Hong Kong University of Science and Technology
COMP 4211: Machine Learning
Spring 2021

Programming Assignment 3
Due: 13 April 2021, Tuesday, 11:59pm

1 Objective

The objective of this assignment is to practise the use of the `PyTorch` machine learning framework through building recurrent neural network (RNN) models to solve the fake news detection problem. Throughout the assignment, students will be guided to process the text dataset, develop an RNN model, and apply different performance improvement techniques to give multiple variants of the model.

2 Major Tasks

The assignment consists of a coding part and a written report:

CODING: Build an end-to-end RNN model and multiple variants using `PyTorch` to detect fake news.

WRITTEN REPORT: Report the results and answer some questions.

The tasks will be elaborated in Section 4. Note that [Qn] refers to a specific question (the nth question) that you need to answer in the written report.

3 Setup

- Make sure that the libraries `NLTK`, `torch`, `torchtext`, `matplotlib` and `sklearn` have been installed on your machine.
- For this assignment, it suffices to use only `PyTorch` with `Python 3.7` or above. Other machine learning frameworks including `TensorFlow` and `Keras` should not be used.
- You are highly recommended to use GPU resources like those provided by the GPU servers of the Department of Computer Science and Engineering (`https://cssystem.cse.ust.hk/Facilities/ug_cluster/gpu.html`) or the Google Colab (`https://colab.research.google.com`).
- The dataset files for this assignment are provided as a ZIP file (`pa3.zip`).

4 Fake News Detection

The prevalence of fake news has increased with the growing popularity of online social media platforms such as Facebook, Twitter, and YouTube. It is now becoming a real threat due to the ease of creating, diffusing, and consuming content online. Therefore, it is imperative to develop methods to distinguish fake news from real news. One promising approach is machine learning. Since the textual information of the news is sequential in nature, it is common to use RNNs for text data. As processing the entire content of news articles by an RNN is computationally expensive, in this assignment, you will build a binary RNN classifier to detect fake news based on the news titles only.

4.1 Binary Text Classification

From Tutorial 7, you have learned to use an RNN model to predict the stock price. For text classification, the workflow is also similar. The RNN model takes a sentence (as a sequence of words) as input and exploits the sequential relationships between words to predict the label.[1]

4.2 Dataset

The Fake News dataset can be found in the `./pa3` directory. For all csv files, each row represents one piece of news. In `train.csv`, there is one **label** column and one **text** column. The label column indicates the target labels of the news **(0: Real; 1: Fake)** and the text column contains their titles.

In `test.csv`, all the labels are set to **2**, meaning **Unknown**. You are required to predict and submit the predicted labels of the news in `test.csv`.

Table 1: Description of the dataset

File	# of instances	Has label?	Purpose
`train.csv`	12,375	yes	training
`test.csv`	2,000	no	prediction

4.3 Tasks

This section will guide you to build an RNN model. Please create a new `Jupyter` notebook for this part.

4.3.1 Dataset and Dataloader

First, you are required to use the `train_test_split` in `scikit-learn` to split the `train.csv` data into a training set and a validation set. The validation set should contain 2,000 examples

[1] You may refer to the tutorial notes on some related concepts and techniques for dealing with textual data, such as tokenization, vocabulary, word embedding, etc.

(1,000 for each class) for *holdout validation*. You should set `random_state` to be 4211 for reproducibility.

The next step is to build a `TabularDataset` class. A `TabularDataset` object loads the `train` dataset and `valid` dataset with properly defined fields for the label data and the text data. The vocabulary should only include the words with a frequency of at least 2. (Hint: the vocabulary used for training, validation, and prediction should only be built using the training data.)

[Q1] Report the number of out-of-vocabulary (OOV) words[2] in your training set and validation set, respectively. How does the `torchtext` library tackle the OOV words?

4.3.2 RNN Baseline Model

An RNN model takes a sequence of words as input and produces the class probabilities as output. Here, you first build a baseline RNN model with the following settings:

Table 2: Configuration of the baseline model

Hyperparameter	Value	Remarks
# embedding layers	1	embedding dimensionality = 50
# RNN layers	1	RNN cell = `nn.RNN` hidden layer dimensionality = 64
# dropout layers	1	$p = 0.1$
# fully connected layers	1	

[Q2] "Print" the model architecture and the number of trainable parameters for the *baseline model* and include them in the written report.

4.3.3 Training and Validation

Since this is a binary classification task, you may use `BCELoss` or `BCEWithLogitsLoss` provided by `PyTorch`. You should classify the example to real news if the predicted probability is at least 0.5, and to fake news otherwise. For optimization, you should use the default setting of the `Adam` optimizer with a `batch size` of 64. During model training, you are required to plot the training and validation losses of each epoch for 15 epochs in one plot using `matplotlib` (this setting is also applicable to the subsequent plotting of all the loss curves for this assignment).[3] You may reuse the model training results in the early questions for the plotting and model comparison in the later questions. It is useful for you to save the best model checkpoint (the model checkpoint with the highest validation accuracy) for later parts.

[Q3] Paste the screenshot of the plot and report the best validation accuracy of the *baseline model*.

[2] The number of unique words not included in your vocabulary built from the training data.

[3] When there are multiple curves in a plot, you should include a legend to help identify the curves for each model.

4.3.4 Empirical Study

Different RNN Settings

In this section, you need to build three models in addition to the baseline model. These three models are variants of the baseline model and are used to compare the performance of different RNN settings. You should only change the settings indicated in Table 3 and keep the other settings the same as the baseline model.

Table 3: Configurations of models 1-3

	RNN cell	Bidirectional?
Model 1	`nn.GRU`	no
Model 2	`nn.LSTM`	no
Model 3	`nn.LSTM`	yes

Model Description: *Model 1* and *model 2* replace the baseline vanilla RNN cell with GRU and LSTM cells, respectively. *Model 3* uses bidirectional LSTM. 'Bidirectional' here means that the sequential relationship of the sequences is modeled in both the forward and backward directions. This can be set in the parameters of `nn.LSTM`.

[Q4] List two advantages of using GRU and LSTM when compared to the baseline RNN model.

[Q5] Plot the training and validation losses of the *baseline model*, *model 1* and *model 2* in one plot and report the best validation accuracy of each model. Does the result follow your expectation in Q4? If not, suggest some possible reason(s).

[Q6] Plot the training and validation losses of *model 2* and *model 3* in one plot and report the best validation accuracy of each model. Briefly explain the effect of adding the backward direction in the bidirectional LSTM compared to the standard LSTM. (Hint: you can discuss the training and validation losses, convergence rate, and the gap between losses in your analysis.)

4.3.5 Improving Model Training

In this part, you will explore some common techniques to improve your model performance. **You are required to implement two of the four categories (A-D).** (If you have implemented more than two categories, only the two with the highest marks will be counted towards the final score.) Except for the settings mentioned in the following questions, all other model configurations and training settings should be the same as those for *model 2*. For the loss curve plotting and validation accuracy comparison in this part, you may reuse the results and curves of *model 2* in Q5. For the code implementation, you may duplicate your previous code and modify it from there. Alternatively, you may also modify your previous code to accept new arguments for different configurations. Make sure that your changes are clearly indicated by appropriate comments in the code.

A. Pre-trained Word Embedding

Pre-trained word embeddings are the trained weights of the embedding layer from large datasets. They can capture the semantic and syntactic information of words, and be used as the initial weights of the embedding layer for solving other tasks. In this task, you are required to use the **GloVe** pre-trained word embedding for your model. Specifically, you should load the `glove.6B.50d` word embedding in `torchtext` and implement two models, one with a fixed pre-trained **GloVe** embedding (*model 4*) and the other with a **GloVe** embedding fine-tuned for this dataset (*model 5*). To verify the effectiveness of the word embedding, you should also implement a model with random vectors as word embedding (*model 6*) (i.e., a randomly initialized non-trainable embedding). A summary of the three models is shown in Table 4.

Table 4: Configurations of models 4-6

	Embedding type	Freeze?
Model 4	`GloVe`	yes
Model 5	`GloVe`	no
Model 6	`random vector`	yes

[Q7a] Report the L2 distances of the pre-trained **GloVe** embedding for the three word pairs below. What can you observe?

- ‘happy’ and ‘good’
- ‘france’ and ‘germany’
- ‘france’ and ‘happy’

[Q8a] Before training, among *model 2*, *model 4*, *model 5*, and *model 6*, which one do you expect to have the best performance and which one has the worst? Why?

[Q9a] Plot the training and validation losses of *model 2*, *model 4*, *model 5*, and *model 6* in one plot and report the best validation accuracy of each model. Does the result follow your expectation? If not, suggest some possible reason(s).

B. Model Weight Initialization

In this task, you will implement two weight initialization methods, **Xavier** and **Kaiming**. Both methods should use the normal distribution.[4]

Table 5: Configurations of models 7-8

	Initialization method
Model 7	`Xavier`
Model 8	`Kaiming`

[Q7b] What is weight initialization? What potential problem is it trying to prevent from happening? Explain it briefly.

[4] You may refer to `https://pytorch.org/docs/stable/nn.init.html` for the detailed implementation.

[Q8b] Plot the training and validation losses of *model 2*, *model 7*, and *model 8* in one plot and report the best validation accuracy of each model.

[Q9b] From the above results, does weight initialization affect model training in terms of convergence rate and model performance in terms of validation accuracy? If yes, describe the difference(s). If no, suggest some possible reason(s).

C. Imbalanced Dataset

In this dataset, the number of real news far exceeds the number of fake news. Such imbalance is common for real-world datasets as the data from some categories are easier to be collected. In this task, you will tackle this issue by oversampling, undersampling, and using a weighted loss (Table 6). Oversampling randomly duplicates examples from the minority class until the dataset is balanced. Undersampling randomly deletes examples from the majority class until the dataset is balanced. As for the weighted loss approach, it applies different weights to different classes when computing the loss.[5]

Table 6: Configurations of models 9-11

	Strategy
Model 9	`oversampling`
Model 10	`undersampling`
Model 11	`weighted loss`

[Q7c] How do you select the value of the weight for each class for the weighted loss approach? Report the values of the weights.

[Q8c] Plot the training and validation losses of *model 2*, *model 9*, *model 10*, and *model 11* in one plot and report the best validation accuracy of each model.

[Q9c] Which model has the highest validation accuracy? Which has the lowest validation accuracy? Suggest some possible reason(s).

D. Model Ensemble

Model ensemble is a method of combining a diverse set of learners (individual models) together to improve the stability and predictive power of the model. In this task, you will implement two common ensemble methods, average ensemble and weighted average ensemble. To be specific, the average ensemble averages the predicted probabilities of the three best models (in terms of validation accuracy) from your previous implemented models to make a final prediction (Figure 1), while the weighted average ensemble assigns specific weights to the models.

[Q7d] Why does the model ensemble improve the performance of the model?

[Q8d] Propose a method to find suitable weights for the models in the weighted average ensemble.

[5] Given a set of examples from class A and class B, the losses calculated for examples from the two classes are $\mathcal{L}_A$ and $\mathcal{L}_B$, respectively. The weighted loss method applies weights w_A and w_B to the two classes such that the total loss is $w_A\mathcal{L}_A + w_B\mathcal{L}_B$. For the implementation details, please refer to the `weight` parameter of the loss function in the `PyTorch` documentation.

Table 7: Description of model 12-16

	Description
Model 12	`best model from previous parts`
Model 13	`2nd best model from previous parts`
Model 14	`3rd best model from previous parts`
Model 15	`average ensemble on models 12, 13, 14`
Model 16	`weighted average ensemble on models 12, 13, 14`

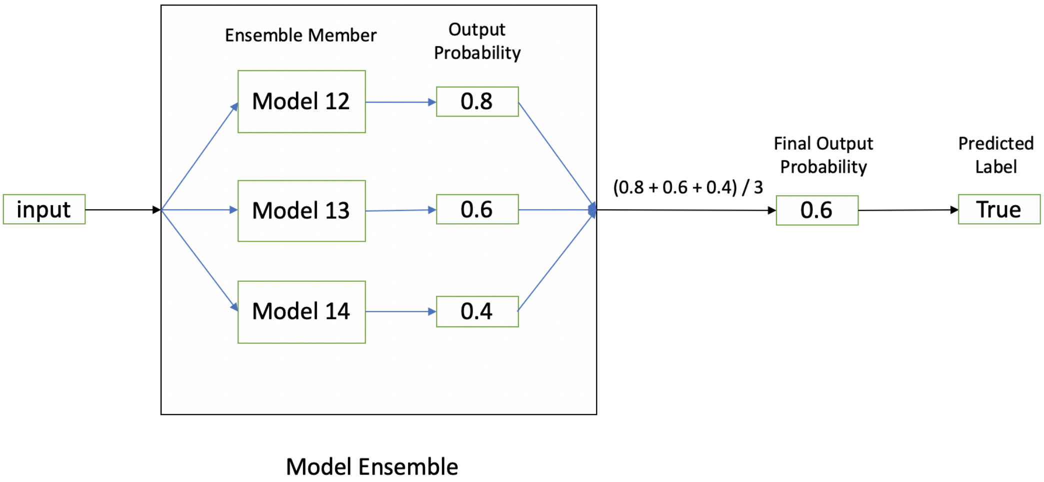

Figure 1: Simple illustration of average ensemble for three models

Explain your method.

[Q9d] Compare the validation accuracy of these five models (three chosen models and two ensemble models). Do the ensemble methods improve the model performance (in terms of validation accuracy)? If yes, describe the difference(s). If no, suggest some possible reason(s).

4.3.6 Prediction

In the prediction stage, choose the model (possibly a model ensemble) with the highest validation accuracy and predict the labels of the news in `test.csv`. (You can simply load the saved model checkpoint. You do not need to retrain the model on both the training and validation sets.) Save the predictions as a csv file. The score for this task is based on your prediction accuracy.

[P1] Submit your predictions as `pred.csv`. The file should consist of two columns, where the first column is the predicted label and the second column is the news text. (The row order should be the same as that in `test.csv`.)

5 Written Report

Answer [Q1] to [Q9] in one single written report.

6 Some Programming Tips

As is always the case, good programming practices should be applied when coding your program. Below are some common ones but they are by no means complete:

- Using a small subset of data to test the code
- Using checkpoints to save partially trained models
- Using functions to structure program clearly
- Using meaningful variable and function names to improve readability
- Using consistent styles
- Including concise but informative comments

7 Assignment Submission

Assignment submission should only be done electronically using the Course Assignment Submission System (CASS):

`https://cssystem.cse.ust.hk/UGuides/cass/student.html`

There should be three files in your submission with the following naming convention required:

1. **Report** with filename `<StudentID>_report.pdf`: in PDF format.

2. **Prediction** with filename `<StudentID>_pred.csv`: in csv format.

3. **Source code and a README file** with filename `<StudentID>_code`: compressed into a single ZIP or RAR file.

Note: The source code files should contain all necessary code for running your program as well as a brief user guide for the TA to run the programs easily to verify your results, all compressed into a single ZIP or RAR file. The data should not be submitted to keep the file size small. When multiple versions with the same filename are submitted, only the latest version according to the timestamp will be used for grading. Files not adhering to the naming convention above will be ignored.

8 Grading Scheme

This programming assignment will be counted towards 15% of your final course grade. Note that the plus sign (+) in the Report and Prediction columns of the table below indicates that reporting without providing the corresponding code will get zero point. The maximum scores for different tasks are shown below:

Table 8: [C]: code, [Q]: Written report, [P]: Prediction

Grading Scheme	Code (43)	Report (52)	Prediction (5)
Dataset and Dataloader (10)			
- [C1] Dataset Splitting	2		
- [C2] Dataset Class	3		
- [C3] OOV + [Q1]	2	+3	
RNN Model (8)			
- [C4] Baseline Model + [Q2]	4	+4	
Training and Validation (13)			
- [C5] Loss Function	2		
- [C6] Training Loop	4		
- [C7] Validation Loop	2		
- [C8] Loss Curve + [Q3]	2	+3	
Empirical Study (20)			
- [Q4]		2	
- [C9] Model 1-2 + [Q5]	4	+7	
- [C10] Model 3 + [Q6]	2	+5	
Improving Model (40)			
Pre-trained Word Embedding			
- [C11] L2 Calculation + [Q7a]	2	+2	
- [Q8a]		4	
- [C12] Model 4-6 + [Q9a]	6	+6	
Weight Initialization			
- [Q7b]		4	
- [C13] Model 7-8 + [Q8b]	6	+6	
- [Q9b]		4	
Imbalanced Dataset			
- [Q7c]		4	
- [C14] Model 9-11 + [Q8c]	6	+6	
- [Q9c]		4	
Model Ensemble			
- [Q7d]		4	
- [Q8d]		4	
- [C15] Model 12-16 + [Q9d]	6	+6	
Prediction (9)			
- [C16] Prediction + [P1]	4		+5

Late submission will be accepted but with penalty. The late penalty is deduction of one point (out of a maximum of 100 points) for every minute late after 11:59pm. Being late for a fraction of a minute is considered a full minute. For example, two points will be deducted if the submission time is 00:00:34. At most one NQA coupon may be used to entitle you to submit this assignment late for one day without grade penalty.

9 Academic Integrity

Please refer to the regulations for student conduct and academic integrity on this webpage: `https://acadreg.ust.hk/generalreg`.

While you may discuss with your classmates on general ideas about the assignment, your submission should be based on your own independent effort. In case you seek help from any person or reference source, you should state it clearly in your submission. Failure to do so is considered plagiarism which will lead to appropriate disciplinary actions.