Final Project Report

Team Name:Botzilla
Github repository link:COMPSCI-383-Spring2025/group-project-botzilla: umass-cs383-s25-group-project-Final_Project-1 created by GitHub Classroom

1. Problem Statement

- Goal:
  We aim to evaluate whether a large language model (LLM) like GPT-4o-mini can simulate the responses, particularly syntax and semantics, of characters from the TV show *Friends* without memorization (using the dialogues in the dataset itself). We use the dataset from the show *Friends* because we want to understand how well AI chatbots could serve as companions/friends.

- Inputs and Outputs:
  Input: A prompt template instructing the LLM to role-play a specific *Friends* character (e.g., "You are Chandler Bing from Friends. Respond naturally as you would in the show.").
  Output: A generated text response from the LLM that emulates the target character.

- Connection to Requirements:
  This project directly explores the limits of LLMs in character simulation, emotional reasoning, and trait generalization — key themes in modern AI. It aligns with the course objectives by applying prompt engineering, automated prompting via templates, and developing custom evaluation metrics (ROUGE, BERTScore, cosine similarity, and memorization) to empirically measure LLM performance.

2. Dataset

- Dataset Name and Source:
  Name: *Friends TV Show Transcripts*
  Source: Cornell Convokit Dataset - Friends Corpus
  The dataset has dialogue transcripts spanning over 60,000 dialogue lines from all 10 seasons of *Friends*, curated and structured for conversational analysis.

Attributes include:

        - season_id: Season number of the episode
        - episode_id: Episode number within that season
        - scene_id: Scene number within the episode
        - utterance_id: The order of the utterance in a given scene
        - speaker: Character who spoke the line (e.g., Ross, Rachel)
        - tokens: Approximate word count (or tokenized word units)
        - transcript: The actual dialogue line

- Dataset Statistics:
    - 67,373 total examples before cleaning
    - 51,297 examples after removing missing transcripts and filtering to main characters
    - 7 columns: all of type `object`
    - No duplicate rows detected
    - Missing values only in the `transcript` column (6,063 missing)
    - Input:
      - `tokens`: nested list of tokenized words per utterance
      - Average length: ~5–15 tokens
    - Output:
      - `transcript`: raw text of the spoken line
      - Variable length, full sentence or phrase
    - Speakers: filtered to 6 main characters from *Friends*
    - Speaker distribution ranges from ~7,500 to ~9,300 utterances per main character

- Dataset Creation or Changes:
    - Removed lines spoken by non-core characters
    - Filtered to include only lines with complete speaker and transcript data
    - For evaluation metrics - removed entire conversations with any characters other than the main Friends characters, kept the rows with empty tokens columns as it

represented the end of a conversation, and also removed conversations with less than 3 characters involved as they wouldn’t provide much context.

3. Prompt Methodology

- Prompt Template:

"You are {test['speaker']} from Friends. Respond as you would in the show."
We tried to keep our prompt as concise as possible to evaluate LLM’s ability to naturally stimulate syntax and semantics.

- Sample Input/Output Example:
    - Context:

Monica Geller: There's nothing to tell! He's just some guy I work with!
Joey Tribbiani: C'mon, you're going out with the guy! There's gotta be something wrong with him!

- Actual (Chandler Bing):

All right Joey, be nice. So does he have a hump? A hump and a hairpiece?

- GPT Reply:

“Yeah, like he probably thinks ‘a moment of silence’ is when you pause for applause at a bad magic show. I mean, he can’t just be a normal guy. That’s like a unicorn that doesn’t sparkle!”

- Sampling Parameters:

Parameter	**Value**
Model	`gpt-4o-mini`
Temperature	`1.0`
Top-p	`1.0`
Max Tokens	`150`
Stop Tokens	Not used

    - Temperature and top-p values were left at defaults to maximize creativity and variability.

    - Max token cap ensured concise responses typical of sitcom dialogue.

- API Call Description:
  We used the **OpenAI Python SDK** to access the **`gpt-4o-mini`** model. The API call included:
    - A `system` message defining the character role
    - A `user` message containing two lines of prior dialogue as context
    - The generated output was stored along with evaluation scores (cosine similarity, ROUGE, BERTScore).

---

4. Evaluation Approach

- Metrics Used:
    - Memorization: We want the LLM's intelligence to be able to generate novel outputs similar but not the same as the dataset. We check memorization by taking conversations including only the main Friends characters and with more than 3 speakers so that we have context and then testing the GPT response by calculating the cosine similarity between the GPT response and the actual dialogue in a particular conversation. We give it the context (the dialogue of the first two characters in the conversations) and ask it to respond as the third character.
    - Rouge and Bert scores:
    - We used Rouge and Bert upon our mentor's suggestion and realized these are key metrics to evaluate as we had learnt the importance of syntax and semantics in the NLP intro lecture. Thus, we realized that having the syntax and semantics be similar would be good and pretty sufficient metrics when evaluating LLM's ability to simulate characters.
    - Bert score calculates how similar the meaning is of the GPT response and the actual dialogue in the conversation.
    - Rouge score calculate n-gram overlap in the GPT response and the actual dialogue. We calculate rouge-1 (Overlap of individual words - unigrams), rouge-2 (Overlap of 2 words - bigrams), and rouge-L (Based on the longest common subsequence between generated and reference).

- Evaluation Process:
  We automated our evaluation through code by using cosine_similarity and libraries to get the Rouge and Bert scores, as well as perform the cosine_similarity analysis. We also used cosine_simiarity upon our mentor's suggestion as it measures the similarity

between two non-zero vectors.

- Strengths and Weaknesses:
  We originally were having difficulties understanding what would entail good metrics as well as how to proceed in formally evaluating the LLM. However, after our mentor's feedback, we believe that the metrics capture a lot of our goals in the problem statement now and we feel that the cosine_simarity is fitting for our empirical analysis.

---

5. Results

- Summary of Metrics:

METRIC	AVERAGE SCORE
Memorization (Cosine Similarity)	0.2190
ROUGE-1 F1	0.08424
BERTScore F1	0.83045

- Discussion:
  We feel the outputs are somewhat useful, but the LLM requires more training on parameters related to language syntax. Our dataset even after removing irrelevant conversations was pretty big and would take a long time to test. Thus, we ended up testing on a few (10) entries. This is how we would interpret our results though:
    - **Low Cosine Similarity**: Indicates that the LLM was not copying memorized responses from training data. Instead, it produced novel and varied outputs, demonstrating generalization capability.
    - **Low ROUGE-1 Scores**: Suggest limited **n-gram overlap** with the actual show transcripts. This means that the model **did not use many of the same words or phrases** as the reference and there is **low surface-level similarity**, even if the meaning might still be similar. Thus, the model may be producing **more creative or paraphrased** outputs rather than directly mimicking the reference.
    - **High BERTScore F1**: Reflects strong **semantic similarity**, even when word choices differ. This implies that generated responses stayed on-topic and captured the **intended meaning**, even if they weren't verbatim replications.

---

6. Feedback and Communication

- Feedback Received:
  The main points of feedback we got from our mentor check-in was about changing our dataset cleaning and preprocessing and our prompt methodology. He also gave specific feedback on metrics and evaluation processes. We were confused about how to evaluate whether the LLM is actually responding how we want it to and what metrics we could use to give that evaluation a number.

- How You Addressed It:
  We realized how we could refine our project and goals by focusing on full conversations rather than each entry (dialogues). We also calculated the BERTscore, rouge score and cosine similarity, the evaluation metrics we discussed with our mentor, Sam, and interpreted those results with respect to what our original goal was.

Final Self-Checklist

[Not counted within the report length]

Before submitting, make sure:

- ☑ ~~Problem Statement is clear and connects to project goals~~
- ☑ ~~Dataset is described with source, stats, and any changes~~
- ☑ ~~Prompt is described, and sample input/output is given~~
- ☑ ~~Sampling parameters and API usage are mentioned~~
- ☑ ~~Evaluation metrics are defined and explained~~
- ☑ ~~Evaluation process + strengths/weaknesses are described~~
- ☑ ~~Results are presented clearly and discussed~~
- ☑ ~~You addressed draft feedback from your mentor~~
- ☑ ~~Everything is proofread and easy to understand~~
- ☑ ~~The Github repository is updated with the latest version of the codebase~~
- □ **Finally, the report has been added to the team Github repository**

]