Artificial Intelligence II

Deep Learning for Natural Language Processing

Spring Semester 2025-2026

Homework 3

Weight: 25% of Course Grade

Announced: May 4, 2026

Due: May 21, 2026, before 23:55

Description

You will develop a classifier for the task of **response clarity classification** using **prompting techniques with instruction-tuned language models**, rather than gradient-based fine-tuning. The dataset consists of political question–answer pairs (CLARITY dataset). Each instance contains a question and its corresponding answer, and your goal is to classify the answer into one of the following categories: *Clear Reply*, *Ambivalent*, or *Clear Non-Reply*.

The main focus of this homework is the systematic comparison of prompting strategies across models of different sizes. Indicatively, you should work with the following Qwen-family models: `Qwen/Qwen3.5-0.8B`, `Qwen/Qwen3.5-2B`, and `Qwen/Qwen3.5-4B`. You may also experiment with other model families if you wish, but the Qwen comparison is the core requirement.

In particular, you must investigate prompting approaches such as:

- **zero-shot prompting**,
- **few-shot prompting**,
- **chain-of-thought (CoT) prompting**,
- and any other prompting technique that you consider relevant.
- Information about these techniques, as well as additional prompting techniques, can be found at `promptingguide.ai/techniques`.

You should use the Hugging Face Transformers ecosystem and any supporting Python tools that help you run and evaluate prompted inference pipelines. The emphasis is on careful experimental design, prompt construction, and analysis of model behavior.

Before you start the homework, make sure that you have studied the relevant slides of the course, especially the material related to transformers, large language models, prompting, and in-context learning, as well as any additional material that you may find useful.

Your goal is to develop a strong prompting-based system, but also to understand the behavior of different prompting methods and models on this task. It is your responsibility to decide how to build an effective prompting pipeline: how to structure the prompt, how to constrain outputs to valid labels, how to select

few-shot examples, how to parse model generations robustly, how to compare models fairly, and how to ensure that your conclusions are well supported by evidence.

Additionally, conduct a systematic comparison with the results obtained from the previous assessments (i.e., vector-based classifiers and encoder-only models). Analyze the differences in overall performance as well as in the observed failure modes. To what extent do the methods exhibit similar error patterns? Do they succeed and fail on the same instances, or do they demonstrate complementary behaviors across different cases? Your analysis should explicitly relate these findings to the earlier results, highlighting whether the prompting-based approach shares common limitations with prior methods or reveals distinct strengths and weaknesses.

Guidelines

This task consists of the following main steps:

1. **Prompt design:** Decide how you will present the question–answer pair to the model and how you will formulate the classification instruction. State your prompt format explicitly.
2. **Prompting strategies:** Implement and compare multiple prompting approaches, including at least **zero-shot**, **few-shot**, and **chain-of-thought (CoT)** prompting.
3. **Model comparison:** Evaluate models of different sizes, with the main comparison centered on `Qwen/Qwen3.5-0.8B`, `Qwen/Qwen3.5-2B`, and `Qwen/Qwen3.5-4B`.
4. **Inference pipeline:** Decide how you will handle generation settings, output constraints, post-processing, invalid outputs, and reproducibility.
5. **Evaluation and comparison:** Compare prompting methods and model sizes systematically. On Kaggle, the primary evaluation metric is **F1-score**, while **Macro F1-score** should be reported as an additional metric. In your report, also include **accuracy**, **precision**, **recall**, and **F1-score**.
6. **Error analysis:** Analyze failure cases and discuss which prompting choices or model properties make some examples easier or harder.

It is your responsibility to choose all details of developing an effective prompting-based system, including prompt wording, demonstration selection, decoding decisions, output normalization, validation setup, and the criteria used to compare methods fairly.

For implementation purposes, students **must use a random seed** to ensure reproducibility.

Presentation of Results: Ensure that your results are supported with clear, high-quality, and well-labeled plots or tables that effectively illustrate your findings. For example, compare prompt variants, model sizes, decoding choices, and performance across different subsets of the data.

Additional Considerations: You are encouraged to mention in the appendices of your report any other prompting or inference approaches you explored that did not improve performance but contributed to your understanding of the task.

Experiments

You must conduct a systematic comparison between prompting strategies and model sizes. Your experimental analysis should go beyond reporting final scores. In particular, you should investigate:

- the effect of prompt formulation choices,
- the effect of different prompting methods (zero-shot, few-shot, CoT, and others you choose),
- the effect of model size on performance and robustness,

- differences between smaller and larger models,
- whether some prompting methods benefit small and large models differently,
- performance across different data subgroups (e.g., categories based on question length and answer length).

Your discussion must explain observed differences in performance rather than simply listing numerical results.

Error Analysis

A central component of this assignment is the analysis of model failures. You must perform a detailed error analysis and discuss questions such as:

- Which class is hardest to predict?
- Are there recurring linguistic patterns in incorrect predictions?
- Do stronger or larger models fail differently from smaller models?
- Do different prompting methods fail for different reasons?
- What is the fundamental source of error: what do prompted models fail to understand in question–answer interactions?
- What characteristics would an ideal prompting or reasoning system need in order to improve performance on this task (even if not implemented in this homework)?

You must include concrete examples of errors and organize your discussion in a meaningful way. In addition, you must describe:

- what changes you attempted in order to improve your prompts or inference pipeline,
- whether these changes improved performance,
- why you believe these changes helped or failed.

The main objective is not only to obtain a strong classifier, but also to understand why prompting strategies and model scales succeed or fail on this task.

Kaggle Competition

Submit your code as a **Jupyter Notebook** via the Kaggle competition. Follow these rules:

- Your team name must be your academic identification number (Αριθμός Μητρώου - sdiXXYYYYY or the one for graduate students).
- You must create and submit **one Kaggle notebook** that contains your full analysis, comparisons, and final submission pipeline.
- The notebook must include a systematic comparison of prompting methods and model sizes, and it must clearly identify the **best-performing combination of prompting technique and model**.
- Notebook naming format (use your academic ID): `[academic-id] homework-3-prompting`. Share the notebook on Kaggle with the Teaching Assistant responsible for grading this assignment (Kaggle username: `myrtotsokanaridou`).

- Your notebook must output a prediction file in csv format named, for example, `submission_best_prompting_system.csv`. Do **not** submit as a file upload.
- On Kaggle, the primary metric is **F1-score**; report **Macro F1-score** as an additional metric in your analysis.
- The resulting file must follow the format specified in the provided `sample_submission.csv` file and must contain the predictions of your **best prompting-based system** over the test set.
- In your report, clearly state which combination of model and prompting strategy performed best overall and explain in detail why, based on your evaluation findings.
- **Do not share your notebook publicly.**

Report

For this project, you are asked to create a detailed report. For this reason, we provide you with a LaTeX template. You may use the Overleaf online editor. Open Overleaf, create an account if you do not already have one, and upload the provided zip file by selecting *New Project* and then *Upload Project* (the template uses the pdfLaTeX compiler). If you encounter difficulties writing in LaTeX, you may initially draft your report in another editor following the structure of the template; however, you are strongly encouraged to produce the final version in LaTeX, as tools such as Overleaf or Prism can significantly simplify the writing process.

Your report must include: a description of the models you used; your prompt design choices; the prompting and inference strategies you evaluated; your evaluation results; a comparative analysis of prompting methods and model sizes; a detailed error analysis; and a discussion of the modifications you tried and their effect.

The emphasis of the report is not only on predictive performance, but also on: understanding model behavior under prompting; explaining performance differences across prompting strategies and model sizes; analyzing model limitations; and justifying experimental decisions.

The report must be written in English for students in the Master's program in Data Science and Information Technologies. All other students may choose their preferred language. The report must include the link to your Kaggle notebook, which must be shared with the teaching assistant and must not be made publicly available.

Grading

Implementation: Code, Kaggle submission [**Total 70%**]

- Data handling and preprocessing: [**10%**]
- Prompting pipeline development: [**20%**]
- Experiments: [**30%**]
- Prompting & inference optimization: [**10%**]

Report: Analysis and Presentation [**Total 30%**]

- Experiments: [**10%**]
- Analysis: [**15%**]
- Plots: [**5%**]

Important grading note: Code submissions without clear prompting-method and model-comparison analysis in the report will **not** receive implementation credit.

Submission Guides

We expect you to:

1. Submit your **executed Jupyter Notebook** (with outputs) and make it available to supervisors in **Kaggle** only.
2. Submit your report in **.pdf** format in e-class. Name your report as **[full-id].pdf**.

For example: `ZZZZZZXXYYYYY.pdf`
**We won't accept code submissions from e-class, e-mails, etc.*

Support

Myrto Tsokanaridou is supervising this assignment. For any questions, please post them on Piazza.