Assignment 4: Classification of Textual Data

COMP 551, Winter 2025, McGill University
Contact TAs: Benjamin Lo and Valliappan Chidambaram Adaikkappan

Please read this entire document before beginning the assignment.

Preamble

- This assignment is **due on December 5th at 11:59 pm (EST, Montreal Time).** There is a penalty of 2^k percent penalty for k days of delay, which means your grade will be scaled to be out of $100 - 2^k$. No submission will be accepted after 6 days of delay.
- This assignment is to be completed in groups of three. All members of a group will receive the same grade, except when a group member is not responding or contributing to the project. If this is the case and there are major conflicts, please reach out to the group TA for help and flag this in the submitted report. Please note that it is not expected that all team members will contribute equally. However, every team member should make integral contributions to the project, be aware of the content of the submission, and learn the full solution submitted.
- You will submit your assignment on MyCourses as a group. You must register your group on MyCourses, and any group member can submit. See MyCourses for details.
- We recommend to use **Overleaf** for writing your report and **Google colab** for coding and running the experiments. The latter also gives access to the required computational resources. Both platforms enable remote collaborations.
- You should use Python for this assignment. You are free to use libraries with general utilities, such as matplotlib, numpy, and scipy for Python, unless stated otherwise in the description of the task. In particular, in most cases, you should implement the models and evaluation functions yourself, which means you should not use pre-existing implementations of the algorithms or functions as found in SciKit learn, and other packages. The description will specify this on a per-case basis.

Background

In this assignment, you will **implement a Long Short-Term Memory Network (LSTM) from scratch, and fine-tune a transformer model (BERT) starting with pre-trained weights from a package**, and compare these two algorithms on a text-classification dataset. The goal is to gain experience implementing machine learning algorithms from scratch, as well as gain familiarity with modern deep learning libraries by getting hands-on experience comparing their performances on a real-world textual dataset.

Task 1: Acquire and pre-process the Web of Science Dataset

The dataset that will be used for this assignment is the Web of Science dataset, found here: `https://data.mendeley.com/datasets/9rw3vkcfy4/6`. The dataset is subdivided into 3 datasets of various sizes. You are asked to work with **WOS-11967**, although you may use all 3 in your analyses.

In this dataset, each training example is an abstract from a scientific paper, and each training label is the scientific field that the abstract belongs to. The file **YL1.txt** corresponds to the numerical index of the scientific field according to the following set: $\{$Computer Science, Electrical Engineering, Psychology, Mechanical Engineering, Civil Engineering, Medical Science, and Biochemistry$\}$. To be clear, this means that a YL1 value of 0 would correspond to the "Computer Science" label. The goal is to train a model to predict the scientific field given an abstract.

In addition to this, the dataset has a secondary classification task. Aside from belonging to a specific scientific field, each abstract also belongs to a sub-field within that domain (for example, Machine Learning is a sub-category of the Computer Science domain). The labels corresponding to the sub-fields can be found in **YL2.txt**. Therefore, this dataset has two classification tasks. Given an abstract, one can learn to predict its field (easier) and/or sub-field (harder). As such, each training example has two labels, one corresponding to its scientific-domain $l_1 \in [0, 7)$, and one corresponding to its sub-category $l_2 \in [0, 33)$. Both of these labels should be considered when performing experiments. Note that, unfortunately, the YL2 labels do not come with their corresponding English sub-category names.

Part of the assignment task is to pre-process the raw text data from these abstracts to build your own features to be passed into your models. Specifically, you will need to tokenize the input text before passing it into your models.

For the LSTM model, you must design the data pre-processing pipeline that turns the unstructured text data into numerical features. You are free to choose your encoding method, including pre-trained methods like word2vec; however, there should be some justification for your choice in your report.

For BERT, you can use the transformers package to tokenize the input text and convert the tokens into numerical features `https://pytorch.org/hub/huggingface_pytorch-transformers/`. Find out how you can do that from this tutorial `https://www.kaggle.com/code/atulanandjha/bert-testing-on-imdb-dataset-extensive-tutorial` that uses PyTorch.

You are free to use any Python libraries you like to extract features and pre-process data.

Task 2: Implement LSTM and BERT models

You must implement the LSTM model from scratch (i.e., you cannot use Scikit-learn or any other pre-existing implementations of these methods). Note that "from scratch" here means using PyTorch (or equivalent).

In particular, your two main tasks in this part are to:

1. **Implement an LSTM**, without external libraries.
2. **Implementing the BERT model with pre-trained weights with a package, and fine-tune it**.

LSTM Model Details

For the LSTM model, you must use Python, and you must implement the model from scratch (i.e., you cannot use SciKit Learn or similar libraries). Using the torch package is encouraged. Regarding the implementation, we recommend the following approach:

- Implement LSTM model as a Python class. You should use the constructor for the class to initialize the model parameters as attributes, as well as to define other important properties of the model. Note that although the torch package is encouraged, the use of nn.LSTM is not- you must build your own LSTM cell.
- Your model class should have (at least) these functions:
  - Define a `forward` function, which takes as input the initial token embedding and computes one forward pass of the model.

- Define a `fit` function, which takes the training data (i.e., $\mathbf{X}$ and $\mathbf{y}$)—as well as other hyperparameters (e.g., the learning rate and/or number of gradient descent iterations)—as input. This function should train your model by modifying the model parameters.
- Define a `predict` function, which takes a set of input features (i.e., $\mathbf{X}$) as input and outputs predictions (i.e., $\hat{\mathbf{y}}$) for these points.
- Define a functions `evaluate_acc` to evaluate the model accuracy. This function should take the true labels (i.e., $\mathbf{y}$) and target labels (i.e., $\hat{\mathbf{y}}$) as input, and it should output the accuracy score.

BERT Model Details

For the BERT model, you can use pre-trained weights by downloading the already existing pre-trained BERT model from Google or others. You can then use this pre-trained model and fine-tune it on the WOS dataset for a small number of Epochs. You are not required to implement this model from scratch, and you are free to use a package like `https://pytorch.org/hub/huggingface_pytorch-transformers/`.

Task 3: Run experiments

The goal of this project is to have you explore traditional machine learning and deep learning NLP techniques. You will need to conduct a classification experiment on the WOS data and **report the performance using accuracy**. You are welcome to perform any experiments and analyses you see fit (e.g., different datasets, ablation studies, etc.), **but at a minimum, you must complete the following experiments in the order stated below**:

1. In a single table, compare and report the performance of the LSTM and BERT models on the WOS-11967 classification tasks, and highlight the winner.
2. Examine the attention matrix between the words and the class tokens for some of the correctly and incorrectly predicted documents. You will need to choose one of the transformer blocks and use a specific attention head for the multi-layer multi-headed transformer architecture.

You are free to do more experiments to your choosing. As a conclusion, you must answer the following question:

1. Is pretraining on an external corpus (like BERT) good for the scientific text classification task? What do you think pretraining does that might help with this task in particular?
2. What conclusions can you make about the performance difference between the two models? How do the respective architectures aid or hinder the model's ability to perform the task well?

These questions are open-ended and must be answered based on your experiment results. Try to demonstrate curiosity, creativity, rigour, and an understanding of the course material in how you run your chosen experiments and how you report on them in your write-up.

Deliverables

You must submit two separate files to MyCourses (**using the exact filenames and file types outlined below**):

1. **code.zip**: Your data processing, classification, and evaluation code (as some combination of .py and .ipynb files).
2. **writeup.pdf**: Your (max 5-page) project write-up as a PDF (details below).

Project write-up instruction

Your team must submit a project write-up that is a maximum of five pages (single-spaced, 11pt font or larger; minimum 0.5 inch margins, an extra page for references/bibliographical content can be used). We highly recommend that students use LaTeX to complete their write-ups. **You have some flexibility in how you report your results, but you must adhere to the following structure and minimum requirements**:

Abstract (100-250 words) Summarize the project task and your most important findings. For example, include sentences like "In this project, we investigated the performance of linear classification models on two benchmark datasets", "We found that the logistic regression approach achieved worse/better accuracy than naive Bayes and was significantly faster/slower to train."

Introduction (5+ sentences) Summarize the project task, the dataset, and your most important findings. This should be similar to the abstract but more detailed. You should include background information and citations to relevant work (e.g., other papers analyzing these datasets).

Datasets (5+ sentences) Very briefly describe the datasets and how you processed them. Describe the new features you come up with in detail (if any). Present the exploratory analysis you have done to understand the data, e.g. class distribution.

Results (7+ sentences, possibly with figures or tables) Describe the results of all the experiments mentioned in Task 3 (at a minimum) as well as any other interesting results you find. At a minimum, you must report:

1. A comparison of the accuracy of RNN and BERT on WOS-11967.
2. Discussion between the BERT and RNN results.

Discussion and Conclusion (5+ sentences) Summarize the key takeaways from the project and possibly directions for future investigation.

Statement of Contributions (1-3 sentences) State the breakdown of the workload across the team members.

Evaluation

The assignment is out of 100 points, and the evaluation breakdown is as follows:

- Completeness (20 points)
  - Did you submit all the materials?
  - Did you run all the required experiments?
  - Did you follow the guidelines for the project write-up?
- Correctness (40 points)
  - Are your models implemented correctly?
  - Are your reported accuracies close to the reference solutions?
  - Do you observe the correct trends in the experiments (e.g., how well RNN and BERT models perform regarding the accuracy)?
  - Do you observe the correct impact of activation choice, initialization, regularization and normalization on the model performance?

- Writing quality (25 points)
  - Is your report clear and free of grammatical errors and typos?
  - Did you go beyond the bare minimum requirements for the write-up (e.g., by including a discussion of related work in the introduction)?
  - Do you effectively present numerical results (e.g., via tables or figures)?
- Originality/creativity (15 points)
  - Did you go beyond the bare minimum requirements for the experiments?
  - **Note:** Simply adding in a random new experiment will not guarantee a high grade on this section! You should be thoughtful and organized in your report.

Final remarks

You are expected to display initiative, creativity, scientific rigour, critical thinking, and good communication skills. You don't need to restrict yourself to the requirements listed above - feel free to go beyond, and explore further.

You can discuss methods and technical issues with members of other teams, but **you cannot share any code or data with other teams.**