CS6493: Natural Language Processing - Projects

Instructions

1. Due at 6pm, Apr. 22, Monday, 2025.
2. This is the group project. Each group has 1-6 members. Please register your group before 6pm, March 9, 2025 on Canvas-People-Groups.
3. Select one topic among the following 6 topics for your group.
4. You are required to submit the project report and source code via Canvas and give a 15-min presentation for your project in class on April 22. The project report should at least consist of following parts, including introduction, related work, methodology, experiments and discussions. The report may contain up to 6 pages of main content, plus unlimited pages of references and appendix. The source code can be submitted by Jupyter Notebook or Python files.
5. Please attach your presentation slides at the end of the report.
6. This project is very open, and you are highly encouraged to come up with fantastic ideas and designs!
7. If you have any questions, please post your questions on the Canvas-Discussion forum or contact TA Mr. Guanzhi Deng or Mr. Sichun Luo or Mr. Mingyang Liu (email: {guanzdeng2,sichunluo2,mingyaliu8}-c@my.cityu.edu.hk).

Topic 1 - Mathematical Reasoning Ability of Large Language Models

Large language models (LLMs) have demonstrated remarkable capabilities in natural language processing. However, their mathematical reasoning ability remains a critical area of research, especially when dealing with complex problems found in challenging datasets such as MATH-500, GSM8K and AIME 2024. These datasets require models to interpret mathematical expressions, understand abstract concepts, and generate accurate solutions.

The prompt-based methods, such as Chain of Thought (COT) and Self-Refine prompting, are structured approaches to mathematical problem solving. These methods provide specific cues to break down complex problems into manageable steps, guiding models towards logical solutions. To solve the mathematical problems, you are encouraged to:

1. Explore the effectiveness of different prompt methods and different models in mathematical reasoning, including but not limited to: COT (Wei J, Wang X et al.), Self-Refine (Madaan A, Tandon N, Gupta P, et al.), Self-Consistency (Wang X, Wei J, Schuurmans D, et al.), etc. You should experiment with at least three different prompt methods and test them on the following two models:

   - Qwen2.5-Math-1.5B
   - DeepSeek-R1-Qwen-1.5B

   You should collect and preprocess the MATH-500, GSM8K (Test Set) and AIME 2024 dataset, then evaluate the models on them.

2. In the metrics section, focus on two key concepts:

   - Accuracy: the ratio of correctly solved problems to the total number of problems.
   - Response length: the number of characters or words in a response.

You are also encouraged to explore new prompt methods or evaluation metrics.

Hint: The token consumption of Self-Consistency method may be extremely high, so you are suggested to set the iteration number to be 5.

Reference

1. Wei J, Wang X, Schuurmans D, et al. Chain-of-thought prompting elicits reasoning in large language models[J]. Advances in Neural Information Processing Systems, 2022, 35: 24824-24837.
2. Madaan A, Tandon N, Gupta P, et al. Self-refine: Iterative refinement with self-feedback[J]. Advances in Neural Information Processing Systems, 2024, 36.
3. Wang X, Wei J, Schuurmans D, et al. Self-consistency improves chain of thought reasoning in language models[J]. arXiv preprint arXiv:2203.11171, 2022.
4. Cobbe K, Kosaraju V, Bavarian M, et al. Training verifiers to solve math word problems[J]. arXiv preprint arXiv:2110.14168, 2021.

Topic 2 - Hallucination Detection and Correction in LLMs

Large language models (LLMs) have demonstrated remarkable capabilities in many tasks. Despite their fluency, LLMs frequently generate factually incorrect statements—known as *hallucinations*— due to limitations in training data, contextual understanding, or reasoning. This project evaluates hallucination rates using datasets like TruthfulQA (Lin et al., 2022), which measures models' propensity to replicate human falsehoods, and HaluEval (Li et al., 2023), a benchmark covering diverse hallucination types (e.g., factual, contextual). The investigation spans three dimensions: factual accuracy (verifiable claims), contextual coherence (logical consistency within a response), and citation reliability (proper sourcing). Developing robust hallucination detection systems is critical for high-stakes applications in healthcare, legal analysis, and news generation.

The following approaches are recommended for systematic evaluation and correction:

1. **Benchmark hallucination rates across models:**
   - Compare `closed models` (GPT, Claude) vs `open models` (Llama, DeepSeek):
   - Test different prompting strategies. Compare at least three methods:
     (a) Standard prompting: Direct queries
     (b) Citation enforcement: Require inline references (e.g., "According to the ...").
     (c) Self-reflection: Ask models to self-assess credibility
   - Hallucination taxonomy analysis: Classify errors into: - *Fabrication*: Generating entirely false information. - *Distortion*: Misrepresenting true facts (e.g., swapping dates). - *Omission*: Excluding critical context (e.g., not mentioning vaccine side effects).
2. **Strategies to Mitigate Hallucinations:**
   - Fine-tuning on factual data: Train models on curated datasets (e.g., biomedical journals, court rulings) to strengthen factual grounding.
   - Retrieval-augmented generation (RAG): Integrate real-time knowledge retrieval from trusted sources before generating responses.
   - Post-hoc correction: Implement a secondary model to detect and rewrite hallucinated content.

References

1. Lin S, Hilton J, Evans O. Truthfulqa: Measuring how models mimic human falsehoods. ACL, 2022.
2. Li J, et al. HaluEval: A Large-Scale Hallucination Evaluation Benchmark for Large Language Models. EMNLP, 2023.
3. Min S, et al. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. EMNLP, 2023.
4. Ji Z, et al. Survey of hallucination in natural language generation. ACM computing surveys, 2023.

Topic 3 - Building Practical LLM Applications with LangChain

LangChain is a cutting-edge framework for integrating large language models (LLMs) with external data sources and computational workflows. This project focuses on developing production-grade LLM applications with emphasis on educational implementation rather than enterprise deployment. Students will explore core challenges in real-world LLM application development while considering computational constraints.

The project requires addressing two core technical challenges:

1. **System Architecture Design**: Construct application pipelines using LangChain's modular components:
   - Implement **at least one** application types from:
     - Document QA System: Build RAG pipelines with adaptive chunking strategies (e.g., 256-token chunks with 10% overlap)
     - Conversational Agent: Develop chatbot with short-term memory management
     - Autonomous Agent: Create simple task-oriented agents with API integration
   - Integrate data connectors for **at least one** source type (PDFs, text files, or web content)
   - Compare performance of 2+ LLM backends (e.g., Mistral-7B vs. smaller models like T5-base)
2. **Capability Evaluation**: Establish practical evaluation metrics:
   - Design test cases measuring `response relevance` and `task completion rate`
   - Analyze memory-performance tradeoffs using different chunking strategies

Advanced suggestions: Explore quantization techniques for model deployment or implement basic human feedback mechanisms.
Hint: For computational efficiency, consider using quantized models (e.g., GPTQ-4bit versions) through Ollama.

Reference

1. Touvron H, et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. Meta 2023.
2. Jiang W, et al. Mistral 7B. arXiv preprint arXiv:2310.06825 (2023)
3. Xiao G, et al. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. ICML 2023.
4. LangChain Documentation: https://python.langchain.com/docs/get_started/introduction

Topic 4 - Retrieval-Augmented Generation for Knowledge-Intensive Tasks

Retrieval-Augmented Generation (RAG) combines the strengths of neural retrieval and language generation, offering promising solutions for knowledge-intensive NLP tasks. This topic explores how effectively RAG systems can leverage external knowledge sources to enhance answer accuracy and reduce hallucinations. Using datasets like Natural Questions (Kwiatkowski et al.) or HotpotQA (Yang et al.), you will evaluate how retrieval quality and generation strategies interact. Metrics include factual correctness, citation quality, and hallucination rates. Understanding RAG's capabilities has implications for enterprise search, education, and domain-specific QA systems.
The project topic focuses on two key components:

1. **Retrieval-Generation Interaction**: Investigate different retrieval strategies (sparse/dense/hybrid retrieval) paired with various LLMs (e.g., **open-source** models like Llama or **closed-source** models like GPT). You should:
    - Implement at least 2 retrieval methods (e.g., BM25 vs. Contriever) and 2 generation models
    - Analyze how retrieval precision affects final answer quality using the HotpotQA dev set
    - Compare zero-shot vs. instruction-tuned models (e.g., `https://huggingface.co/Intel/neural-chat-7b-v3-3`)
2. **Hallucination Mitigation**: Design experiments to measure how RAG reduces model fabrication:
    - Quantify hallucination rates using metrics like `FActScore` (Min et al.)
    - Evaluate citation quality through human assessment (e.g., citation precision/recall)
    - Compare vanilla RAG vs. advanced variants (e.g., Self-RAG (Asai et al.))

Students are encouraged to explore innovative retrieval strategies (e.g., query rewriting) or propose new evaluation frameworks.
Hint: Limit retrieved documents to 3-5 per query to balance performance and computational cost.

Reference

1. Lewis P, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. NeurIPS 2020.
2. Izacard G, et al. Leveraging passage retrieval with generative models for open domain QA. EACL 2021.
3. Asai A, et al. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. ICLR 2024.
4. Min S, et al. FActScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation. EMNLP 2023.
5. Yang Z, et al. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. EMNLP 2018.

Topic 5 - Smart Meeting Assistant

In this project, you are expected to design a smart meeting assistant that enhances virtual and in-person meetings by providing real-time support for participants. The system should assist users by transcribing conversations, summarizing discussions, extracting key action items, and offering relevant insights.
Your smart meeting assistant should incorporate the following key features:

1. **Real-time Speech-to-Text Transcription**: Implement speech recognition to accurately transcribe spoken conversations into text. The system should support multiple speakers and differentiate between them for better clarity in meeting notes.
2. **Automatic Meeting Summarization**: Develop an intelligent summarization module that extracts the most important points from the conversation. The summary should be concise and provide an overview of key topics, decisions made, and follow-up actions.
3. **Machine Translation for Multilingual Meetings**: Implement a translation module that allows real-time translation of meeting discussions into multiple languages. This feature should enable seamless communication in multilingual teams by translating speech or text-based discussions while preserving context and meaning.
4. **Context-Aware Action Item Extraction**: Enable the system to identify and track action items from the discussion. The assistant should recognize commitments like "I will send the report by Friday" and automatically assign tasks to relevant participants.

This project challenges you to apply NLP, speech processing, and context awareness to create a useful AI tool for workplace productivity. You are encouraged to design a dataset and explore innovative features to enhance the system's effectiveness.

Reference

1. Tan, Haochen, et al. "Reconstruct Before Summarize: An Efficient Two-Step Framework for Condensing and Summarizing Meeting Transcripts." Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023.
2. Wu, Han, et al. "VCSUM: A Versatile Chinese Meeting Summarization Dataset." Findings of the Association for Computational Linguistics: ACL 2023.
3. H. Zhang, P. S. Yu, and J. Zhang, 'A systematic survey of text summarization: From statistical methods to large language models', arXiv preprint arXiv:2406. 11289, 2024.
4. Park, Chanjun, et al. "BTS: Back TranScription for speech-to-text post-processor using text-to-speech-to-text." Proceedings of the 8th Workshop on Asian Translation (WAT2021). 2021.

Topic 6 - Open Your Mind

This is a highly flexible project where you can explore any idea involving language models that interest you, provided it connects to course concepts. We strongly recommend experimenting with cutting-edge LLMs (GPT, DeepSeek, Llama, etc.) through API integrations or local deployment. For example, you could consider combining multiple AI approaches (e.g., RAG + fine-tuning) or addressing ethical dimensions (bias detection, fact-checking). We recommend doing some fun things. Just open your mind and take a try. **Please contact TAs to verify your own selected topic before starting the group work**.