University of Nebraska Lincoln

Practical Machine Learning
ECEN 478/878

Programming Assignment 3

Fall 2024

Designing and Training MLP Classifiers Using Backpropagation

Assignment Objective

The objective of this assignment is to understand how to effectively apply Multi-Layer Perceptron (MLP) models to solve multi-class classification problems and explore strategies for optimizing model performance.

- Part A: Comprehensive Study of MLP Models
- Part B: MLP Design and Optimization

Assignment Instructions

Note: You must use TensorFlow Keras API to create the MLP models. You are allowed to use Python libraries such as Scikit-Learn, Pandas, NumPy, and Matplotlib.

i. The code should be written in two Jupyter Notebooks. Use the following naming convention.
   lastname_assignment3a.ipynb
   lastname_assignment3b.ipynb
ii. For each experiment, use unique names for your machine learning models.
iii. If you need to partition the data multiple times for training different models, use distinct names for each training and testing subsets.
iv. The Jupyter Notebook should be submitted via Canvas.

The programming code will be graded on **implementation, correctness, and quality of the results**.

Score Distribution

ECEN 478: 100 points
ECEN 878: 115 points

Part A: Comprehensive Study of MLP Models

You will create Multi-Layer Perceptron (MLP) neural network models based on the specification. The MLP model will be trained using the Backpropagation algorithm that is implemented using the mini-batch Stochastic Gradient Descent (SGD) optimization algorithm.

Mini-batch Size & Training:
Set the mini-batch size to 64. Train the network for 50 epochs, with ***early stopping*** enabled.

Dataset:
You will use the MNIST (Modified National Institute of Standards and Technology) dataset, which is a set of 70,000 small images of digits handwritten by high school students and employees of the US Census Bureau. Each image is labeled with the digit it represents.

There are 70,000 images. Each image is grayscale 28 x 28 pixels, and each feature simply represents one pixel's intensity, from 0 (white) to 255 (black). The task is to classify a given image of a handwritten digit into one of 10 classes representing integer values from 0 to 9, inclusively.

You may directly load this dataset using the Keras API:
https://keras.io/api/datasets/mnist/

The train set contains 60,000 images, from which you should ***randomly*** select 5000 images as the validation set.

Experiments:
Perform multi-class classification on the MNIST dataset using the 10 models listed below. After loading the dataset, create a separate validation set with 5,000 samples. Scale the data using the min-max scaling technique.

Build the models based on the specified architecture, SGD learning rate, and layer configurations.

Report:

- In the last column of the "Layer Configuration" table (Table 2, provided below), report the following for each model: training accuracy, test accuracy, and the number of epochs at which training stopped. Report accuracy values to **3 decimal places**.
- Provide learning curves (accuracy & loss) for each model.

[**50 pts**]

Note: Grading will be based on the quality of the results obtained. Simply implementing the model will not earn full credit.

Part A: Table 1

Architecture		**SGD Learning Rate**
Experiment 1 to 7	2 hidden layers Hidden layer 1: neurons=300 Hidden layer 2: neurons=100	0.1
Experiment 8	2 hidden layers Hidden layer 1: neurons=300 Hidden layer 2: neurons=100	0.5
Experiment 9	10 hidden layers each with 100 neurons	0.1
Experiment 10	20 hidden layers each with 100 neurons	0.1
Experiment 11 [**only for graduate students**]	4 hidden layers. - 1st layer: 100 neurons - 2nd layer: 300 neurons - 3rd layer: 300 neurons - 4th layer: 100 neurons	Use an optimal learning rate; you will need to determine it and include the rationale in the report.

[Experiment 11: **10 pts**]

For all experiments, the final classification layer should have 10 neurons (corresponding to the number of classes), and the activation function for this layer must be "softmax".

Part A: Table 2

Layer Configuration				**Result**
	kernel_initializer (all layers)	activation (hidden layers)	Dropout (hidden layers)	
Experiment 1	zeros	sigmoid	No	Train accuracy = Test accuracy = Epochs =
Experiment 2	ones	sigmoid	No	Train accuracy =

				Test accuracy = Epochs =
Experiment 3	random_normal	sigmoid	No	Train accuracy = Test accuracy = Epochs =
Experiment 4	random_normal	tanh	No	Train accuracy = Test accuracy = Epochs =
Experiment 5	random_normal	relu	No	Train accuracy = Test accuracy = Epochs =
Experiment 6	random_normal	relu	Yes Hidden layer 1: rate=0.1 Hidden layer 2: rate=0.1	Train accuracy = Test accuracy = Epochs =
Experiment 7	random_normal	relu	Yes Hidden layer 1: rate=0.5 Hidden layer 2: rate=0.1	Train accuracy = Test accuracy = Epochs =
Experiment 8	random_normal	relu	Yes Hidden layer 1: rate=0.5 Hidden layer 2: rate=0.1 SGD learning rate=0.5	Train accuracy = Test accuracy = Epochs =
Experiment 9	random_normal	relu	No	Train accuracy = Test accuracy = Epochs =
Experiment 10	random_normal	relu	No	Train accuracy = Test accuracy = Epochs =
Experiment 11	Use optimal values for the hyperparameters. Your goal is to maximize test accuracy. Apply your knowledge from experiments 1–10 and conduct an empirical investigation.			Train accuracy = Test accuracy = Epochs =

Answer the following questions.

- Q-1) Among experiments 1 to 5, which experiment performed the best (based on test accuracy & the number of trained epochs)? Explain why.

- Q-2) Among experiments 1 to 5, which experiment performed the worst (based on test accuracy)? Explain why.

- Q-3) Compare experiment 6 with experiment 7 and determine which model experiences less overfitting. Explain why. To answer this question, use the training accuracy, test accuracy, and learning curves from these two experiments. Include these measures (statistics and learning curves) in your explanation.

- Q-4) Compare experiment 7 with experiment 8. Explain the change in the number of epochs in experiment 8. If the number of epochs has increased, explain why. If it has decreased, explain why.

- Q-5) Compare experiment 8 with experiment 9. Present the accuracy learning curves for both models and explain the differences observed. Which model performed poorly, and what factors contributed to its lower performance?

- Q-6) Compare experiment 9 with experiment 10. Identify which model performed poorly and explain the reasons for its lower performance.

[18 pts]

Part B: MLP Design and Optimization

You will design an MLP to achieve optimal performance on the following dataset.

Dataset:
The CIFAR-10 dataset (Canadian Institute For Advanced Research) contains 60,000 32 x 32 color images in 10 different classes. The 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images of each class. You may directly load this dataset using the Keras API:
https://keras.io/api/datasets/cifar10/

The train set contains 50,000 images, and the test set contains 10,000 images.

Pre-processing: **[5 pts]**
You will need to perform some pre-processing on the data.

- The train and test labels will be loaded as a 1D column vector. You need to convert the 1D vector into a 1D array.

  You may convert the label data into a 1D array by using the NumPy "ravel" function:
  https://numpy.org/doc/stable/reference/generated/numpy.ravel.html

  Example: y_train = y_train.ravel()

- Scale the features using the min-max scaling technique, i.e., by dividing the training & test data matrix by 255.0.

  Example: X_train = X_train/255.0

Experiments:

Experiment 12) Design an MLP classifier and perform multi-class classification. Your design must be optimal by determining the best values for the following hyperparameters:

hidden layers, neurons in each hidden layer, weight initializer, activation function for the hidden layers, regularization techniques (weight decay, dropout, early stopping), mini-batch size, and SGD learning rate.

Important Instructions:

- You must not use hyperparameter tuning techniques such as grid search, random search, or Hyperband. Instead, hyperparameters should be selected based on heuristics.
- You will need to report your heuristics, the combinations of hyperparameters you tried, the results obtained, and any insights gained from your experiments. See Question #7 below.
- If you used any AI tools, such as ChatGPT, to design your hyperparameter selection heuristics, you must report the question you asked to the AI tool, the response you received, and your rationale for believing it to be a correct response, considering the possibility of hallucinations in AI-generated outputs. Critical evaluation of AI suggestions is essential before adoption. See Question #8 below.
- Additionally, follow best practices for training an MLP, such as utilizing a validation dataset (either static or dynamic) and analyzing learning curves to monitor training and validation performance throughout the training process.
- Report the following: A two-column table (use Table 3) showing the attributes in the first column and their corresponding values in the second column, along with the training accuracy, test accuracy, test confusion matrix, test classification report, and learning curves (accuracy and loss).

Part B: Table 3

Number of hidden layers	
Number of neurons in each hidden layer	- 1st Layer: - 2nd Layer: - Etc.
Weight initializer function	
Activation function (hidden layer)	
Regularization technique(s)	
Size of mini batch	
Learning rate	
Train accuracy	
Test accuracy	

Note: Experiment 12 will be graded primarily on the quality of test performance. No assistance will be provided regarding optimal design. You are free to explore any approach you choose. I will not indicate whether your results are optimal or not.

[18 pts]

Answer the following questions. [**4 + 2 + 3 pts**]

- Q-7) What heuristics did you use to select hyperparameters for your MLP classifier? Please report the combinations of hyperparameters you tried, the results you obtained from those combinations, and any insights gained from your experiments.
- Q-8) If you utilized any AI tools, such as ChatGPT, in designing your hyperparameter selection heuristics, please report the question you asked, the response received, and how you critically evaluated the AI tool's suggestions before adopting them, especially considering the potential for hallucinations in the responses.
- Q-9) Discuss how your selected hyperparameters influenced the performance on the test data. What justifications can you provide for these choices, and what trade-offs or limitations did you encounter?

Deliverables
You will submit two artifacts.

- Two Jupyter Notebooks containing all experiments for Parts A and B, respectively. For each experiment, display the required performance measures. Ensure your code is clearly annotated and includes header descriptions for each block. ***Note that if the notebook lacks annotations or has poor annotation, 5 points will be deducted***.
- A **PDF** copy of the written report, including a cover page (available on Canvas). The report must include the following items.

a) Part A: Using the table 2 (given in Part A) report training accuracy, test accuracy, and epochs at which the training stopped.
b) Part A: learning curves for experiments 1 to 10.
c) Part A: Answers to Q-1 to Q-6.
d) Part B: a 2-column table that shows the model attributes in the 1st column and their values in the 2nd column, train accuracy, test accuracy, test confusion matrix, the test classification report, and learning curves (accuracy and loss).
e) Part B: Answers to Q-7 to Q-9.

The PDF file should be named as follows: lastname_assignment3.pdf