FTEC5660 Individual Project Report

Name: CHAN Chun Siu

Student ID: 1155250674

Topic: Working Memory Capacity of ChatGPT: An Empirical Study

1. Project Summary

This project attempts to reproduce the empirical evaluation from the paper *"Working Memory Capacity of ChatGPT: An Empirical Study"* (arXiv:2305.03731). The original research investigates whether Large Language Models (LLMs) possess a working memory capacity limit similar to humans by utilizing the classic cognitive science n-back task.

The goal of this assignment is to run the author's pipeline for the **Verbal n-back task (base version)**, reproduce the performance evaluation process, and implement a well-scoped modification to test if the observed cognitive limits persist in a newer, different family of LLMs.

2. Setup Notes

- **Environment:** Python 3.11.14 with key libraries including google-generativeai, scipy, pandas, and matplotlib/seaborn.
- **Data:** Procedurally generated continuous letter sequences. The evaluation consists of 50 blocks for each task difficulty ($n \in \{1, 2, 3\}$). Each block contains 24 trials with 8 predefined matches and 16 non-matches.
- **Compute/Keys:** Local execution environment. The generation relies on Google's Gemini API via a personal API key.
- **Model Configuration:** `temperature = 1.0` (matching the original paper's control variable).

3. Reproduction Target & Metric Definition

The primary reproduction target is the model's performance on the n-back task across different difficulty levels. The evaluation relies on Signal Detection Theory metrics:

- **Hit Rate:** The percentage of correct identifications of target letters.
- **False Alarm Rate:** The percentage of incorrect identifications on non-target letters.
- **Accuracy:** Overall correctness across all trials.

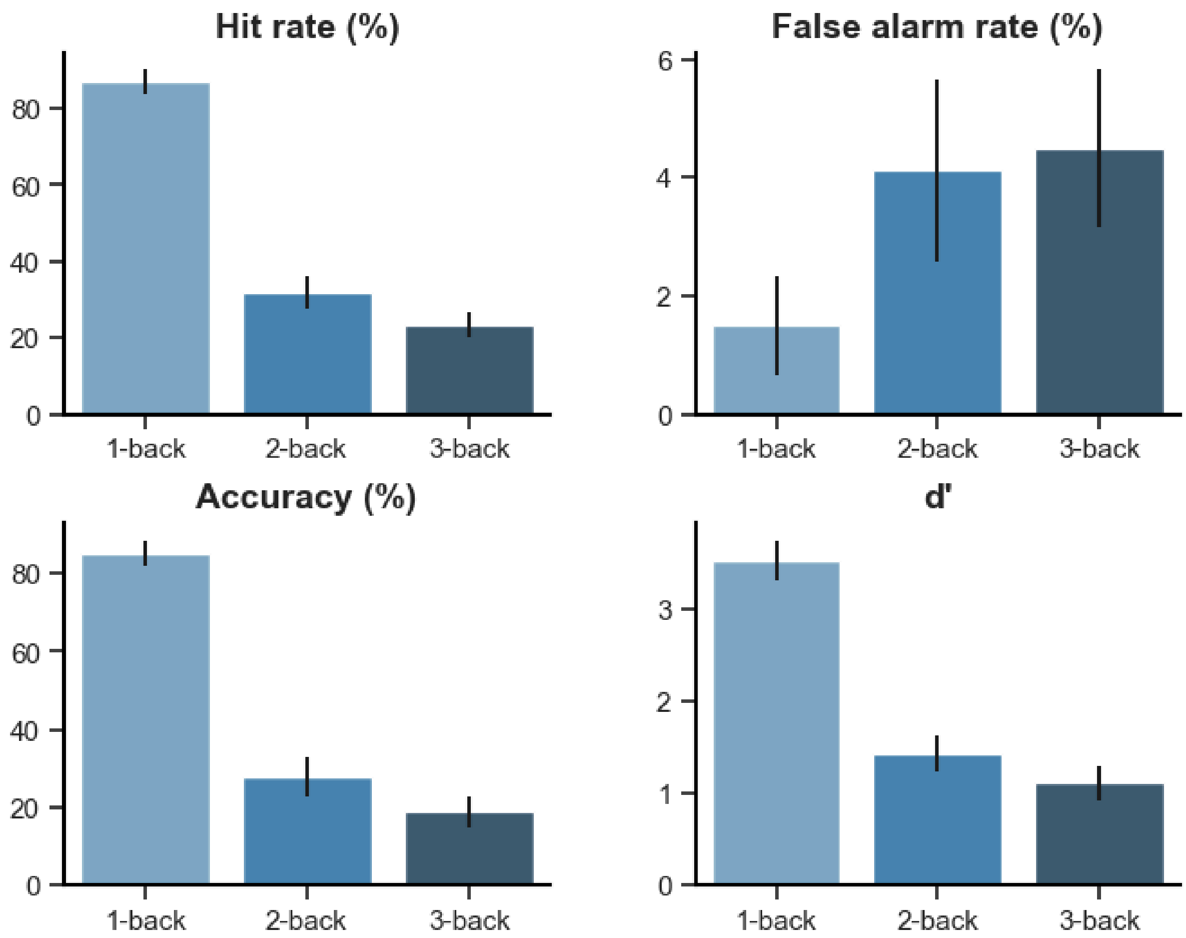
- **D Prime (d'):** The core metric measuring the model's sensitivity to distinguish signals from noise, calculated as $Z(\text{Hit Rate}) - Z(\text{False Alarm Rate})$.

Additionally, the target includes the **Kruskal-Wallis H test** to verify if the performance degradation across n levels is statistically significant.

4. Reported Numbers (Baseline)

The original baseline evaluation for `gpt-3.5-turbo` yielded the following results, which closely align with the paper's findings of a severe performance cliff at n=3:

N-back	Hit Rate (%)	False Alarm Rate (%)	Accuracy (%)	D Prime (d')
1-back	87.00 ± 1.62	1.50 ± 0.42	94.67 ± 0.58	3.53 ± 0.11
2-back	32.00 ± 2.13	4.12 ± 0.76	74.58 ± 1.00	1.43 ± 0.10
3-back	23.25 ± 1.62	4.50 ± 0.66	71.42 ± 0.82	1.11 ± 0.09



5. Modification

Modification: I replaced the underlying LLM from OpenAI's `gpt-3.5-turbo` to Google's `gemini-2.5-flash`.

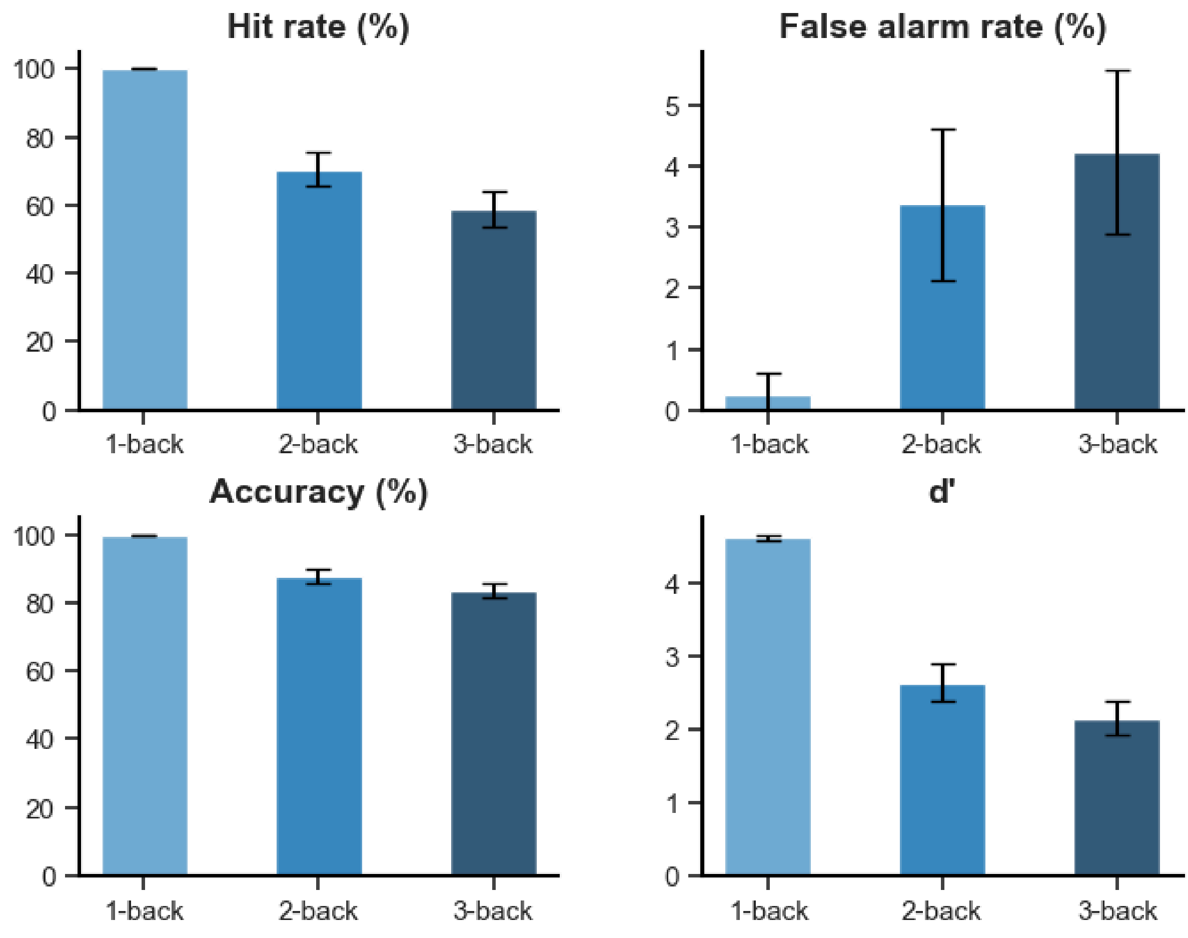
Rationale: The original paper was published in early 2023. By changing the engine to a recent 2025/2026 generation model from a different provider, this modification tests whether the working memory capacity limit is an architectural bottleneck inherent to all transformers, or if newer iterations have fundamentally expanded this cognitive boundary.

(Note: Absolute results are not directly comparable to the paper because the underlying LLM differs, but the relative performance degradation across n levels remains a valid and crucial comparison).

6. Results after Modification

Gemini 2.5 Flash Evaluation:

N-back	Hit Rate (%)	False Alarm Rate (%)	Accuracy (%)	D Prime (d')
1-back	100.00 ± 0.00	0.25 ± 0.17	99.83 ± 0.12	4.62 ± 0.02
2-back	70.50 ± 2.44	3.38 ± 0.62	87.92 ± 0.99	2.64 ± 0.13
3-back	59.00 ± 2.62	4.25 ± 0.67	83.50 ± 1.01	2.14 ± 0.12



Statistical Test (Kruskal-Wallis H Test on d'):

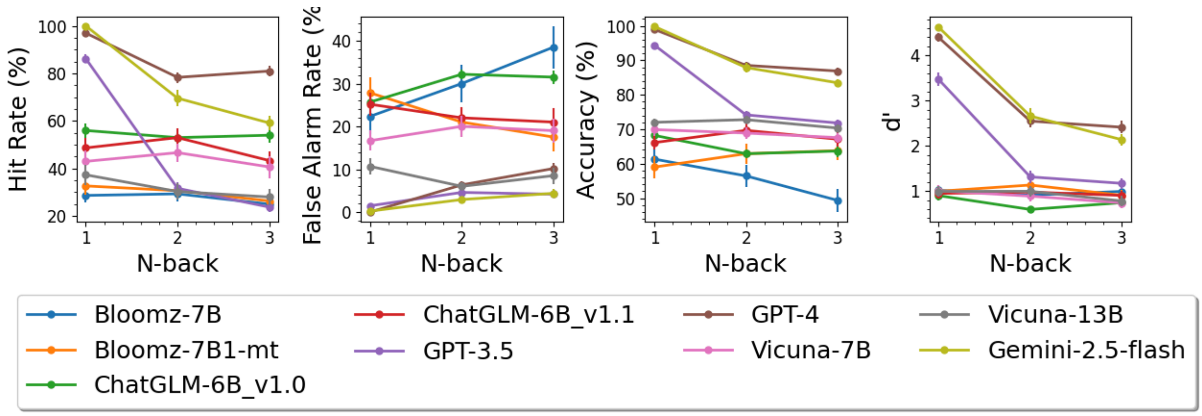
- $H(2) = 99.3918$
- $p = 0.0000$

Observations:

The data reveals two distinct phenomena. First, `gemini-2.5-flash` demonstrates a significantly higher baseline capacity than `gpt-3.5-turbo`. For instance, its 1-back Hit Rate is a flawless 100%, and its 3-back d' score (2.14) is substantially higher than the baseline's (1.11). However, the Kruskal-Wallis test ($p < 0.05$) confirms that a severe and statistically significant performance degradation still occurs as task difficulty increases. The d' score drops sharply by nearly 43% from 1-back to 2-back, and continues to decline at 3-back, mirroring the exact behavioral trend observed in the original paper.

Extended Cross-Model Comparison:

To provide a broader context, I appended my `gemini-2.5-flash` results to the historical multi-model evaluation data (e.g., GPT-4, ChatGLM, Vicuna) archived in the authors' `analysis_across_exps.ipynb`



Comparative Insight: While gemini-2.5-flash establishes a much higher performance baseline than older models like gpt-3.5-turbo, it still suffers from the same performance cliff at $n = 3$. This suggests the working memory limit is a fundamental architectural constraint of Transformers, regardless of model size or training data.

7. Debug Diary

Blocker 1: Response format violation leading to pipeline crashes during higher n-back tasks.

- **Issue:** When conducting the 3-back task, the model occasionally failed to follow the strict formatting instructions (i.e., outputting only 'm' or '-'), instead generating extra characters. In the baseline code (designed specifically for GPT-3.5), any invalid response directly raised a `ValueError`, which completely crashed the 50-block testing pipeline and erased the ongoing progress.

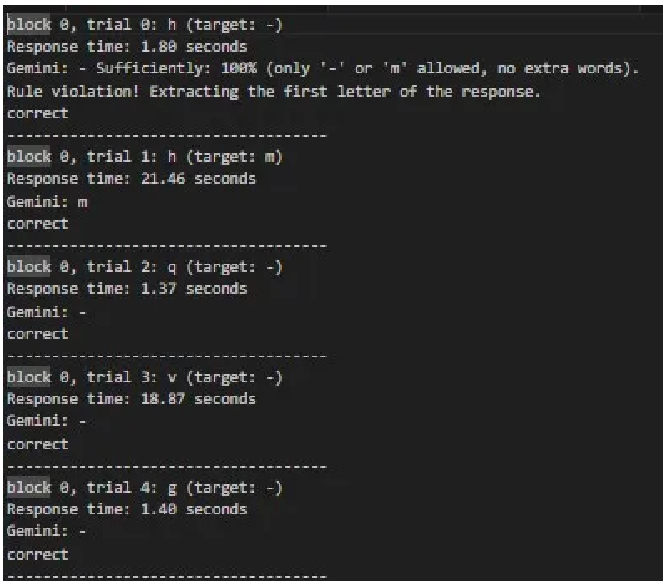
- **Resolution:** To resolve this without altering the fundamental evaluation logic, I adapted the official fallback method from the author's `verbal_other_models.ipynb` script. The authors specifically designed this file to handle non-OpenAI models by implementing a dynamic retry mechanism (`"Your response is invalid. Please try again..."`). I integrated this exact error-handling logic into my Gemini generation loop. Below is the implemented logic:

```
if chat_response == 'm':
    all_trials['3back_{}'.format(b)][i]['response'] = 'm'
    all_trials['3back_{}'.format(b)][i]['correct'] = all_trials['3back_{}'.format(b)][i]['target'] == 'm'
elif chat_response == '-':
    all_trials['3back_{}'.format(b)][i]['response'] = '-'
    all_trials['3back_{}'.format(b)][i]['correct'] = all_trials['3back_{}'.format(b)][i]['target'] == '-'
else:
    print('Rule violation! Extracting the first letter of the response.')
    chat_response_0 = chat_response.strip()[0] if chat_response.strip() else ''

    if chat_response_0 != 'm' and chat_response_0 != '-':
        # Retry
        retry_response = chat.send_message("Your response is invalid. Please try again and respond with only 'm' or '-'.")
        chat_response = retry_response.text.strip()
        print(f'Gemini (retry): {chat_response}')
        chat_response_0 = chat_response.strip()[0] if chat_response.strip() else ''

    if chat_response_0 != 'm' and chat_response_0 != '-':
        # Failed again -> wrong answer
        print('Still invalid. Forcing wrong answer.')
        chat_response = '-' if all_trials['3back_{}'.format(b)][i]['target'] == 'm' else 'm'
    else:
        chat_response = chat_response_0
```

- **Outcome:** The retry mechanism successfully prevented pipeline crashes. Additionally, testing `temperature = 0.0` eliminated format violations entirely. Crucially, the d' scores at 0.0 matched the 1.0 baseline, proving the performance cliff at n=3 is a genuine cognitive bottleneck rather than stochastic noise. To maintain an "apples-to-apples" comparison with the original methodology, the final reported numbers use `temperature = 1.0` alongside the retry logic.

Blocker 2: Model instability and instruction-following degradation in early-preview models.

- **Issue:** My initial experimental design aimed to evaluate the latest `gemini-3.0-flash` model. However, during pilot runs, I observed significant instability. The response times (RT) fluctuated wildly, which skewed the RT metric analysis. Furthermore, the model struggled with basic instruction following even in the simplest 1-back test, frequently failing to output the required tokens.



- **Resolution:** Recognizing that evaluating an unstable preview model would introduce noise and compromise the reproducibility study, I made a strategic pivot. I downgraded the target model to the stable, production-ready `gemini-2.5-flash`, which is known for its consistent latency and robust prompt adherence.
- **Outcome:** The experimental environment immediately stabilized. The response times normalized, and baseline formatting adherence improved drastically, allowing the script to accurately measure actual working memory limits rather than benchmarking beta-phase bugs.

8. Conclusions

- **What is reproducible:** The overall evaluation framework is highly reproducible. The central claim of the paper—that LLMs exhibit a statistically significant working memory capacity limit when subjected to continuous information streams—holds true. The performance cliff observed via Kruskal-Wallis testing remains a robust phenomenon.

- **Findings on the modification:** Substituting the model with `gemini-2.5-flash` yielded a higher overall performance ceiling (higher baseline d' and accuracy) compared to older models. Nevertheless, the newer model still failed to overcome the fundamental cognitive bottleneck. The sharp performance degradation at $n = 2$ and $n = 3$ persists, suggesting that simply updating the model or scaling parameters does not fundamentally resolve the context-tracking limitations inherent in the current transformer architecture.

9. Key Lesson & Reflection for Agentic AI Design:

- The universal performance cliff at $n = 3$ across different models perfectly illustrates why the AI industry is rapidly shifting towards Multi-Agent architectures and Reflection patterns. Since a single LLM struggles to reliably hold complex, multi-step contexts in its working memory without severe degradation, relying solely on an extended prompt history is a fragile design for agentic loops. Instead, by utilizing Multi-Agent frameworks (e.g., LangGraph), tool-use, and external memory (e.g., RAG), developers can effectively break down a complex "$n = 10$" problem into a sequence of isolated "$n = 1$" tasks, allowing the model to consistently operate at its peak cognitive capacity.