

Deep Learning Assignment

Advanced Management Programme in Business Analytics, Indian School of Business

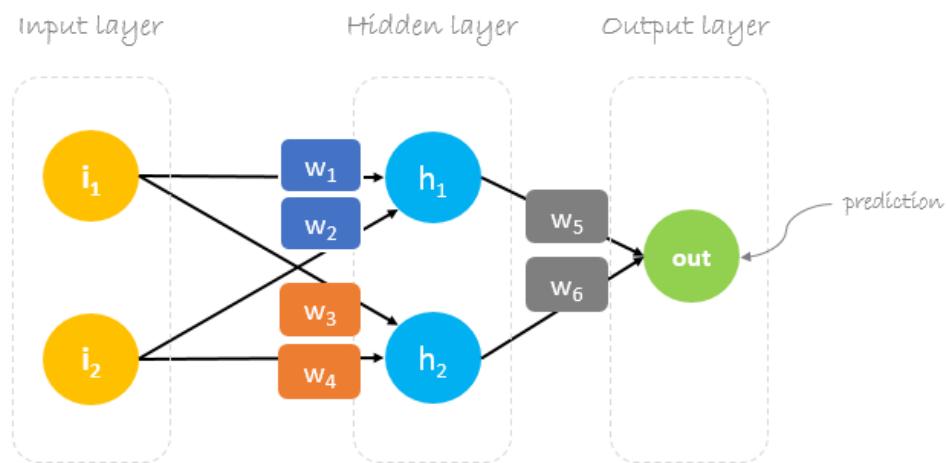
Honor Code: 2N-b, Weightage: 45%

(DO NOT SHARE THIS ASSIGNMENT WITH ANYBODY, PRIVATELY OR PUBLICLY)

Part 1: Understanding Neural Network Forward Propagation and Backpropagation

Objective: The objective of this assignment is to assess your understanding of how neural networks perform forward propagation to make predictions and backpropagation to update weights during training.

You are provided with a simple feedforward neural network architecture with one hidden layer. Implement forward propagation to compute predictions for a given input. Implement backpropagation to compute gradients and update weights using gradient descent.



Grading Components – 100 points:

1. Implement forward propagation (15 points)

Randomly initialize the weights based on the Xavier initialization technique discussed in the class. The initialization will assign weights to $\{W_1, W_2, W_3, W_4, W_5, W_6\}$. Report the weights that you assigned.

Use input 1 (i_1) = 2, and input 2 (i_2) = 4

Use Forward propagation and report the output (out) that represents the prediction for the inputs $i_1 = 2$, and $i_2 = 4$, based on the weights you have assigned. Assume h_1 and h_2 neurons use a linear activation function.

2. Derivation for Backpropagation (20 points)

Using Mean Square loss as the loss function and learning rate as a parameter (ϕ), show all steps needed for backpropagation. You will need to derive equations for all six connection weights.

3. Assuming the true output to be 1, learning rate = 0.05 and MSE loss, calculate the new weights using gradient descent (20 points).
Show the calculations and report the new weights.
4. Implement another Forward Propagation with “Relu” activation function applied to neurons representing “h1” and “h2”. Keep the other parameters the same as suggested in step 1. (15 points). Show the calculations.
5. Assuming the true output to be 1, learning rate = 0.05, and MSE loss and “Relu” activation function applied to hidden layer neurons, calculate the new weights using gradient descent. Show the calculations and report the new weights. (30 points)

Deliverables:

1. Jupyter notebook containing your code and results.
2. A report summarizing your findings with a section on each grading component as mentioned above.

Part 2: Predicting Customer Churn using Neural Networks

Objective: The objective of this assignment is to develop a neural-network model that can predict customer churn for a telecom company. You will use a dataset containing customer information, and whether or not the customer churned.

Tasks:

1. Download the dataset from Kaggle
<https://www.kaggle.com/datasets/blastchar/telco-customer-churn>
2. Load the dataset and perform exploratory data analysis.
3. Pre-process the data by converting categorical variables to numerical variables, and scaling the data.
4. Split the data set in train (80%) and test (20%) sets. Train the model using the training set.
5. Implement and compare three approaches:
 - **Approach A:** Separate model for predicting tenure . While predicting tenure, keep either “TotalCharges” or “MonthlyCharges”, but not both (for obvious reasons)
 - **Approach B:** Separate model for predicting churn. Check if there is a benefit to balancing the churn classes in the training data. Note that test data should reflect the real-world distribution; therefore, it should never be balanced.

- **Approach C:** Joint model that predicts both tenure and churn simultaneously using shared layers and task-specific output heads. Keep either “TotalCharges” or “MonthlyCharges”, but not both in the input data.
- Evaluate the model using the testing set and report the accuracy, precision, recall, and F1 score. Experiment with different hyperparameters (learning rate, layer count and neurons per layer, Epochs) and architectures to improve the performance of the model. On the test dataset, compare the losses of these approaches and discuss when multi-task learning is beneficial. Plot a confusion matrix for Churn prediction, and Mean Error for predicting tenure using all three approaches.
 - Write a report summarizing your findings, including the best-performing model/models and the factors that contributed to its success. Also, comment on if and when there are benefits of training models in a multi-task fashion. If joint training improved the performance, explain why; if not, explain why not.

Deliverables:

- Jupyter notebook containing your code and results. (**Retain the outputs in the notebooks and give proper comments/explanations/interpretation for the code**)
- A report summarizing your findings with a section on each grading component as mentioned below:

Grading Components – 150 points:

- Exploratory data analysis: 10 points
- Preprocessing: 10 points
- Model architecture: 20 points
- Training and evaluation: 80 points
 - Approach A: Separate model for predicting tenure. (20 points)
 - Approach B: Separate model for predicting churn (classification) (20 points)
 - Approach C: Joint model that predicts both tenure and churn simultaneously using shared layers and task-specific output heads. (40 points)
- Experimentation with different hyperparameters and architectures: 20 points
- Summary: 10 points

Part 3: Spam Classification using Recurrent Neural Networks

Objective: The objective of this assignment is to develop an RNN model for spam classification. You will use a dataset containing SMS messages labeled as spam or ham (not spam). You will use this data to train an RNN model to classify messages as spam or not spam.

Tasks:

- Download the spam classification dataset from Kaggle.
<https://www.kaggle.com/datasets/team-ai/spam-text-message-classification>

2. Load and pre-process the data by converting the message into sequences of tokens, padding the sequences to a fixed length, and splitting the data into training and testing sets.
3. Build an RNN model using libraries such as Keras or PyTorch to classify messages as spam or ham.
4. Train the model using the training set and evaluate the model using the testing set.
5. Experiment with different hyperparameters and architectures to improve the performance of the model.
6. Write a report summarizing your findings, including the best performing model and the factors that contributed to its success.

Deliverables:

1. Jupyter notebook containing your code and results. (**Retain the outputs in the notebooks and give proper comments/explanations/interpretation for the code**)
2. A report summarizing your findings.

Grading Components – 100 points:

1. Data pre-processing: 10 points
2. Model architecture: 20 points
3. Training and evaluation: 30 points
4. Experimentation with different hyperparameters and architectures: 30 points
5. Summary: 10 points

Part 4: Image Classification using Transfer Learning

Objective: The objective of this assignment is to develop an image classification model using transfer learning. You will find a pretrain a neural network model, and then use **transfer learning** to classify logos of popular food chains. **Note that this exercise could require many hours of training, so start early.**

Tasks:

1. Download the dataset from: <https://www.kaggle.com/datasets/kmkarakaya/logos-bk-kfc-mcdonald-starbucks-subway-none>
2. Preprocess the data by resizing the images and splitting them into training (80%) and validation sets (20%).
3. Load a pre-trained model, such as VGG-16 or ResNet, using Keras or PyTorch. (or a library of your choice)
4. Replace the last fully connected layer of the pre-trained model with a new fully connected layer with the appropriate number of output nodes for the new dataset.

5. Freeze the weights of the pre-trained layers and train only the new fully connected layer using the training set.
6. Evaluate the performance of the model using the validation set.
7. Fine-tune the entire model by unfreezing some of the pre-trained layers and training them along with the new fully connected layer using the new dataset
8. Again, evaluate the performance of the model using the validation set.
9. Write a report summarizing your findings, including the best performing model and the factors that contributed to its success.

Deliverables:

1. Jupyter notebook containing your code and results. (**Retain the outputs in the notebooks and give proper comments/explanations/interpretation for the code**)
2. A report summarizing your findings with a section on each grading component as mentioned below:

Grading Components – 100 points:

1. Data preprocessing: 15 points
2. Pre-trained model selection: 15 points
3. Model architecture modification: 30 points
4. Training and evaluation of the new model and fine-tuning: 30 points
5. Summary: 10 points

General Instructions:

1. This is an Individual Assignment.
2. **Do NOT submit .zip files** otherwise the submission will not be considered.
3. Please note that both Report (PDF file) and Code files (.ipynb) are mandatory.
4. **Any late submission** will attract a penalty as mentioned in the course outline.
5. Not following assignment instructions will attract penalties.
6. The honor code for this submission is **2N-b. Please look through the honor code restrictions carefully before attempting the assignment as there will be strong consequences for breaking them.**
7. Upload your submissions to the '**DL Assignment Submission**' folder on LMS.
8. Handwritten content (except for backpropagation derivation) will not be considered for evaluation.
9. Assignment Submission forms should be submitted separately.
10. Name the report as **Name_PGiD.pdf**.
11. **Email submissions are not allowed.** All the submissions must be made on the LMS.

Assignment Deliverables:

- **4 Python code files (.ipynb)**, one for each part of the assignment. (**Retain the outputs in the notebooks and give proper comments/explanations/interpretation for the code**)
- A report (pdf) containing the summary of your findings including information about the best performing model, the models tried, and their performance for all four questions. (**Note: Do not submit pdf of (.ipynb) jupyter notebook as a report**)
- The Assignment Submission Form (available in the assignment submission folder on LMS). Submit this as a separate file, not as part of the report. Please note that submissions not containing the Assignment Submission form will not be considered for evaluation.

Deadline: 6th Feb 2026,11:55PM