- studytime
- failures
- schoolsup
- famsup
- paid
- activities
- nursery
- higher
- internet
- romantic
- famrel
- freetime
- goout

- Dalc
- Walc
- health
- absences
- G1 (First period grade)
- G2 (Second period grade)
- G3 (Final grade)

Tasks

1. **Data Loading and Preprocessing**

   Download the Student Performance dataset from the appropriate source.
   Load the dataset using Pandas or any other suitable library.
   Perform necessary preprocessing steps such as handling missing values, encoding categorical variables, and scaling numerical features.

2. **Data Splitting**

   Split the dataset into training and testing sets. Use a suitable ratio (e.g., 80% training, 20% testing) to ensure an adequate amount of data for training and evaluation.

3. **Model Implementation with PyTorch**

   Implement a Multilayer Perceptron (MLP) model using PyTorch to predict student performance.
   Design the architecture of the MLP model, including the number of input nodes, hidden layers, neurons per layer, and output nodes.
   Experiment with different activation functions, dropout layers, and regularization techniques to improve model performance.

4. **Training the MLP Model**

   Train the MLP model using the training data. Utilize techniques such as mini-batch gradient descent and backpropagation to update the model parameters iteratively.
   Monitor training progress by tracking metrics such as loss and accuracy on both training and validation sets.

5. **Model Evaluation**

   Evaluate the performance of the trained MLP model using the testing data.
   Calculate relevant evaluation metrics such as mean squared error (MSE), mean absolute error (MAE), and R-squared to assess the model's predictive performance.

6. **Analysis and Interpretation**

   Analyze the results of the MLP model and interpret its predictions.
   Identify factors that significantly influence student performance based on feature importance or coefficients from the model.
   Discuss potential interventions or strategies for improving student performance based on the analysis.

Dataset

You can download this dataset from this link.

Submission Guidelines

Submit your implementation as a Jupyter Notebook file (.ipynb).
Include any additional files or resources used in your implementation.
Provide a brief report documenting your approach, methodology, results, and any challenges faced during the implementation process.