CSIT6000R Natural Language Processing

Individual Project: Evaluation of Language Models

Spring 2025

1 Introduction

In this project, you will evaluate language models (LMs). This project contains two parts: you will test examples and explore different properties of LMs.

- In Section 2, you will evaluate the performance of LMs on natural language inference. You will need to implement an NLI classifier using LMs, and then apply it to a downstream task for hallucination detection.
- In Section 3, you will measure the bias of LMs on a demographic that interests you. We provide an approach to evaluate the bias of masked language models for your reference.

Language Models: In Section 2, please evaluate at least **two** language models on natural language inference. In Section 3, you will evaluate biases, stereotypes, and associations in at least **two** masked language models. Here we provide a recommendation list of language models: (i) Masked Language Models: BERT, RoBERTa, DeBERTa; (ii) Causal Language Models: GPT2, XLNet, LLaMA; (iii) Seq-to-Seq Language Models: BART, T5, Flan-T5. You can also use other LMs, but only testing models with APIs-calling (e.g., for models like ChatGPT, GPT4, Claude) will lead to deduction of scores, as we hope you will learn to deploy and evaluate models locally.

There are multiple sizes for each language model, such as `roberta-base` and `roberta-large` (125 million and 355 million parameters, respectively). You can choose any size to conduct the following experiments. Also, some models may have `cased` and `uncased` versions, such as `bert-base-cased` and `bert-base-uncased`. In this project, you can use either version.

Learning Goals: Upon finishing the project, you will understand: (1) how to evaluate LMs with standard practices, (2) how the evaluation of LMs is sensitive to your design choices.

2 Natural Language Inference (60%)

2.1 Applying LMs for NLI (40%)

In this section of the project, you will assess the performance of language models on the natural language inference (NLI) task. We will focus on the three-way classification NLI, where each input comprises a pair of contexts: the premise and the hypothesis. Given the premise text, The goal of the NLI task is to determine whether the hypothesis is entailed (always true), contradicted (always false), or neutral (neither entailed nor contradicted). For a more detailed discussion of this task, please refer to [1].

We use the MultiNLI dataset [5][1]. MultiNLI is one of the largest corpora available for natural language inference, improving upon prior datasets in both its coverage and difficulty. Performance on this dataset is measured using **accuracy**, and a random guess would achieve a performance of around 33%. Here we require you to conduct prompting or fine-tuning for solving this natural language inference task. For reducing the computational cost for you, we sampled 5000 evaluation samples.

[1] Please use the evaluation set that we provide to you on Canvas.

Met my first girlfriend that way.	FACE-TO-FACE **contradiction** C C N C	I didn't meet my first girlfriend until later.
8 million in relief in the form of emergency housing.	GOVERNMENT **neutral** N N N N	The 8 million dollars for emergency housing was still not enough to solve the problem.
Now, as children tend their gardens, they have a new appreciation of their relationship to the land, their cultural heritage, and their community.	LETTERS **neutral** N N N N	All of the children love working in their gardens.
At 8:34, the Boston Center controller received a third transmission from American 11	9/11 **entailment** E E E E	The Boston Center controller got a third transmission from American 11.
I am a lacto-vegetarian.	SLATE **neutral** N N E N	I enjoy eating cheese too much to abstain from dairy.
someone else noticed it and i said well i guess that's true and it was somewhat melodious in other words it wasn't just you know it was really funny	TELEPHONE **contradiction** C C C C	No one noticed and it wasn't funny at all.

Figure 1: Randomly sampled examples from the dev set of MultiNLI. Each line shows the premise text (left) and the hypothesis text (right) together with their genre labels, gold labels (bolded), and the individual annotator labels (E, N, C are abbreviations for Entailment, Neutral, and Contradiction).

There are 2500 samples in the matched setting, and the other 2500 samples in the mismatched setting. For details of these setttings, please refer to [5]. The samples are given on the assignment page on Canvas, and you can directly use them.

Here are some possible methods for completing this task.

Prompting. For causal LMs (e.g., GPT-2) and seq-to-seq LMs (e.g., T5), the NLI task can be addressed using prompting. Here's a straightforward three-step procedure:

1. **Construct a Query**: Create a query from the input x to be submitted to the language model. For guidance, you may refer to the reference on prompt designs mentioned at (https://stanford-cs324.github.io/winter2022/lectures/capabilities/).

2. **Specify Decoding Hyperparameters**: Consult the paper and model documentation to choose appropriate decoding hyperparameters.

3. **Define a Verbalizer**: Map the language model's response to a class label $\hat{y}$. The verbalizer $v : L \to V$ links each label to a vocabulary entry. A simple approach is to map each label to its corresponding string (e.g., mapping the entailment category to "entailment"), but alternatives like "true" or "correct" might be more effective.

After defining the query and decoding parameters, input them into the language model. The model will return a response, which could be a probability distribution over next words, likelihood, or a sampled completion. Transform this response into a predicted label $\hat{y}$ using the verbalizer. For example, if the model returns a probability distribution p over V, the predicted label $\hat{y}$ is:

$$\hat{y} = \arg\max_{y \in L} p(v(y)).$$

Fine-tuning. In the prompting approach, we only inference with LMs (i.e., their parameters are not updated according to the data). You may also choose to fine-tune the models. To achieve this, you might need to use additional training data from the MultiNLI dataset.

Note: If you choose the prompting approach, it is not recommended to only evaluate LMs through API calls (e.g., from OpenAI, Claude, etc.). We expect you to at least prompt one model locally. You might test smaller models like GPT-2, Flan-T5, etc. Lower performance would NOT hurt your grading results as long as it is reasonable.

For finetuning, you should fine-tune it by yourself, you are not allowed to directly use the checkpoints that are fine-tuned by other people.

2.2 NLI for hallucination detection (20%)

In Section 2.1, we have explored building models to predict NLI labels based on LMs. In this section, we will see how to leverage such models for downstream applications.

We focus on the task of hallucination detection. The dataset is `wikibio-gpt3-hallucination` (https://huggingface.co/datasets/potsawee/wiki_bio_gpt3_hallucination) [2], which contains GPT-3 generated Wikipedia-like passages using the prompt: `This is a Wikipedia passage about concept`, where concept represents an individual from the WikiBio dataset. Each entry in the dataset contains the GPT3 generated sentences (`gpt3_sentences`), the corresponding Wikipedia reference (`wiki_bio_text`), as well as human annotated labels (`annotation`) for whether each generated sentence contains hallucination.

The meaning of dataset labels are presented below[2].

- Major Inaccurate (**Non-Factual**, 1): The sentence is entirely hallucinated, i.e. the sentence is unrelated to the topic.
- Minor Inaccurate (**Non-Factual**, 0.5): The sentence consists of some non-factual information, but the sentence is related to the topic.
- Accurate (**Factual**, 0): The information presented in the sentence is accurate.

To conduct hallucination detection, you may treat it as a binary classification task (consider using the two labels Non-Factual v.s. Factual). You will need to report the sentence-level detection performance in terms of **accuracy**, **precision**, **recall**, and **F1**.

Using NLI models to solve this detection task might require some adaptation. For instance, one possible way is to treat the reference `wiki_bio_text` as premise and each sentence from (`gpt3_sentences`) as a hypothesis. You may then input them to your models to acquire NLI labels and scores, and then transform them to the labels for hallucination detection.

Your task is to apply the model you developed to this dataset. You are allowed to access NLI models finetuned by others to establish baseline performance.

2.3 Deliverables

Your report should include:

Setup.

1. **Method.** Describe the method you select to conduct the experiment for Section 2.1 and 2.2.
2. **Implementation details.** Please include important implementation details, such as model size, the prompt and hyper-parameters you use, etc.
3. **Experimental details.** Include experimental details such as how you set up the datasets and models for evaluation, how the model outputs are transformed to classification predictions, etc.

Results.

1. **Quantitative results.** For Section 2.1, you should include the quantitative results on both the matched and mismatched development sets of MultiNLI. You do not need to report the performance on the MultiNLI test set as they are not publicly available.

   For Section 2.2, include the classification results using your implemented NLI models. You may also evaluate NLI models on the Internet to set up baseline performance, but that is not mandatory.
2. **Case study.** At least an example of model input and outputs should be depicted in details. In addition, you can compare the results from different models to provide further discussions.

Note: Your grades are not decided by the performance your models achieve. As long as the performance is reasonable (the standard of "reasonable" will be different for different models), you will get full mark for this part. Please refer to the grading rubrics at the end of this document, which are correctness, coverage, clearness, and insights.

[2]Details in Section 6 in the paper [2].

3 Biases in Language Models (40%)

In this part of the project, you'll evaluate biases, stereotypes, and associations in Masked LMs concerning social groups using the "Minimal Pairs" method. This approach focuses on various social domains such as race, gender, and political ideology. You'll experiment with two masked language models listed in Section 1 or other models you prefer.

To start with, you will choose a group that is important to you (e.g., Hispanic, female, Chinese, LGBT+) and study the social domain containing that group throughout this part. You will measure biased stereotypes related to that domain, especially when mentioned in the text. For example, if the *female* group is important to you, you can select *gender/gender identity* as the domain you study in this part [4].

Methodology –Minimal Pairs: Previous work [3, 4] introduces evaluation methods known as minimal pairs. In this approach, a model is presented with pairs of sentences that differ by only a few words, contrasting stereotypical and astereotypical associations. The logic is that discrepancies in the probabilities assigned by a language model to these sentences reveal whether the model has a bias or preference for a particular stereotype. Ideally, an unbiased language model should assign equal probabilities to both the stereotypical and astereotypical sentences in each pair.

Dataset and Social Domain Please use the dataset "CrowS-Pairs" curated by [4], which recognizes nine social domains for detecting bias and stereotypes in language models: race/color, gender/gender identity, sexual orientation, religion, age, nationality, disability, physical appearance, and socioeconomic status/occupation.

You need to choose a domain that contains the group important to you. You should identify sentence pairs that encode stereotypes related to this domain to evaluate LMs (and abandon sentence pairs related to other domains).

Some domains contain much more examples than others, i.e., race/color, gender/gender identity, socioeconomic status/occupation, nationality, and religion. If you choose one of those, you are allowed sample 80 examples to conduct the experiments.

Metric Please use the pseudo-log-likelihood proposed in Section 3 in the CrowS-Pairs paper [4]. This metric use Masked LMs to compute pseudo-log-likelihood scores for each sentence. Then, this metric measures the percentage of pairs for which a model assigns a higher pseudo-log-likelihood to the stereotyping sentence. A masked LM without any stereotypes shuold achieve the ideal score of 50%.

3.1 Deliverables

Your report should include:

Setup.

1. **Social group.** The social group and domain you choose. Please also include the total number of examples you used.
2. **Metric.** A clear explanation of how the pseudo-log-likelihood metric works.
3. **Implementation details:** Please also include some important implementation details, such as model size and hyper-parameters.
4. **Experimental details.** Include other details for how you set up the experiments.

Results.

1. **Quantitative Results.** Quantitative results for evaluating masked language models on your bias evaluation data. You should compare the biases of the models you choose.
2. **Case Study.** Three examples of sentence pairs should be depicted in a *figure* or *table*, demonstrating specific biases in both models.

4 Logistics

4.1 Grading Rubrics

- Correctness of implementation. We will examine your implementation of models and experimental setup. Note that your points would not be deducted solely because of low performance, as long as the model performance is within a normal range.
- Content clearness.
- Coverage. Whether essential elements (e.g., case study) are covered in the report. Higher completeness and coverage are encouraged, such as more models, analyses, case studies, etc. Also, notice that we do not grade based on the sizes of your language models as we are trying to keep this project computation-efficient.
- Insights. Whether there are unique understandings of the field when analyzing cases.

Plagiarism. We will use Turnitin on canvas to check similarity scores. Penalties will be added to those whose similarity scores are greater than 30%.

4.2 Data

All of the datasets you will use in the project can be accessed through the Python package *Hugging Face Datasets*. Documentation and tutorials for how to use this package to download and interact with these datasets can be found at https://huggingface.co/docs/datasets. You can find the list of downloadable datasets at https://huggingface.co/datasets.

4.3 Models

All the pre-trained LMs you will use in the project can be acquired using the Python package *Hugging Face Transformers*. Documentation for this package can be found at https://huggingface.co/docs/transformers/index. You also can find documents for each model with example code in the list of supported models at https://huggingface.co/docs/transformers/index#supported-models. Meanwhile, model cards are also a great reference for you to learn how to use each language model, which can be found at https://huggingface.co/models.

4.4 Submission

Report. The main deliverable for this project is a report, which should ideally be written in LaTeX or otherwise in Microsoft Word, Pages for Mac, or an equivalent program, and submitted as a PDF. This report should include the results of both parts (Section 2 Capabilities and Section 3 Risks). The name of this report should be `CSIT6000R_individual_<your_itsc_id>.pdf`. Replace `<your_itsc_id>` with your itsc id without angle brackets.

Your report should properly cite all papers and resources you used in your project. Furthermore, you should acknowledge any individuals other than yourself who helped you in the report.

Code. We require that you submit a *.zip* file of your codebase. We do not expect to run the code you use in the project though, if we find plagiarism in the report, we reserve the right to do so. We hope not to have to do this and operate on a trust-based system for this course. The name of this file should be `CSIT6000R_individual_<your_itsc_id>.zip`

Note: All materials should be uploaded to Canvas by 23:59 on 16^{th} May. Submission after the due date will incur 20% penalty per day.

References

[1] Samuel R. Bowman et al. "A large annotated corpus for learning natural language inference". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 632–642. DOI: 10.18653/v1/D15-1075. URL: https://aclanthology.org/D15-1075.

[2] Potsawee Manakul, Adian Liusie, and Mark Gales. "SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 2023, pp. 9004–9017.

[3] Moin Nadeem, Anna Bethke, and Siva Reddy. "StereoSet: Measuring stereotypical bias in pretrained language models". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2021, pp. 5356–5371.

[4] Nikita Nangia et al. "CrowS-Pairs: A Challenge Dataset for Measuring Social Biases in Masked Language Models". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 1953–1967.

[5] Adina Williams, Nikita Nangia, and Samuel Bowman. "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 1112–1122. URL: http://aclweb.org/anthology/N18-1101.