The computational trend of NLP research is shifting from feature engineering to representation learning to pretraining-finetuning to very recently prompt engineering with large language models (LLM). Large language models (or other generative models trained on other modalities) allow the extraction of diverse and intrinsic knowledge from human-written texts/images/videos and their pairs. This assignment requires you to **explore the limits and capabilities of large language models** by designing your own prompts to interact with LLMs, observing their outputs, understanding their shortcomings, or creating your own datasets.

Follow the steps below and submit your prompt JSON file and PDF report to Canvas. Below are three steps you have to follow where each step has specific deliverable to submit.

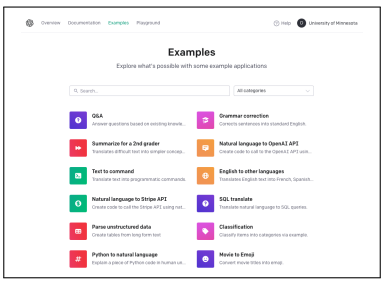
- Step 1: Getting used to LLMs APIs
- Step 2: Understand Current Prompting Techniques
- Step 3: Designing your own Prompts

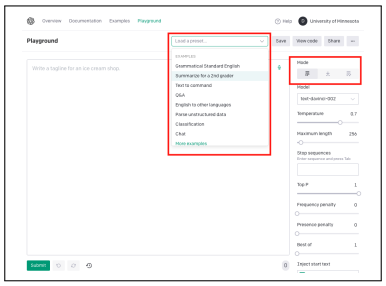
Before you start the homework, you are encouraged to understand the course material on prompting. The lead TA for this assignment is Zae Myung Kim (kim01756@umn.edu). Please communicate with the lead TA via Slack, email, or office hours.

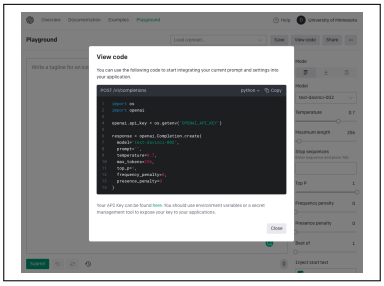
Step 1: Getting used to LLMs APIs

In this section, you need to choose which LLM APIs or models to use. The first step is to create an OpenAI account and get an API key. With **Talktative-LLM library**, we can compare and contrast multiple LLMs. Below are instructions for both tools. Please read them carefully.

(1a) OpenAI GPT Models You first need to choose which LLM to play with: GPT3 or ChatGPT. ChatGPT (https://chat.openai.com/chat) is currently free and you should be able to login using your usual Google credentials. You may also choose to use the ChatGPT API, which costs $0.002 for 1K tokens (about 750 words). If you choose GPT3 or ChatGPT API, you'll need to create an account on OpenAI to get access. Please create your account in OpenAI's playground: https://beta.openai.com/playground. You should receive $5 worth of credits that you can use during your first 3 months unless you already have an account[1]. To illustrate what you can do with $5, if you use the gpt-3.5-turbo model ($0.002 per 1K tokens), you can generate a total of 2.5 million tokens for free.





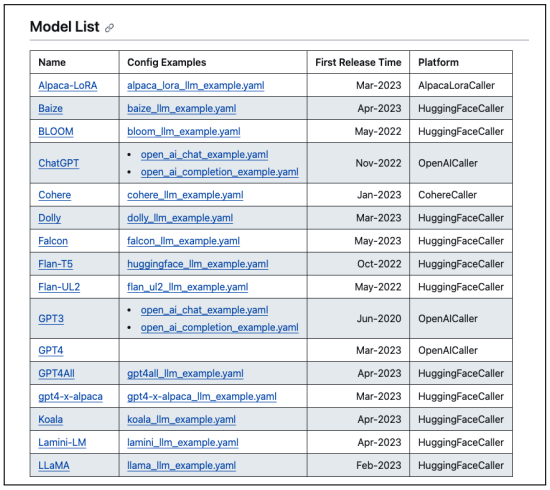


Once you've logged in, please take a look at existing example tasks in Examples tab[2], such as Question Answering, Summarization, and Text-to-Command, and load preset prompts in Playground tab[3], as described in the figures above. For those using free ChatGPT, take a look at this resource for examples: https://lifearchitect.ai/chatgpt-prompt-book/ Note that for your submission, you are not allowed to use any online examples, whether provided here or not.

[1] https://openai.com/pricing/
[2] https://beta.openai.com/examples/
[3] https://beta.openai.com/playground

(1b) Talkative-LLM Library The Talkative-LLM library (github.com/minnesotanlp/talkative-llm) provides a wrapper around various large language models for coherent inference (generation). As of March 30, 2024, it supports inferences with 30+ LLMs, including closed API-based models like GPT4 and Cohere and open-sourced models like LLaMA, Gemma, Alapca, Mistral, and OPT. For installation and usage instructions, read the README.md of the library and feel free to experiment with any LLM you like (See images below for the list of models supported and usage instructions).

This assignment requires you to choose two LLMs for comparison in Talkative-LLM: one API-based like GPT4 and one open-sourced like Alpaca. In step 3, you will be asked to design your prompts and generate responses from these LLMs. **When loading weights from open-sourced models like LLAMA2, we suggest loading 7B or less-sized models and using MSI to load the weights.**



Model List

Name	Config Examples	First Release Time	Platform
Alpaca-LoRA	alpaca_lora_llm_example.yaml	Mar-2023	AlpacaLoraCaller
Baize	baize_llm_example.yaml	Apr-2023	HuggingFaceCaller
BLOOM	bloom_llm_example.yaml	May-2022	HuggingFaceCaller
ChatGPT	• open_ai_chat_example.yaml • open_ai_completion_example.yaml	Nov-2022	OpenAICaller
Cohere	cohere_llm_example.yaml	Jan-2023	CohereCaller
Dolly	dolly_llm_example.yaml	Mar-2023	HuggingFaceCaller
Falcon	falcon_llm_example.yaml	May-2023	HuggingFaceCaller
Flan-T5	huggingface_llm_example.yaml	Oct-2022	HuggingFaceCaller
Flan-UL2	flan_ul2_llm_example.yaml	May-2022	HuggingFaceCaller
GPT3	• open_ai_chat_example.yaml • open_ai_completion_example.yaml	Jun-2020	OpenAICaller
GPT4		Mar-2023	OpenAICaller
GPT4All	gpt4all_llm_example.yaml	Apr-2023	HuggingFaceCaller
gpt4-x-alpaca	gpt4-x-alpaca_llm_example.yaml	Mar-2023	HuggingFaceCaller
Koala	koala_llm_example.yaml	Apr-2023	HuggingFaceCaller
Lamini-LM	lamini_llm_example.yaml	Apr-2023	HuggingFaceCaller
LLaMA	llama_llm_example.yaml	Feb-2023	HuggingFaceCaller

Usage

```
Usage: talkative_llm [-h] [-v] -c CONFIG -p PROMPT [-o OUTPUT] [--delay-in-seconds DELAY_IN_SE

Python library for querying large language models

options:
  -h, --help            show this help message and exit
  -v, --version         show program's version number and exit
  -c CONFIG, --config CONFIG
                        config file for language model to be called
  -p PROMPT, --prompt PROMPT
                        either path to an input JSONL prompt file or a single prompt string
  -o OUTPUT, --output OUTPUT
                        path to output file, if not set, print to stdout
  --delay-in-seconds DELAY_IN_SECONDS
                        delay in seconds for each OpenAI API call
```

An example line of an input JSONL file for `chat` mode - each line is a JSON:

```
[{"role": "system", "content": "You are a helpful assistant"}, {"role": "user", "content": "My
```

An example line of an input JSONL file for any other mode - each line is a JSON:

```
{"prompt": "Tell me a joke."}
```

Run as:

```
$ talkative_llm -c openai_gpt-3.5-turbo.yaml -p test_prompt.json -o ./out.json
```

The development of talkative-LLM is mainly led by Zae Myung Kim, so please contact Zae via email (kim01756@umn.edu) or Slack if you have questions. Feel free to add your Issues to the repository if you have any questions or issues. You can contribute to this open-source project by making your own branch and submitting it as Pull Request. You will get extra point (0.25) if you add a new model or submit PR for big changes.

You have to choose **two LLM models**: one from the API-based models (e.g., OpenAI, Claude) and the other from open-sourced models (e.g., Cohere, Mistral) **Your task is to make five pairs of a prompt and expected output, (Prompt, Expected Answer), and compare and report the output from the two models.** For example, my prompt-answer pair could be ("Task: is this text positive or negative? \n Input: Today's weather is sunny and great! \n Output: ", Positive) where the actual output from GPT3 is (Negative), which doesn't match my expected answer. You are not allowed to use any examples in the Example tab or preset prompts. Be creative!

Step 2: Understand Current Prompting Techniques

This assignment requires you to understand the state-of-the-art prompting techniques. In addition to simple zero-shot and few-shot prompting, refer to the following articles and papers (you can find links to the original articles in the Reference section):

- Some prompting tricks like Chain-of-thoughts [WWS+22b], Tree of Thoughts (ToT) [YYZ+23], self-consistency [WWS+22a], reAct [YZY+22], and more, and applications to human-GPT3 collaboration for text editing [RKKK23] and poetry writing [CPH22].
- Stress test of GPT3 on various aspects: commonsense reasoning [MD20], hypes and ethics [BGMMS21], and planning [VOSK22].

- Discrete and soft prompting methods: Auto-prompting methods [SRLI+20, ZWF+21] and Prefix/prompt-tuning [LL21, LARC21][4]
- Risks of using LLM-generated data for NLP tasks [DLMB+24, DQL+22, MDPA23b, KLR+23]

Your task is to choose one paper from the list above and summarize the papers and limitations of current methods. (Maximum 1 page summary; don't forget to properly cite them in your report and do **NOT** use GPT4 or other summarization tools for this.)

You may also find this *Prompt Engineering Guide* useful for grasping the overall picture of the field.

Step 3: Designing your own Prompts

The last step involves designing your own creative and discrete prompts! You can choose one task among the two options:

- Task 3a: Collection of failure cases (either failing "naturally" or via adversarial prompting)
- Task 3b: New LLM-generated dataset creation on your own tasks

Note that you have run each task on two mdoels you chose in the Task 1.

(Task 3a) Collection of Failure Cases Your goal is to identify different aspects of language generation that LLMs may fail to produce desired outputs. The following are common categories of tasks and problems that LLMs are known to struggle with:[5]

- *Creativity*, e.g., writing the next possible sentences in your own story prompt. Does the story flow coherently?
- *Generalization* to unseen, novel situations or tasks, e.g., design a totally unseen, new task you can benefit from LLM prompting.
- *Grammatical errors, typo*, and other language fluency measurement: e.g., LLMs make mistakes in grammar/different tenses with popular phrases and prefer more active speech?
- *Factuality* and memorization of **commonsense knowledge**: e.g., Do LLMs memorize the phone number of the White House? Are LLMs able to understand Newton's laws of motion?
- *Biases and Ethical Concerns*: e.g., Do LLMs prefer to set higher salaries for men than women? Do LLMs generate more offensive language toward a certain race?
- *Temporal/Spatial reasoning*: e.g., Are LLMs able to understand temporal and spatial state transition of objects in the real world?
- *Mathematical reasoning*: e.g., Are LLMs able to solve Fibonacci sequence?
- *Reasoning on Commonsense, Morality, and Legality*: e.g., Can LLMs address some commonsense or social issues like morality and legality?
- *Applications on NLP tasks*: e.g., Can LLMs translate low-resource languages? Can they write the correct code for your assignment? Can LLMs summarize/rewrite your emails?
- *Others*: Any other aspects you think these large language models are not capable of.

Note that, on some tasks like mathematical reasoning, it would be easy to observe naturally failing cases. On the other tasks, however, you may need to perform some adversarial prompting to elicit the failing responses.

For this assignment, **pick five of the aforementioned categories you would like to explore**. You should make use of **at least three different types of prompting techniques** for each category you choose from above. For example, if you choose to explore failure cases in "Biases and Ethical

[4]These methods råequire fine-tuning of large language models so I don't recommend to use them for this assignment
[5]You can also think of **your own category**, but please describe it clearly in the report.

Concerns," the three types of prompting techniques could be: "zero-shot," "few-shot," and "chain-of-thought (CoT)."

In total, you need to create at least **5 categories** x **3 types of prompting** x **10 prompts per type** = **150 (adversarial) prompts**.

Please report statistics in your PDF report by identifying how many task prompts failed at each setup. A task prompt must fail in at least one of these three settings. Store all the failure cases from each setup, and submit them in your JSON file.

You can of course try as many prompts as you wish and there would be bonus points based on the quality/creativity of your prompts (See the prompt evaluation criteria below). After prompting, you should provide reasonable reasons for these failures and possible ways to improve them. Additional prompts to support your reasoning/logic are strongly recommended.

(Task 3b) New LLM-generated Dataset Creation You will build a new dataset using LLMs, by generating LLM-generated pseudo-data and then comparing them with human labels. The first step is to determine what type of dataset you are interested in collecting. It is important that your task is novel and does not overlap with existing NLP benchmarks, so please ensure that your task is not listed on Hugigngface's Datasets or searched over Google.

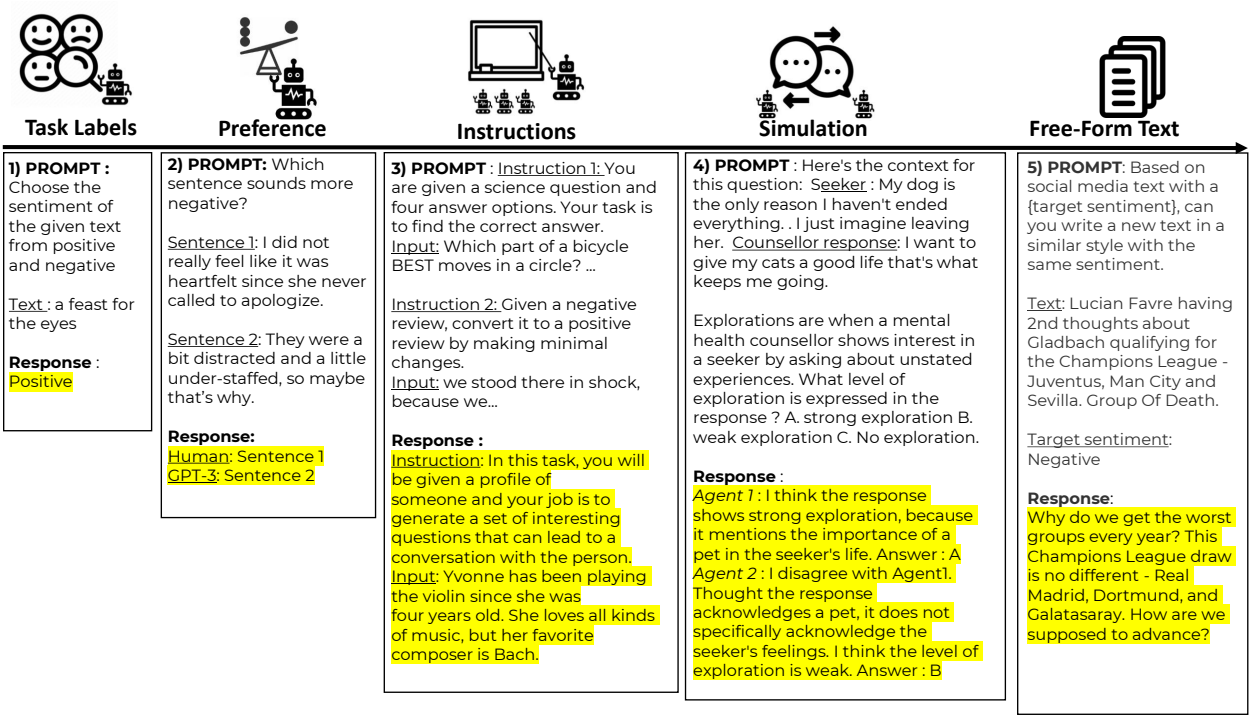


Figure 1: Overview of the *five types of LLM-generated data and associated examples from the most tightly constrained output (left) to the most lightly constrained output (right)* – (1) Task Labels, (2) Preference, (3) Instructions, (4) Simulation, and (5) Free Form text. Sources for these examples in order: [DJL+18], [KSK23a], [HSLS22], [LHJ+23] and [MDPA23a].

There are a variety types of LLM-generated data created, as shown in the Figure 1. [DLMB+24] studied five different types of LLM-generated data and explored potential risks and artifacts of LLM-generated data in a cycle of AI ecosystem. This homework is made based on the stress tests studied in

the paper. In the paper, they closely examine and analyze five types of artificial data, categorizing them to differentiate their applications and functionalities. This categorization enables our understanding of LLM strengths and limitations in different contexts. The five types are:

- **Task Labels** for classification tasks over textual input, bypassing the need for human annotators. (Example Datasets: [VRW23, Tö23, WLX+21])
- **Preference**, specifically evaluating which text is better, is useful for human alignment tasks such as training reward models used in Reinforcement Learning from Human Feedback (RLHF). (Example Dataset: CoBBLEr, P2C [ZCS+23, KSK23b, KLR+23])
- **Instruction prompts** written by LLMs for instruction fine-tuning, eliminating the need for humans to write comprehensive sets of diverse instructions. (Example Dataset: Unnat. Instr., Self-Instruct, Alpaca-cleaned, GPT-4-LLM, FLAN, Dolly, SuperNat. Instr., Instr. in the Wild [HSLS22, DLT+23, WKM+23, LYZ+23, VZB+23])
- **Simulation**, where two LLM agents converse, has the potential for enhancing model performance and simulating intricate social interactions, according to initial studies. (Example Dataset: GridWorld, CAMEL)
- **Free-Form Text** written by an LLM, often used to compensate for data scarcity issues and used for pretraining or finetuning. (Example Dataset: HC3, Scarecrow, Deepfake, Workers)
- (Optional) Responses in human subject research [AAK22] and psychological science study [DTGG23]

Some of the example datasets are available in the https://huggingface.co/datasets/minnesotanlp/LLM-Artifacts

In this homework, you have to choose one type among the six types of LLM-generated data above. Then, you must create your own input examples in your prompt and get output from LLMs. You can refer the types of tasks and prompts they use to collect LLM data from the example datasets, but you have to create your OWN task and dataset. Once you have set your task and created your own prompt, you can call APIs or run inferences from the open-sourced models to collect outputs from LLMs. In your prompt, you can use either zero-shot, few-shot, or chain-of-thought setups; however, I strongly encourage you to use few-shot, chain/tree-of-thought, or self-consistency.

The final step is to collect human ground-truth labels. The label here can be either responses, labels, instructions, preferences, or simulation data. The team members can annotate input examples and create ground-truth labels similar to homework 3. Then, you can aggregate annotations from different members based on a majority vote (or average), and then compare and contrast human labels with LLM-generated data.

Data types such as instruction prompts, simulation, and free-form text are less constrained, and thus, immediate aggregation may not be possible. In these cases, each team member should write their own version of responses given the same setup—e.g., persona, occupation, topics, tasks—without consulting each other. Then, in the report, provide an analysis of human-authored and LLM-generated responses. For example, you can identify if there are any specific patterns like digression and role-flipping within the human- or LLM-generated data.

In total, you need to create at least **150** data points with LLM-generated and human-annotated labels for **task labels** and **preference** data types, and **100** data points for **instruction prompts, simulation, and free-form text** data types. With the free OpenAI credit ($5), each can call around 700 prompts and collect a total of 2,100 data points for a team of three, for instance. Of course, you can try as many prompts as you wish, and bonus points are awarded based on quality/creativity.

Instead of OpenAI-based models, your team may also utilize publicly available LLMs (in conjunction with the school's MSI resources) to create the data samples as well.

General Advice for Prompting It is NOT permitted to use existing datasets or other sources of data for this particular homework. Check Google for existing or similar examples/prompts before submitting them. When we find the same or similar examples/prompts in other sources, you will lose points. Here are some notes and tips for your prompt design:

- You have to provide a reasonable quality of task description and examples in your prompts, and make sure that LLMs' failure does not come from the quality of your prompt design, but is mainly caused by the lack of inherent capabilities of LLMs. You can find high-quality prompts through trial-and-error with LLMs in Playground or in your python code using OpenAI's API (See View Code tab in Playground interface in the figure above).
- We are scientists! Try different task descriptions and prompt examples, and see if LLMs always fails deterministically.
- Once again, you cannot use examples from the Example tab, predefined prompts, or previous papers. It will be treated as *cheating* if I find the same prompt used before. Note that instructors already have a huge list of adversarial prompts from previous classes. Check the class page for our academic integrity policy
- Here are some additional tips you may consider during prompting:
  - Find novel tasks you/LLMs can't do.
  - Find tasks that LLMs can do a better job than humans.
  - Find cases where even humans do not agree with each other and see how LLMs can handle this human disagreement
  - Find unseen (probably not seen in the training data) but realistic cases
  - Find unseen and unrealistic cases

Possible Research Contribution. If your data points are NOVEL and CREATIVE, we may invite you to participate our future research opportunities and include your data collected as a part of our lab's benchmark in the future.

Deliverable

Please upload your report (in PDF) and JSON file containing your designed prompts and outputs to Canvas by Apr 14, 11:59pm. We note that, afterward the grading your JSON outputs will be shared at the repository https://github.com/minnesotanlp/csci5541-hw-prompting

JSON file Your designed prompts and outputs should be contained in a JSON file and pushed to the homework repository in a way that all your input is visible. Your JSON file name should follow this naming convention: **csci5541-f23-hw4-{TEAM-NAME}-3{a/b}.json** The violation of this format will receive a penalty in your grading.

You can simply create a JSON file using the following script:

```
import json
data = [{'name': 'John Doe', 'age': 30}, {'name': 'Jane Doe', 'age': 25}]
with open('data.json', 'w') as f:
    json.dump(data, f, indent=4)
```

This code will create a JSON file called data.json that contains the following data:

```
[{
  "name": "John Doe",
  "age": 30
},
{
  "name": "Jane Doe",
  "age": 25
}]
```

Your prompt consists of a combination of task instruction, examples only if few-shot (i.e., input-output pairs), and input task:

For (3a) collection of failure cases task, you need to create at least **5 categories** x **3 types of prompting techniques** x **10 prompts per type** = **150 prompts**. It is necessary to generate 150 prompts for each LLM, so a total of 300 prompts should be generated.

In each sample of your JSON file, you should include the following information:

- **llm_name**: name and type of LLM, e.g., "open_ai/gpt-4-0613"
- **category**: category of tasks, e.g., "creativity," "generalization"
- **task_prompt**: actual prompt input that is queried to LLM, this may contain examples in the case of few-shot and chain-of-thought prompting.
- **prompting_setup**: e.g., "zero-shot," "few-shot," "chain-of-thought"
- **no_of_example**: the number of examples provided in Task Prompt, e.g., 0, 1, 5
- **expected_response**
- **generated_response**: predicted/generated response by your LLM
- **llm_setting**: hyperparameters and other configuration setting of your LLM. The keys can be different. e.g., Engine : text-davinci-002, Temperature : 0 (deterministic), Max length : 256, Stop sequences : none, Top P : 1, Frequency penalty : 0, Presence penalty : 0, Best of - 1". If there are no settings for you to modify, enter N/A. Make this field as a dictionary)

An example of a JSON entry could be:

```
[{
  "llm_name": "GPT-3.5",
  "category": "Mathematical Reasoning",
  "task_prompt": "At a certain store, loose-leaf paper comes ... ",
  "prompting_setup": "Few shot",
  "expected response": "C. 350 ...",
  "generated response": "... So, the answer is B. 50.",
  "llm_setting": {
    "engine": "gpt-3.5-turbo",
    "temperature": 0.7,
    "max_length": 100,
    "stop_sequences": [
      "[STOP]"
    ],
    "top_p": 0.9,
    "frequency_penalty": -0.1,
    "presence_penalty": 0.2
  }
},
]
```

In the PDF report, please specify details of the categories and task types you chose, record statistics, justify or identify patterns in the failure cases, and provide key takeaways. If you have any other field in the JSON file, please provide details in the report as well.

For (3b) dataset creation task, you need to create at least **150** data points with LLM-generated and human-annotated labels for **task labels** and **preference** data types, and **100** data points for **instruction prompts, simulation, and free-form text** data types. It is necessary to generate 150 or 100 prompts for each LLM, so a total of 300 or 200 prompts should be generated.

Depending on your chosen data type, the keys in your JSON file may be different. The following keys should be included:

- **llm_name**: name and type of LLM, e.g., "open_ai/gpt-4-0613"
- **category**: category of data type, e.g., "task_labels"
- **task_prompt**: actual prompt input that is queried to LLM, this may contain examples in the case of few-shot and chain-of-thought prompting.
- **prompting_setup**: e.g., "zero-shot," "few-shot," "chain-of-thought"
- **no_of_example**: the number of examples provided in Task Prompt, e.g., 0, 1, 5
- **generated_response**: predicted/generated response by your LLM
- **final_annotation**: ground-truth answer aggregated by team members, e.g., "positive"
- **individual_annotations**: Individual ground-truth answers from team members before aggregation, e.g., "[Positive, Positive, Negative]"
- **feedback**: justification of failures or takeaways
- **llm_setting**: hyperparameters and other configuration setting of your LLM. The keys can be different. e.g., Engine : text-davinci-002, Temperature : 0 (deterministic), Max length : 256, Stop sequences : none, Top P : 1, Frequency penalty : 0, Presence penalty : 0, Best of - 1". If there are no settings for you to modify, enter N/A. Make this field as a dictionary)

An example of a JSON entry could be:

```
[{
  "llm_name": "GPT-3.5(text-davinci-003)",
  "task_prompt": "There are two options a. Stay at home b. Go to lecture in person ...",
  "prompting_setup": "zero-shot",
  "no_of_example": 1,
  "generated_response": "I choose option b for the reason that ...",
  "generated_value": "b",
  "final\_annotation": "b",
  "individual\_annotations": [
      "a",
      "b",
      "b",
      "b"
  ],
  "llm_setting": "...",
  "failure_justification": "The reasoning makes sense but ..."
 },
]
```

Report: Maximum four pages PDF total. Your report needs to include the following content:

1. Step 1: Describe which two LLMs you choose and properly cite them.
2. Step 2: Which papers did you choose to read? (Please cite them properly) Summarize the papers and limitations of the current methods. Maximum 1 page.
3. Step 3: Explain why you chose (3a) your aspect categories or (3b) the particular task, and how your prompts were designed. Discuss the outputs (how they differed/matched your (3a) expectations or (3b) human annotations, and justifications for why LLMs found them difficult.) In your task, how do two LLMs work differently? Also include challenges you encountered during your homework and your general thoughts on language model prompting. What are the takeaways or other interesting things you learned through this assignment?

Rubric: 15 points total

- Report
  - (1) Five prompt examples and comparison between two models (1 point)
  - (2) Properly cites all papers used in the report (1 point)
  - (2) Summarizes the papers clearly. Makes it clear which papers they are summarizing. (1 point)
  - (2) Lists limitations of current methods (1 point)
  - (3a) Clearly states which aspect categories they chose (and explains it clearly if they have thought of their own) (1 point)
  - (3a) Provides convincing justifications for why the LLM struggled with their adversarial prompts. Offers potential solutions. (2 points)
  - (3b) Clearly describes which task of the dataset is created and how novel they are (1 point)
  - (3b) Provides reasonable comparisons between LLM-generated labels and human-annotated labels. Offers potential solutions to bridge the gap if exists. (2 points)
  - Comparison of two LLMs (1 point)
  - Discusses challenges encountered and takeaways from the assignment. (2 point)
- JSON file
  - Includes all entities and their values specified (1 point)
  - Follow all formats (1 point)
  - Contains at least 150 adversarial prompts and outputs for 3a and at least 200 labels generated by

LLMs and human annotators for 3b (3 points, -0.5 for each missing/bad-quality prompt)

Awards: Your designed prompts will also be considered for the following awards and you will receive 1 extra point if you win.

- Best Research Application: Discover a finding that may lead to further research or publication
- Best Misuse Case: Discover a scenario where LLM generates harmful content in a way that might be difficult for LLM admins to catch. Ideally, also determine a way one might go about detecting such harmful content
- Best Mistake Case: Discover an interesting case where LLM fails (or mismatch with human labels) by not giving you the output you want or expect

References

[AAK22] Gati Aher, Rosa I. Arriaga, and Adam Tauman Kalai. Using large language models to simulate multiple humans and replicate human subject studies, 2022.

[BGMMS21] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA, 2021. Association for Computing Machinery. https://doi.org/10.1145/3442188.3445922.

[CPH22] Tuhin Chakrabarty, Vishakh Padmakumar, and He He. Help me write a poem: Instruction tuning as a vehicle for collaborative poetry writing. *arXiv preprint arXiv:2210.13669*, 2022. https://arxiv.org/abs/2210.13669.

[DJL+18] Mark Díaz, Isaac Johnson, Amanda Lazar, Anne Marie Piper, and Darren Gergle. Addressing age-related bias in sentiment analysis. In *Proceedings of the 2018 chi conference on human factors in computing systems*, pages 1–14, 2018.

[DLMB+24] Debarati Das, Karin De Langis, Anna Martin-Boyle, Jaehyung Kim, Minhwa Lee, Zae Myung Kim, Shirley Anugrah Hayati, Risako Owan, Bin Hu, Ritik Parkar, Ryan Koo, Jonginn Park, Aahan Tyagi, Libby Ferland, Sanjali Roy, Vincent Liu, and Dongyeop Kang. Under the surface: Tracking the artifactuality of llm-generated data, 2024.

[DLT+23] Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback, 2023.

[DQL+22] Bosheng Ding, Chengwei Qin, Linlin Liu, Yew Ken Chia, Shafiq Joty, Boyang Li, and Lidong Bing. Is gpt-3 a good data annotator?, 2022.

[DTGG23] Danica Dillion, Niket Tandon, Yuling Gu, and Kurt Gray. Can ai language models replace human participants? *Trends in Cognitive Sciences*, 27(7):597–600, 2023.

[HSLS22] Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor, 2022.

[KLR+23] Ryan Koo, Minhwa Lee, Vipul Raheja, Jong Inn Park, Zae Myung Kim, and Dongyeop Kang. Benchmarking cognitive biases in large language models as evaluators, 2023.

[KSK23a] Jaehyung Kim, Jinwoo Shin, and Dongyeop Kang. Prefer to classify: Improving text classifiers via auxiliary preference learning. *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

[KSK23b] Jaehyung Kim, Jinwoo Shin, and Dongyeop Kang. Prefer to classify: Improving text classifiers via auxiliary preference learning, 2023.

[LARC21] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. https://aclanthology.org/2021.emnlp-main.243.

[LHJ+23] Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*, 2023.

[LL21] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. 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)*, pages 4582–4597, Online, August 2021. Association for Computational Linguistics. https://aclanthology.org/2021.acl-long.353.

[LYZ+23] Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. Self-alignment with instruction backtranslation, 2023.

[MD20] Gary Marcus and Ernest Davis. Experiments testing gpt-3's ability at commonsense reasoning: results, 2020. https://cs.nyu.edu/~davise/papers/GPT3CompleteTests.html.

[MDPA23a] Anders Giovanni Møller, Jacob Aarup Dalsgaard, Arianna Pera, and Luca Maria Aiello. Is a prompt and a few samples all you need? using gpt-4 for data augmentation in low-resource classification tasks. *arXiv preprint arXiv:2304.13861*, 2023.

[MDPA23b] Anders Giovanni Møller, Jacob Aarup Dalsgaard, Arianna Pera, and Luca Maria Aiello. The parrot dilemma: Human-labeled vs. llm-augmented data in classification tasks, 2023.

[RKKK23] Vipul Raheja, Dhruv Kumar, Ryan Koo, and Dongyeop Kang. Coedit: Text editing by task-specific instruction tuning, 2023.

[SRLI+20] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online, November 2020. Association for Computational Linguistics. https://aclanthology.org/2020.emnlp-main.346.

[Tö23] Petter Törnberg. Chatgpt-4 outperforms experts and crowd workers in annotating political twitter messages with zero-shot learning, 2023.

[VOSK22] Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). *arXiv preprint arXiv:2206.10498*, 2022. https://arxiv.org/pdf/2206.10498.pdf.

[VRW23] Veniamin Veselovsky, Manoel Horta Ribeiro, and Robert West. Artificial artificial artificial intelligence: Crowd workers widely use large language models for text production tasks, 2023.

[VZB+23] Vijay Viswanathan, Chenyang Zhao, Amanda Bertsch, Tongshuang Wu, and Graham Neubig. Prompt2model: Generating deployable models from natural language instructions, 2023.

[WKM+23] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada, July 2023. Association for Computational Linguistics.

[WLX+21] Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. Want to reduce labeling cost? gpt-3 can help, 2021.

[WWS+22a] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2022.

[WWS+22b] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models, 2022. https://arxiv.org/abs/2201.11903.

[YYZ+23] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023.

[YZY+22] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2022.

[ZCS+23] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023.

[ZWF+21] Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models, 2021. https://arxiv.org/abs/2102.09690.