

Instructor: Alina Vereshchaka

Assignment 1

CNN, RNN and LSTM Architectures

Checkpoint: Feb 27, Tue, 11:59 pm

Due date: Mar 7, Thu 11:59pm

Welcome to Assignment 1! This assignment focuses on developing both theoretical and practical skills related to convolutional neural networks (CNNs), recurrent neural networks (RNNs), and Long Short-Term Memory (LSTM) networks. It consists of four hands-on exercises where you will derive and analyze network setups, work with various datasets, and implement, train, and adjust different neural networks. There is also a theoretical component covering CNNs and LSTMs.

- You can use inbuilt functions or define some functions from scratch.
- All libraries are allowed, except those with pre-trained models or predefined architectures. Submissions with pre-trained models or predefined architectures will not be considered.

Part I: CNN Classification [20 points]

In this part implement VGG. It is one of the commonly used CNN architectures. It has different versions and, in this part, we implement the VGG-13 (Version B) and apply it to solve an image dataset containing three classes: dogs, cars and food. The expected accuracy for this part is more than 75% without optimization techniques and 80% with optimization techniques.

CNN Dataset



CNN dataset consists of 10,000 examples for each category, thus in total 30,000 samples. Each example is a 64x64 image. The dataset can be found on UBLearn.

CSE 676-B: Deep Learning, Spring 2024

STEPS:

1. Data preprocessing:
 - a. Read, preprocess, and print the main statistics about the dataset
 - b. Using any data visualization library (e.g. [matplotlib](#), [seaborn](#), [plotly](#)), provide at least 2 visualization graphs related to your dataset.
 - c. Prepare the dataset for training, e.g. normalizing, converting target to categorical values and splitting it into training, testing and validation sets.

2. Implement the VGG-13 (Version B) architecture following the proposed architecture. Be sure to use the same kernel sizes, stride, and padding as specified in the paper. You can refer to the [original VGG paper](#) for more details.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

3. Train the model on a dataset provided at UBLEarns. For your dataset, adjust the size, e.g. for the input and the output layers.
4. Apply techniques to prevent overfitting and improve the results. Discuss each of the techniques you have used and how they impact the performance:
 - a. Regularization: Apply regularization techniques, such as L1 or L2 regularization, to the model's parameters.
 - b. [Dropout](#): Introduce dropout layers between the fully connected layers.
 - c. [Early stopping](#): Monitor the performance of the model on a validation set and stop training when the validation loss stops improving.
 - d. [Image augmentation](#)
6. Save the weights of the trained network that provides the best results. [Saving and loading models \(PyTorch\)](#)
7. Discuss the results and provide relevant graphs:
 - Report training accuracy, training loss, validation accuracy, validation loss, testing accuracy, and testing loss.
 - Plot the training and validation accuracy over time (epochs).
 - Plot the training and validation loss over time (epochs).

CSE 676-B: Deep Learning, Spring 2024

- Generate a confusion matrix using the model's predictions on the test set.
- Calculate and report other evaluation metrics such as Precision, recall and F1 score ([more details](#)). You can use [sklearn.metrics.precision_recall_fscore_support](#)

Notes:

- Works with in-build or pretrained models won't be considered for evaluation.
- You can partially reuse related implementations that you have completed for other courses with a proper citation, e.g. "Part I, Step 1 is based on the CSE 676 Deep Learning Assignment 0 submission by My Full Name"

In your report for Part I:

1. Describe the CNN model you have defined.
2. Describe how the techniques (regularization, dropout, early stopping) have impacted the performance of the model.
3. Discuss the results and provide relevant graphs:
 - a. Report training accuracy, training loss, validation accuracy, validation loss, testing accuracy, and testing loss.
 - b. Plot the training and validation accuracy over time (epochs).
 - c. Plot the training and validation loss over time (epochs).
 - d. Generate a confusion matrix using the model's predictions on the test set.
 - e. Report any other evaluation metrics used to analyze the model's performance on the test set.

Part II: Implementing ResNet Architecture [20 points]

In this part implement ResNet-18 by manually defining a residual block and apply it to solve an image dataset that we use in Part I.

The expected accuracy for this part is more than 75% without optimization techniques and 80% with optimization techniques.

STEPS:

1. Implement residual blocks of ResNet, including convolutional layers, batch normalization, ReLU activation, and residual connections. You can use `nn.Conv2d`, `nn.BatchNorm2d`, `nn.Sequential`, `nn.Identity`.
2. Design a Resnet-18 model: configuration with 18-layer [original ResNet paper](#)
3. Train the model on the dataset that you used in Part I. For your dataset, adjust the size, e.g. for the input and the output layers.

CSE 676-B: Deep Learning, Spring 2024

4. Apply at least two techniques to prevent overfitting and improve the results.
8. Save the weights of the trained neural network that provides the best results. [Check saving and loading models \(PyTorch\)](#)
9. Discuss the results and provide relevant graphs:
 - Report training accuracy, training loss, validation accuracy, validation loss, testing accuracy, and testing loss.
 - Plot the training and validation accuracy over time (epochs).
 - Plot the training and validation loss over time (epochs).
 - Generate a confusion matrix using the model's predictions on the test set.
 - Calculate and report other evaluation metrics such as Precision, recall and F1 score ([more details](#)). You can use [sklearn.metrics.precision recall fscore support](#)

ResNet Architectures

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Note: works with in-build or pretrained models won't be considered for evaluation.

In your report for Part II:

1. Describe the ResNet architecture you have defined.
2. Describe how the techniques (regularization, dropout, early stopping) have impacted the performance of the model.
3. Discuss the results and provide relevant graphs:
 - a. Report training accuracy, training loss, validation accuracy, validation loss, testing accuracy, and testing loss.
 - b. Plot the training and validation accuracy over time (epochs).
 - c. Plot the training and validation loss over time (epochs).
 - d. Generate a confusion matrix using the model's predictions on the test set.

- e. Report any other evaluation metrics used to analyze the model's performance on the test set.

Part III: Time-Series Forecasting using RNN [20 points]

In this part, we work on time-series forecasting using RNN and LSTM methods. The expected accuracy for this part is more than 75% for the final model.

DATASETS

Choose ONE dataset from the following real-world time-series datasets:

- [Energy Consumption](#): electricity consumption data collected from multiple households
- [Air Quality Dataset](#): Hourly air quality data with various features like temperature, humidity, and gas concentrations
- [Web Traffic Dataset](#): Time-series data representing the number of visits or page views to websites.
- [Electric Grid Stability Dataset](#): Time-series data representing the stability status of an electrical grid
- [Taxi Trip Duration Dataset](#): Historical data of taxi trip durations with features like pickup/dropoff coordinates, timestamps, and other related information.
- [Retail Sales Dataset](#): sales figures of a retail store or chain over time, typically including features such as product categories, promotions, and seasonality.

STEPS:

1. Select one dataset from the list above.
 - a. Read, preprocess, and print the main statistics about the dataset
 - b. Using any data visualization library (e.g. [matplotlib](#), [seaborn](#), [plotly](#)), provide at least 3 visualization graphs related to your dataset.
 - c. Prepare the dataset for training, e.g. normalizing, converting features to categorical values and splitting it into training, testing and validation sets.
2. Building RNN:
 - a. Build the RNN model architecture, that consists of at least three RNN layers.
 - b. Train RNN model and try various setups and hyperparameters tuning to obtain an expected accuracy.
10. Save the weights of the trained neural network that provides the best results. [Check saving and loading models \(PyTorch\)](#)
11. Discuss the results and provide relevant graphs:

CSE 676-B: Deep Learning, Spring 2024

- Report training accuracy, training loss, validation accuracy, validation loss, testing accuracy, and testing loss.
- Plot the training and validation accuracy over time (epochs).
- Plot the training and validation loss over time (epochs).
- Generate a confusion matrix using the model's predictions on the test set.
- Calculate and report other evaluation metrics such as Precision, recall and F1 score ([more details](#)). You can use [sklearn.metrics.precision recall fscore support](#)

In your report for Part III:

1. Provide brief details about the nature of your dataset. What is it about? What type of data are we encountering? How many entries and variables does the dataset comprise?
2. Provide the details related to your RNN architecture.
3. Discuss the results and provide relevant graphs:
 - a. Report training accuracy, training loss, validation accuracy, validation loss, testing accuracy, and testing loss.
 - b. Plot the training and validation accuracy over time (epochs).
 - c. Plot the training and validation loss over time (epochs).
 - d. Generate a confusion matrix using the model's predictions on the test set.
 - e. Report any other evaluation metrics used to analyze the model's performance on the test set.

Part IV: Sentiment analysis using LSTM [20 points]

In this part, we perform a sentiment analysis using LSTM model. The expected accuracy for this part is more than 75% for the final model.

DATASETS

Choose ONE dataset from the following options for sentiment analysis:

- [Amazon Product Reviews Dataset](#): reviews of various products sold on Amazon, labeled with sentiment.
- [Yelp Reviews Dataset](#): reviews from the Yelp platform, labeled with sentiment
- [Twitter Airline Sentiment Dataset](#): tweets about airline experiences, labeled with sentiment.
- [Stanford Large Movie Review Dataset](#) movie reviews classified as positive or negative
- [Medical Dialogue Dataset](#): contains conversations (in English) between doctors and patients. It has 0.26 million dialogues
- [News Headlines Dataset](#): news headlines labeled with sentiment, covering various news topics and categories.

CSE 676-B: Deep Learning, Spring 2024

STEPS:

1. Select one dataset from the list above.
 - a. Read, preprocess, and print the main statistics about the dataset
 - b. Using any data visualization library (e.g. [matplotlib](#), [seaborn](#), [plotly](#)), provide at least 3 visualization graphs related to your dataset.
 - c. Prepare the dataset for training, e.g. tokenizing the text and converting it into numerical sequences. Splitting it into training, testing and validation sets.
2. Build LSTM:
 - a. Build the LSTM model architecture, that consists of at least three LSTM layers.
 - b. Train LSTM model and try various setups and hyperparameters tuning. Save the weights of the trained model, that returns best results.
 - c. Report accuracy and loss for your network
 - d. Visualize and analyze the results
3. Build an improved LSTM:
 - a. Use any other version of LSTM model architecture to improve the results. E.g. Gated Recurrent Unit (GRU), Bidirectional LSTM, Stacked LSTM
 - b. Train the improved LSTM model and try various setups and hyperparameters tuning. Save the weights of the trained model, that returns best results.
 - c. Report accuracy and loss for your network
 - d. Visualize and analyze the results

In your report for Part IV:

1. Provide brief details about the nature of your dataset. What is it about? What type of data are we encountering? How many entries and variables does the dataset comprise?
2. Provide the details related to your LSTM and Improved LSTM architectures.
3. Discuss the results and provide the graphs for both models. Compare the performance of both Improved LSTM and LSTM models applied to the same dataset.
4. Discuss the strengths and limitations of using recurrent neural models for sentiment analysis.

Part V: CNN & LSTM Theoretical Part [20 points]

Part V.I: CNN [10 points]

In a convolutional neural network (CNN), suppose you're working on a computer vision project that involves classifying images of different emotions expressed by human faces. The network is designed for a multi-class classification task with 5 output classes, where each class represents a different emotion category: happiness, sadness, anger, surprise, and neutral.

CNN Architecture:

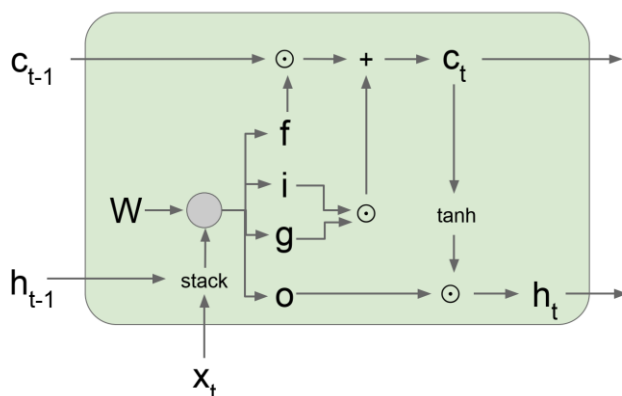
- The input consists of 32x28 RGB images. The first layer of the CNN is a convolutional layer with 10 filters, each having a size of 5x5, zero padding, and a stride of 1.
- The output values of the network represent the predicted probabilities for each emotion category and the predicted probabilities for each class are non-negative and sum up to 1.

TASKS:

1. What is the output size after the first layer?
2. How many parameters are there in first layer?
3. What would be the output size if a padding of 1 was used instead of zero padding?
4. What would be the number of parameters if the input images were grayscale instead of RGB?
5. Considering the above specific task and the requirement of probabilistic outputs, which activation function would be most suitable for the output layer in this scenario?
6. Prove that the activation function used here is invariant to constant shifts in the input values, meaning that adding a constant value to all the input values will not change the resulting probabilities.

Part V.II: LSTM Derivation [10 points]

Consider the following LSTM structure:



i input gate: whether to write to cell

f forget gate: whether to erase cell

o output gate: how much to reveal cell

g gate gate: how much to write to cell

CSE 676-B: Deep Learning, Spring 2024

The definitions are provided as follows:

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} \left[\begin{pmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ \mathbf{W}_3 \\ \mathbf{W}_4 \end{pmatrix} \begin{pmatrix} \mathbf{h}_{t-1} \\ \mathbf{x}_t \end{pmatrix} \right]$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t)$$

where:

- σ denotes the sigmoid activation function
- $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \mathbf{W}_4$ represent weight matrices
- \mathbf{h}_{t-1} represents the hidden state at previous time step
- \mathbf{x}_t represents the current input.

TASK:

Consider \mathcal{L} is the loss of an LSTM model. Derive the gradient formulars for the following partial derivatives:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{i}_t}, \quad \frac{\partial \mathcal{L}}{\partial \mathbf{f}_t}, \quad \frac{\partial \mathcal{L}}{\partial \mathbf{o}_t}, \quad \frac{\partial \mathcal{L}}{\partial \mathbf{g}_t}, \quad \frac{\partial \mathcal{L}}{\partial \mathbf{c}_t}, \quad \frac{\partial \mathcal{L}}{\partial \mathbf{h}_t}$$

In your report for Theoretical Part (Part V):

You may provide the answer using handwritten notes, typed in the MS Word or Latex. The submitted version has to be converted to a single pdf and named as avereshc_assignment1_checkpoint.zip

Bonus points

Transfer Learning [5 points]

Use pre-trained models in PyTorch and apply them to the [Food-11 dataset](#).

TASK:

- Work with at least three different pre-trained models, e.g. ResNet, VGG, or DenseNet. [Check PyTorch documentation for more details](#)
- Perform parameters tuning
- Evaluate the model's performance and compare the results for applying different pre-trained models

Bonus part submission:

- Create a separate zip folder named as UBIT

CSE 676-B: Deep Learning, Spring 2024

TEAMMATE1_TEAMMATE2_assignment1_bonus.zip

e.g., avereshc_ankithba _ assignment1_bonus.

- Include all the files needed in the folder.
- Report is not required, you can include all the comments as part of your Jupyter Notebook/Script.

ASSIGNMENT STEPS

1. Submit checkpoint (February 27)

- Complete Part I, partially Part II (e.g. Steps 1-3) and Part V.I (CNN) of the assignment
- For the checkpoint, it is ok if your report is not fully completed. You can finalize it for the final submission.
- Include all the references at the end of the report. There is no minimum requirement for the report size, just make sure it includes all of the information required.
- Do not include a CNN dataset as part of your submission
- Add all your assignment files in a zip folder including .ipynb files, draft report as pdf, part V.2 as a pdf and saved weights at a zip folder. Name zip folder with all the files as: UBIT TEAMMATE1_TEAMMATE2_assignment1_checkpoint.zip (e.g., avereshc_ atharvap_assignment1_checkpoint.zip).
- Submit at UBLearns > Assignments

2. Submit final results (March 7)

- Fully complete all parts of the assignment and submit to UBLearns > Assignments
- In your report or Jupyter notebook include all the references that were used to complete the assignment or the task. You can include that at the end of the report.
- Your Jupyter notebook should be saved with the results. If you are submitting python scripts, after extracting the ZIP file and executing command python main.py in the first level directory, all the generated results and plots you used in your report should appear printed out in a clear manner.
- **Add .txt file “dataset_assignment1_part1.txt” and include a link that points to the dataset you have selected. Do not submit the dataset itself.**
- Report (as a pdf file): combine the reports for all parts into one pdf file named as UBIT TEAMMATE1_TEAMMATE2_assignment1_report.pdf
e.g., avereshc_atharvap_assignment1_report.pdf
- Theoretical Part (Part V) can be combined with a final report or submitted as a separate file named as

CSE 676-B: Deep Learning, Spring 2024

UBIT TEAMMATE1_TEAMMATE2_assignment1_part_5.pdf
e.g., avereshc_atharvap_assignment1_part_5.pdf

- Code (as ipynb file with saved outputs): separate file for Part I, II and Part III, IV named as
UBIT TEAMMATE1_TEAMMATE2_assignment1_part_1_2.ipynb
UBIT TEAMMATE1_TEAMMATE2_assignment1_part_3_4.ipynb
- Pickle or .h5 files with saved models that generate the best results for your model for each parts named as
UBIT TEAMMATE1_TEAMMATE2_assignment1_part_#.h5
e.g., avereshc_atharvap_assignment1_part_1.h5
- For Bonus part, create a separate folder named as UBIT TEAMMATE1_TEAMMATE2_assignment1_bonus.zip
- All assignment files and bonus folder should be packed in one ZIP file named: UBIT TEAMMATE1_TEAMMATE2_assignment1.zip
e.g. avereshc_atharvap_assignment1.zip
- Submit your ZIP file at UBlearns > Assignments
- Verify that the submission was successful, e.g., download the submitted files and verify they are all correct
- You can make unlimited number of submissions and only the latest will be evaluated

Notes:

- Ensure that your code follows a clear structure and contains comments for the main functions and specific attributes related to your solution. You can submit multiple files, but they all need to be labeled with a clear name.
- Recheck the submitted files, e.g. download and open them, once submitted and verify that they open correctly

Assignments Grading

Checkpoint submission:

- Graded based on the 0/1 grade.
 - “1” is assigned, if more than 80% of the work has been fully completed for the checkpoint-related parts. “0” is assigned for all other cases
- Obtaining “0” for a checkpoint submission results in a fixed 30 points loss from the final assignment evaluation.

CSE 676-B: Deep Learning, Spring 2024

Final submission:

- Graded based on the X out of 100 points + bonus [if applicable]
- During the final evaluation, all the parts are evaluated, so please include a final version of all the parts of the assignment in your final submission.

Notes:

- Only files submitted on UBLearn are considered for evaluation.
- Files from local device/GitHub/Google colab/Google docs/other places are not considered for evaluation
- We strongly recommend submitting your work in-progress on UBLearn and always recheck the submitted files, e.g. download and open them, once submitted

Academic Integrity

The standing policy of the Department is that all students involved in any academic integrity violation (e.g., plagiarism in any way, shape, or form) will receive an F grade for the course. The catalog describes plagiarism as “Copying or receiving material from any source and submitting that material as one’s own, without acknowledging and citing the particular debts to the source, or in any other manner representing the work of another as one’s own.”. Refer to the [Office of Academic Integrity](#) for more details.

Important Information

This assignment can be done individually or in a team of up to two students. The grading rubrics are the same for a team of any size.

- No collaboration, cheating, and plagiarism is allowed in assignments, quizzes, the final exam or final project.
- All the submissions will be checked using SafeAssign as well as other tools. SafeAssign is based on the submitted works for the past semesters as well the current submissions. We can see all the sources, so you don't need to worry if there is a high similarity with your Checkpoint submission.
- The submissions should include all the references. Kindly note that referencing the source does not mean you can copy/paste it fully and submit as your original work. Updating the hyperparameters or modifying the existing code is a subject to plagiarism. Your work has to be original. If you have any doubts, send a private post on piazza to confirm.
- All group members and parties involved in any suspicious cases will be officially reported using the Academic Dishonesty Report form. What does that mean?

CSE 676-B: Deep Learning, Spring 2024

- In most cases, the grade for the assignment/quiz/final project/midterm will be 0 and all bonus points will be subject to removal from the final evaluation for all students involved.
- Final grade reduction (e.g. if you obtain B, that will result in B-)
- Those found violating academic integrity more than once throughout their program will receive an immediate F in the course.

Please refer to the [Academic Integrity Policy](#) for more details.

- The report should be delivered as a separate pdf file. You can combine report for Part I, Part II, Part III, Part IV and Part V into the same pdf file. You can follow the [NIPS template](#) as a report structure. You may include comments in the Jupyter Notebook; however, you will need to duplicate the results in a separate pdf file.
- All the references can be listed at the end of the report. There is no minimum requirement for the report size, just make sure it includes all the information required.
- For the Bonus part, no report is needed.

Late Days Policy

You can use up to 5 late days throughout the course that can be applied to any assignment-related due dates. You do not have to inform the instructor, as the late submission will be tracked in UBlerns.

Important Dates

February 27, Tuesday, 11:59 pm - Checkpoint Submission is Due

March 7, Thu, 11:59 pm - Final Submission is Due