Middle East Technical University | Department of Computer Engineering

CENG 463

Introduction to Natural Language Processing

Fall 2025-2026

Programming Assignment 3

Due date: 2 January 2025, Friday, 23.59

In this programming assignment, you will be dealing with encoder-based and decoder-based language models. You will use Python for this task. You can use libraries for your implementations, or implement your own functions. However, you are expected to analyse and reason about your implementation and results.

Important notes:

- **Do not clear the output of your cells since this notebook will count as your written report and your cell outputs will be used for grading.** If a question in your submitted notebook does not have a printed output, you will get no grade from that question. If you encounter a problem with this, please email me so that we can work out a solution.
- Ideally, you should be able to complete this assignment on Google Colab, without any payment for resources to any service (or you can use your own computers). However, if you believe you need additional resources, please send an email to Çağrı Hoca.

Problem Definition and Details

On your last programming assignment, you will experiment with encoder-based and decoder-based models (specifically, BERT and Llama) to identify misinformation on English and Turkish tweet data from 2022. You can read the paper for more information about the dataset. However, the dataset that is shared with you with the assignment is slightly simplified, where each row only consists of a `label` and a `text` field. A tweet is considered as "misinformation" if the `label` is `False`. Your models should aim to classify a given text according to the labels `True`, `False`, and `Other`.

Put the "data" folder in the same directory with your notebook to work on your solutions in case we need to run your notebooks to reproduce your outputs. The dataset consists of 5 train and test folds. For each model, you will train on 5 folds and report the average performance of all folds. On the last part of the assignment, you will compare the time and space efficiency of the models you have used. So, don't forget to keep track of the training time.

Important Note: For this assignment, using tutorials, online forums, or AI tools to help with debugging and implementation may be needed. However, you are expected to add your resources as disclaimers. **You will be penalized if you do not disclose any help from AI tools, GitHub repositories, or tutorials.**

Tutorials/resources

Here are some resources that might come in handy:

- `MiDe22` repository
- `transformers` library
- finetuning a pretrained model
- `pipeline` from HuggingFace for inference
- finetuning Llama
- quantization for working on low resources

or any other tutorial you find easy to follow. If you find a particularly helpful resource, you can share it in the discussion forum for everyone to use.

Q1 - Encoder-based language models for classification (25 points)

Part A: BERT and preprocessing

In this part, you will finetune the `google-bert/bert-base-uncased` model for the misinformation detection task for English tweets ("EN" folder in the dataset). However, you will train two versions: one with preprocessed texts, one with the raw texts.

For the preprocessing steps, lowercase the text, and use `nltk` to lemmatize and remove stopwords. You can also split each tweet into sentences and tokens, then combine the token list into a single space-separated string to represent each tweet as a single text again. Be careful not to combine different tweets. You can add additional preprocessing steps as long as you keep the integrity of the label-text mappings for each tweet. You can also use the `preprocess_review` function from PA2 and then combine the returned list into a single space-separated string.

Report the performance of both classifiers as the average of 5 train/test folds in "EN" dataset and accuracy, precision, recall, and F1-score with respect to the `False` class. Discuss which model performed better and why.

Part B: BERT, Mutlilingual BERT and crosslingual performance

In this part, you will finetune and compare the performances of `google-bert/bert-base-uncased` and `google-bert/bert-base-multilingual-uncased` models. However, for the train/test folds, you will take the train fold from "EN" folder and the test fold with the corresponding number from the "TR" folder. This means that you will train the models on English but test them on Turkish.

Report the performance of both classifiers as the average of 5 train/test folds on accuracy, precision, recall, and F1-score with respect to the `False` class. Discuss which model performed better and why you think that is the case.

Q2 - Decoder-based language models for classification (55 points)

Part A: Zero-shot and Few-shot inference with Llama for text classification

In this part, you will use the `meta-llama/Llama-3.1-8B-Instruct` model and the "EN" English tweets dataset for designing a classifier based on inference. You will design two prompts:

- **Prompt1** A zero-shot prompt with only the task description, no examples. Explain the task and ask for a classification of the tweets in test folds.
- **Prompt2** A one-shot or few-shot prompt, with one or a couple examples. This can be tricky as longer tweets may cause problems. Explain the task using examples from the training folds, and ask for a classification of the tweets in test folds.

Report the performance of both classification pipelines as the average of 5 test folds on accuracy, precision, recall, and F1-score with respect to the `False` class. Explain your prompt design process in detail and discuss the performance of the models with respect to your prompts.

Part B: Finetuning Llama

In this part, you will again use the `meta-llama/Llama-3.1-8B-Instruct` model and the "EN" English tweets dataset, but this time you will try to finetune the model with the training folds. However, this may become tricky and resource demanding, considering you are using free tools (such as Google Colab) or your own computers. Check out the tutorials to see how you can better manage your resources.

Report the performance of the finetuned classifier as the average of 5 test folds on accuracy, precision, recall, and F1-score with respect to the `False` class, with the better performing prompt you have designed in Part A. Discuss the effect of finetuning on the model performance. Do you think it improved the performance? What else can be integrated into the process to improve the performance of the model, specifically for the misinformation detection task?

Part C: Domain transfer to sentiment analysis

In this part, you will try to classify game reviews with the model you finetuned in Part B to see the effect of finetuning on the generalized performance of the model.

You should design a simple zero-shot prompt that asks the model whether a user recommends the game given their review text. With the same prompt, you will use both the base and your fine-tuned version of the `meta-llama/Llama-3.1-8B-Instruct` model to classify the reviews. Report the performance of both classifiers on the whole dataset with respect to accuracy, precision, recall, and F1-score. Discuss and explain your findings.

Note: Dataset for Part C
You will use the `game_reviews.csv` file shared with you. It is taken from this Kaggle dataset. Review text attribute is `user_review`. User recommendation is the attribute `user_suggestion` where `1` means the user recommended the game, `0` means the user did not recommend the game.

Q3 - Discussion of classification performance and resource use (20 points)

In this part, you are expected to discuss the misinformation detection performance and resource use of the models you have trained in Q1 Part A-B and Q2 Part A-B. Which model performs better? Which model uses more resources? Please discuss your OWN findings with YOUR OWN WORDS. Do not forget to share the time and resource need details you have measured and observed.

1 Specifications and Regulations

- You will implement your solutions and write your discussion in the shared `e1234567_ceng463_pa3.ipynb` file. You should change the student ID placeholder in the file name to your student ID before submitting your solutions.
- Descriptions of the questions are also written in the shared notebook. Please obey the directives in the notebook for modifying the cells.
- Please add comments to your code when necessary.
- Limited discussions and codes with no explanation will result in lower grades.
- You are also given the dataset files to be used for the questions. Do not modify the datasets directly.
- Please obey the folder structure of the downloaded student pack to make sure that your notebook can read the dataset if needed files when grading.
- If you have never worked with an iPython Notebook before, check out Jupyter or Google Colab.
- This is an individual work. We expect you to complete this assignment by yourself, ideally with no generative AI tool assistance. If you do not disclose the use of AI tools as specified in the previous sections, you will be penalized upon discovery of any AI tool usage.
- Do not clear the output of your cells since this notebook will count as your written report, and your cell outputs will be used for grading. If a question in your submitted notebook does not have a printed output, you will get no grade from that question.