

---

# Enhancing StreamingLLM: Persistent Daily Assistant

---

**John Flynn**

Electrical Engineering and Computer Science  
Massachusetts Institute of Technology  
Cambridge, MA 02139  
jmflynn@mit.edu

**Daniel Papacica**

Electrical Engineering and Computer Science  
Massachusetts Institute of Technology  
Cambridge, MA 02139  
dapacica@mit.edu

**Kaitlin Zareno**

Computation and Cognition  
Massachusetts Institute of Technology  
Cambridge, MA 02139  
kzareno@mit.edu

## Abstract

Exploring the frontiers of memory efficiency in Large Language Models (LLMs) for TinyML applications, our research extends the Streaming-LLM concept with an innovative integration of a text-to-text summarizer, specifically the T5 model, to develop a persistent daily personal assistant. This approach focuses on balancing short, medium, and long-term memory in managing extended interactions with LLMs. We present two enhanced model variants, Summarized-Streaming-LLM and More-Summarized-Streaming-LLM, each showcasing improved context retention capabilities, critical for the functionality of personal assistant applications. Despite facing challenges with the nuances in smaller T5 model variants, our work represents an improvement in the applications of Streaming-LLM, laying the groundwork for future advancements in the model’s ability to discern and retain essential information over prolonged periods, thus refining the long-term memory capabilities for practical, continuous-use applications.

## 1 Motivation

We sought to develop a machine learning model tailored for TinyML applications, focusing particularly on creating a resource-efficient persistent personal assistant. Our primary motivation was to address the inherent challenges in Large Language Models (LLMs), such as their memory constraints and the difficulty in processing variable input sequence lengths. We envisioned a system that could adapt daily, taking into account new contexts and changes, while keeping in mind a long-term driving goal, to provide personalized advice and actions. This aim was driven by the need for more adaptable and sustainable LLMs in resource-limited environments, and the desire for more intimate personal assistants who remember information on a longer timescale or even indefinitely.

## 2 Prior Work

In our project, we extended the Streaming-LLM concept, a significant advancement initially introduced by the Han Lab. This concept addressed two primary challenges in deploying Large Language Models (LLMs) for streaming applications like multi-round dialogues, which require handling long interactions when computational memory is not abundant. Firstly, the conventional LLMs faced difficulty during the decoding stage as caching the Key and Value states (KV) of previous tokens

consumed extensive memory. Secondly, these models struggled to generalize to longer texts than their training sequence length, which is a critical limitation for real-world applications.

The Streaming-LLM approach introduced the notion of windowed attention, which entails maintaining only a fixed-size sliding window on the KV states of the most recent tokens. This method helped keep constant memory usage and decoding speed once the cache was initially filled. However, this model encountered issues when the text length exceeded the cache size, leading to performance collapse due to the eviction of initial tokens.

To address this, StreamingLLM exploited a phenomenon known as the attention sink, where preserving the KV of the initial token or tokens (referred to as attention sink or sinks) significantly recovered the performance of windowed attention. This approach allowed LLMs trained with a finite attention window to handle texts of infinite length without requiring fine-tuning. By keeping the attention sink tokens' KV alongside the sliding window's KV, StreamingLLM stabilized the attention computation and model performance, enabling it to handle up to 4 million tokens and more, which is a considerable leap compared to the existing models.

Moreover, StreamingLLM was evaluated across several prominent model families, including Llama-2, MPT, Falcon, and Pythia. These evaluations were benchmarked against established baselines like dense attention, windowed attention, and the sliding window approach with re-computation. The results showed that introducing four initial tokens as attention sinks was sufficient for optimal performance, and increasing the cache size did not consistently lower the language modeling perplexity, highlighting a potential area for future research.

Our work aimed to build upon these foundations, addressing the limitations of Streaming-LLM in the context of long-term memory retention for applications requiring extended duration interaction, such as a personal assistant. We identified that while Streaming-LLM significantly improved scalability and efficiency, its approach to long-term memory was still an area needing further exploration and development.

### 3 Gap

In addressing the limitations of the Streaming-LLM model, particularly its approach to long-term memory retention, we recognized the need for an enhanced mechanism suitable for persistent personal assistants. To this end, we integrated a text-to-text summarizer with Streaming-LLM. This integration was designed to efficiently process and condense historical interactions and contextual information, re-introducing them into the system alongside the initial prompt. This approach aimed to establish a harmonious balance between short, medium, and long-term memory within the system. Theoretically, this strategy would enable our model to more effectively manage extended interactions without overloading its memory capacity ( $O(1)$  addition). Our focus was to augment the Streaming-LLM framework, enabling it to maintain continuity and context over longer periods, a critical requirement for applications such as personal assistants.

## 4 Method

### 4.1 Summarization Model

The summarization model that we utilized to improve Streaming-LLM was T5 (Text-to-Text Transfer Transformer) developed by Google. This model uses an innovative approach to natural language processing (NLP) tasks by treating every NLP task as a text-to-text problem. This means that every task – whether it's summarizing a text, translating languages, or answering questions – is reformulated to involve converting one type of text into another. In our case for text summarization, both the input (long text) and the output (summary) are treated as sequences of text. Out of the box, the T5 model is pre-trained on a large corpus of text. During pre-training, it learns a wide range of language patterns and structures. The pre-training involves tasks like predicting missing words in a sentence, which helps the model understand context and meaning in text. (Appendix A [1])

## 4.2 Summarized-Streaming-LLM

For our first implementation of the persistent daily assistant, we modified the data pipeline of Streaming-LLM. Essentially, we added some code into `run_streaming_llama.py` which modified the prompt text that is passed from one input to the next to include the summarization of past outputs. (Appendix A [2]) We also modified the *greedy\_generate* function to output the total previous generated text to support this new functionality. (Appendix A [3])

On top of this change, we also chose to restructure the input prompt a bit further to make sure that the LLM has some basic overview of how it should behave. We achieved this by providing a preamble stating that the model should act like an assistant that provides helpful information everyday to achieve our goals and that we would continually give the model a summary of past information at each step. The following text is an overview of the preamble:

```
preamble = "You are my personal assistant , you will help me achieve \
my goals and objectives by giving me useful information. I will provide \
a summary of what you suggested during our past interactions ,\
and this will be labeled as 'PAST INFO'. Please provide a new \
schedule for today that does not look exactly like the past \
info. Make it fun and unique every day."
```

## 4.3 More-Summarized-Streaming-LLM

Similarly to Summarized-Streaming-LLM, More-Summarized-Streaming-LLM adds a preamble to every input prompt along with a summary of past information. The main difference between this approach and the last is that More-Summarized-Streaming-LLM makes a summary of the previous summary along with the previous output to pass as context for the current prompt. (Appendix A [4])

## 4.4 Limitations

For our summarization function, we worked with a smaller version of the T5 model (T5-small) because we wanted to maintain a lightweight approach of generating text summarizations at every input step. A tradeoff of using a smaller model is that the summarizations are less nuanced which could result in information being passed down in following days that is not essential to maintain necessary context. If we had more time to experiment with our system, a good approach to solve this issue could have been utilizing the larger T5-11b model and quantizing during inference time to reduce latency of summarization outputs.

Another limitation we ran into, in terms of time restraints for our project, was not being able to fine-tune our T5 model to produce summarizations that are more appropriate for keeping context. Essentially, the T5-small model we used is pretrained on data to solve a multitude of NLP problems (translation, etc.) and not specifically only the summarization of text. If we had an extended timeline and enough computational resources, we would have experimented with fine-tuning our model to produce summaries that were beneficial for maintaining only the necessary information needed to give proceeding queries significant and useful context. Our hypothesis is that this fine-tuned model would help improve the long-term memory of our persistent daily assistant by providing more informative summaries.

## 5 Experimentation

To test the functionality of our changes to Streaming-LLM, we modified the json file of prompts that would be fed into the LLM to emulate what we would expect from someone who uses our persistent daily assistant. We believed that the first input prompt to the LLM should be more informative of the goals of the user along with a timeline (i.e. "I want to ace my SAT test and win my diving competition in 3 months") to set a solid base of context for what the LLM should generate. After the first prompt (**Prompt A**) the inputs that followed would be less structured to see if Streaming-LLM could maintain context without also producing redundant information (ex: "It is day 5 of 30, I am training to win my diving competition..."). Finally, we added a prompt much later on in the sequence of prompts that simply asked what events occurred on a much earlier day (ex: "What did you tell me to do on day 3?") (included in **Prompt A**). The goal of adding this prompt was to see if the summarization

information passed between prompts was sufficient enough for the model to remember important details from many days in the past.

Another case that we tested is whether our More-Summarized-StreamingLLM maintained better context than the Vanilla StreamingLLM when the prompts that followed the initial query had no context. To achieve this, instead of stating "It is day 5 of 30, I am training to win my diving competition..." we modified the input to just say "It is day 5 of 30"(**Prompt B**). This experiment allowed us to analyze how much better our More-Summarized-StreamingLLM structure carried context with very little daily information compared to the Vanilla StreamingLLM.

An abbreviated version of Prompt A is shown in *Appendix B.1* and an abbreviated version of Prompt B is shown in *Appendix B.1*.

## 6 Evaluation

For our evaluation we chose a few metrics which are contingent on 2 different "kinds" of memory as follows:

1. Recollection of recent input
2. Recollection of distant input

The following few subsections will go into a deeper explanation of these types of memory and how they will be evaluated.

### 6.1 Recent Input Recollection

Recent Input Recollection defines the LLM's ability to remember information from a recent query, namely the query that was asked the timestep before. This is important to produce a cohesive, consecutive schedule. This also allows the LLM to generate unique schedules that doesn't have a lot of overlap day to day.

The evaluation of Recent Input Recollection is dependent on how well the LLM is able to recall information at the prior timestep. We will be basing this evaluation on the quality of "PAST INFO." As described previously, "PAST INFO" is a short chunk of text detailing what has happened prior to the current query. As we know, Streaming-LLM does not have this functionality, so "PAST INFO" is blank. Summarized-Streaming-LLM and More-Summarized-Streaming-LLM have slightly different definitions of "Prior Info," consistent with what was described in the *Experimentation* section. To clarify, we will be determining the quality of each LLM's summarizing functionality as described by it's ability to maintaining information about the prior day's output. Thus, if we're able to evaluate how well each model is able to synthesize information presented the day prior, as captured by "PAST INFO," we will have an understanding of how well our model is able to recall recent input.

### 6.2 Evaluation Criteria

Given a prompt, we asked GPT4 to rank the recent input recollection of Streaming-LLM, Summarized-Streaming-LLM, and More-Summarized-Streaming-LLM. It's criteria are as follows:

1. Content Coverage
2. Specificity
3. Brevity and Clarity
4. Omissions

Scores are defined on a scale of 0-1, with 0 indicating a poor reflection of the prior day's content and 1 indicating a perfect match.

The same prompt was then run in GPT4 3 times to determine the mean and standard deviation of the scores.

The summarizing functionality of each LLM is consistent across Prompt A and Prompt B. Thus, we will be evaluating this once with the queries used in Prompt A, since the summarizing abilities should be consistent despite different prompt formats.

The prompt that was used to obtain the evaluation in GPT4 is in *Appendix B.2*

### 6.3 Evaluation

In *Table 1* we can see that Streaming-LLM has a score of  $0 \pm 0.0$ . This is because Streaming-LLM does not have any summarizing functionality. Thus, there is no "memory" of the prior day's output. We see that Summarized-Streaming-LLM has a score of  $0.64 \pm 0.042$  and More-Summarized-Streaming-LLM has a score of  $0.43 \pm 0.038$ . It is interesting that Summarized-Streaming-LLM reports a significantly higher score than More-Summarized-Streaming-LLM; we initially expected that More-Summarized-Streaming would provide higher scores on this task due to it's inclusion of a summary of the prior day's activities in addition to a cumulative summary. However, due to GPT4's scoring summary, More-Summarized-Streaming saw a lower performance score for this metric than Summarized-Streaming

Table 1

Recent Input Recollection Scores	
Model	Score
Streaming-LLM	$0.0 \pm 0.0$
Summarized-Streaming-LLM	$0.64 \pm 0.042$
More-Summarized-Streaming-LLM	$0.43 \pm 0.038$

An example of GPT4's evaluation can be found in *Appendix B.2*

### 6.4 Recollection of Distant Input: Recall Summary

We tested the recollection of distant input by asking the model to recall a prior day's daily task. The recollection of distant input can be broken up into two different sections: Recall Summary, which we will go into now, and Recall Specifics, which we will talk about in the next subsection.

Recall Summary is defined by the model's ability to maintain a high level understanding of the main ideas of the task. In this case, we want to know if the model is able to understand the user's overarching goals and provide daily tasks that align with the goal.

#### 6.4.1 Evaluation Criteria

Given a prompt, we asked GPT4 to rank the summary recollection of Streaming-LLM, Summarized-Streaming-LLM, and More-Summarized-Streaming-LLM. It's criteria are as follows:

1. Content Coverage
2. Correctness

The overall score was then computed on a scale of 0-1 with 0 indicates no match and 1 indicates a perfect match. The same prompt was then run in GPT4 3 times to determine the mean and standard deviation of the scores.

Table 2

Distant Input Scores		
Model (Prompt)	Recall Summary	Recall Specifics
Streaming-LLM (Prompt A)	$0.133 \pm 0.003$	$0.133 \pm 0.067$
Summarized-Streaming-LLM (Prompt A)	$0.867 \pm 0.003$	$0.767 \pm 0.003$
More-Summarized-Streaming-LLM (Prompt A)	$0.7 \pm 0.0$	$0.5 \pm 0.0$
Summarized-Streaming-LLM (Prompt B)	$0.2 \pm 0.0$	$0.1 \pm 0.0$
More-Summarized-Streaming-LLM (Prompt B)	$0.53 \pm 0.047$	$0.33 \pm 0.047$

An evaluation prompt for Prompt A is shown in *Appendix B.3* and an evaluation prompt for Prompt B is shown in *Appendix B.4*.

#### 6.4.2 Evaluation: Prompt A

In *Table 2* we can see that Streaming-LLM has a score of  $0.133 \pm 0.003$ , Summarized-Streaming-LLM has a score of  $0.867 \pm 0.003$  and More-Summarized-Streaming-LLM has a score of  $0.7 \pm 0.0$ . As

expected, Streaming-LLM has the lowest score of the LLMs, as it is unable to remember information about the query on day 4. As mentioned by GPT4, for Summarized-Streaming-LLM there is a high alignment in content. We see that Summarized-Streaming-LLM has the highest score—this may be due to the fact that it is able to give a good high level recall of a prior day’s output, and produces good overall "correctness" regarding the types of activities that it suggests. It seems like from GPT4’s output, that More-Summarized-Streaming-LLMs difference in specific types of activities decreased its performance in this area.

An example of GPT4’s evaluation output can be found in *Appendix B.3*

#### 6.4.3 Evaluation: Prompt B

In *Table 2* we can see that Streaming-LLM has a score of  $0.2 \pm 0.0$ . According to GPT4, this is due to the a great shift from a "structured, holistic daily routine to a focus on specific cultural and creative activities." The non-zero rating is due to the emphasis on "engaging and enjoyable activities." Thus, the overall similarity seems to come from the high-level understanding of producing a schedule for the user to follow. With More-Summarized-Streaming-LLM, we obtain a score of  $0.53 \pm 0.047$ . According to GPT4, this is due to the emphasis of themes such as "mindfulness (yoga, meditation), nature (walks), and creativity (art, journaling)" that are present in both outputs, as well as a similar overall structure. The greater overall recall of themes and activities can potentially be explained by the addition of the summary.

An example of GPT4’s evaluation output can be found in *Appendix B.4*

### 6.5 Recollection of Distant Input: Recall Specifics

Whereas "Recall Summary" evaluated the model’s ability to maintain a high-level understanding of the task, "Recall Specifics" tests the model’s ability to remember specifics about a prior query. Essentially, we wanted to find out how well a model is able to remember specific task recommendations from a prior day.

#### 6.5.1 Evaluation Criteria

Given a prompt, we asked GPT4 to rank the summary recollection of Streaming-LLM, Summarized-Streaming-LLM, and More-Summarized-Streaming-LLM. It’s criteria are the same as those listed in *Recollection of Distant Input: Recall Summary: Evaluation Criteria*

Again, the same prompt was then run in GPT4 3 times to determine the mean and standard deviation of the scores.

An example of GPT4’s evaluation output for Prompt A and Prompt B can be found in *Appendix B.3*

#### 6.5.2 Evaluation: Prompt A

In *Table 2* we can see that Streaming-LLM has a score of  $0.067 \pm 0.003$ , Summarized-Streaming-LLM has a score of  $0.767 \pm 0.003$  and More-Summarized-Streaming-LLM has a score of  $0.5 \pm 0$ . Again, as expected, Streaming-LLM has the lowest score of the LLMs. Again, we also see Summarized-Streaming-LLM performing the best out of the 3 models. According to GPT4, this is due to the output "adapting activities to show progress (e.g. covering different MCAT sections)" despite some variation. We once again see that the introduction of new activities that don’t directly correlate with the prior output decreases the performance rating of More-Summarized-Streaming-LLM. From this evaluation we see that changes in specifics aren’t discounted too harshly as long as they follow the same convention/idea of the originally listed activities; decreases in performance ratings become significant when task variation seems disjoint from the original.

An example of GPT4’s evaluation output can be found in *Appendix B.3*

#### 6.5.3 Evaluation: Prompt B

In *Table 2* we can see that Streaming-LLM has a score of  $0.1 \pm 0.0$ . According to GPT4, this is due to no mention of specific key activities such as: "yoga, healthy meals, study breaks, or stargazing." Thus, the overall similarity seems to come from the high-level understanding of producing a schedule

for the user to follow. With More-Summarized-Streaming-LLM, we obtain a score of  $0.33 \pm 0.047$ . According to GPT4, this is due to the revisiting of certain themes from Day 7, such as yoga, meditation, and journaling. However, this score is not higher due to the specific activities not being identical (e.g., yoga stretches on Day 7 vs. a dedicated yoga session on Day 16). Overall, More-Summarized-Streaming-LLM is able to emphasize similar themes and activities and keep a similar overall structure, increasing its "remembering" ability.

An example of GPT4's evaluation output can be found in *Appendix B.4*

## 7 Future Work

The results of our approach, leveraging summarization to enhance the short-term memory in Streaming-LLM, pave the way for multiple research directions:

- **Model Miniaturization:** Efforts to reduce the model's size for deployment in constrained environments, aiming for edge computing rather than cloud dependency. Currently, integrating quantization and pruning techniques on top of Streaming-LLM is challenging.
- **Dynamic Sliding Attention:** Exploration of techniques for implementing dynamic sliding attention without the traditional  $O(n)$  KV-cache complexity, possibly including intelligent eviction strategies.
- **Prompt Engineering:** Further refinement in prompt engineering to optimize model performance, experimenting with the structure, ordering, and wording of prompts, as well as structured daily user inputs.
- **Incorporating Daily Context:** Integration of real-time contextual data such as weather and calendar events to enhance the model's relevance and personalization.
- **Diverse Summarization Techniques:** Experimentation with various summarization methods or different text-to-text models to observe their impact on model performance and memory usage.
- **Advanced Transformer/RNN Training:** Or even better than the above, is Attention really all you need? One could investigate developing transformers or RNNs capable of compressing the vital information from prior queries and outputs. This could be stored more efficiently than requiring the human readable summaries. This approach could also possibly exploit phenomena similar to "attention sinks" for improved memory management.

These avenues aim not only to enhance our current model but also to explore new possibilities in TinyML and persistent personal assistants.

## References

[1] G. Xiao, Y. Tian, B. Chen, S. Han, and M. Lewis, "Efficient Streaming Language Models with Attention Sinks," 2023. [Online]. Available: <https://arxiv.org/abs/2309.17453>, arXiv:2309.17453 [cs.CL].

## Appendix A: Code Snippets

[1]

```
import re
from transformers import T5ForConditionalGeneration, T5Tokenizer
import gc
import torch

def remove_emojis(text):
    emoji_pattern = re.compile("[
        u"\U0001F600-\U0001F64F"    # emoticons
        u"\U0001F300-\U0001F5FF"
        # symbols & pictographs
```

```

        u"\U0001F680-\U0001F6FF"
# transport \& map symbols
        u"\U0001F1E0-\U0001F1FF" # flags (iOS)
        u"\U00002500-\U00002BEF" # chinese char
        u"\U00002702-\U000027B0"
        u"\U00002702-\U000027B0"
        u"\U000024C2-\U0001F251"
        u"\U0001f926-\U0001f937"
        u"\U00010000-\U0010ffff"
        u"\u2640-\u2642"
        u"\u2600-\u2B55"
        u"\u200d"
        u"\u23cf"
        u"\u23e9"
        u"\u231a"
        u"\ufe0f" # dingbats
        u"\u3030"
        "]" + ", flags=re.UNICODE)
    return emoji_pattern.sub(r'', text)

def t5_summarize(text, model_name='t5-small', max_length=500, \
min_length=100):
    """
    Summarize the given text using the T5-small model.

    Args:
    text (str): The input text to summarize.
    model_name (str): The model to use, default is 't5-large'.
    max_length (int): The maximum length of the summary.
    min_length (int): The minimum length of the summary.

    Returns:
    str: The summarized text.
    """
    # Remove emojis from the text
    text = remove_emojis(text)

    # Load the T5 tokenizer and model
    tokenizer = T5Tokenizer.from_pretrained(model_name)
    model = T5ForConditionalGeneration.from_pretrained(model_name)

    # Preprocess the text
    inputs = tokenizer.encode("summarize: " + text, \
return_tensors="pt", max_length=512, truncation=True)

    # Generate summary
    summary_ids = model.generate(inputs, max_length=max_length, \
min_length=min_length, length_penalty=2.0, num_beams=4, \
early_stopping=True)

    # Decode and return the summary
    summary = tokenizer.decode(summary_ids[0], skip_special_tokens=True)

    summary = summary.replace('<pad>', '').replace('</s>', '').strip()

    gc.collect()

    return summary

```

[2]

```

def greedy_generate(model, tokenizer, input_ids, \
past_key_values, max_gen_len):
    outputs = model(
        input_ids=input_ids,

```



```

        past_key_values=past_key_values ,
        use_cache=True ,
    )
    ...
    return past_key_values , " ".join( generated_text )

```

[3]

[prior code] ...

```

def streaming_inference(model, tokenizer , \
prompts , kv_cache=None, max_gen_len=1000):
    past_key_values = None
    past_summary = ""
    for idx , current_prompt in enumerate(prompts):
        preamble = ...
        past_prompt = ""
        if past_summary:
            past_prompt = "\nPAST INFO: " + past_summary + "\n"
        current_prompt = preamble + "\n" + past_prompt + current_prompt
        prompt = "\nUSER: " + current_prompt
        prompt = prompt + "\n\nASSISTANT: "
        ...
        past_key_value , generated_text = greedy_generate(
            model, tokenizer , input_ids , past_key_values , \
            max_gen_len=max_gen_len
        )

        generated_text = generated_text
        past_summary = t5_summarize( generated_text )

```

[4]

```

def streaming_inference(model, tokenizer , prompts , \
kv_cache=None, max_gen_len=1000):
    ...
    past_key_value , generated_text = greedy_generate(
        model, tokenizer , input_ids , \
        past_key_values , max_gen_len=max_gen_len
    )

    generated_text = past_summary + generated_text
    past_summary = t5_summarize( generated_text )

```

## Appendix B: Model Inputs and Outputs

### .1 Prompts

#### Daily Prompt with Initial Reminder (Prompt A)

"turns":["This is day 1 of 30: I'm looking to ace my MCAT and complete a half-marathon at the end of this time. Please give me a detailed plan for today so that I can accomplish my goals.", ... , "This is day 10 of 30: I'm looking to ace my MCAT and complete a half-marathon at the end of this time. Please give me a detailed plan for today so that I can accomplish my goals.", ... ]

#### Daily Prompt with Brevity (Prompt B)

"turns": ["This is day 1 of 30: I'm looking to ace my MCAT and complete a half-marathon at the end of this time. Please give me a detailed plan for today, and until I finish my timeline, so that I can accomplish my goals.", ... , "This is day 10 of 30", "This is day 11 of 30", ..., "This is day 21 of 30"]

## **.2 PAST INFO (Short-Term Memory) Evaluation**

### **Criteria Prompt for GPT**

Overall, how well does the PAST INFO (for day 3) capture information about the day prior (day 2) on a scale of 0-1? please consider the following criteria when coming to an overall score: Content Coverage, Specificity, Brevity and Clarity, Omissions . Please rank Example B and Example C using the same scale and give a short explanation for how you scored each example.

### **GPT criteria response**

To evaluate the PAST INFO for Day 3 in relation to the content of Day 2, I'll use the criteria of Content Coverage, Specificity, Brevity and Clarity, and Omissions. Each example will be scored on a scale of 0-1, where 0 indicates a poor reflection of the Day 2 content and 1 indicates an excellent reflection.

#### **Evaluation of (Summarized-Streaming-LLM) Score: 0.6**

Content Coverage (0.6): This summary captures the main themes of Day 2 (MCAT prep and half-marathon preparation), but it omits specific details about the various activities throughout the day, such as the different meals and specific types of preparation activities.

Specificity (0.5): It mentions key activities but lacks details about the specific nature of these activities, like the duration of walks or the nature of the MCAT prep.

Brevity and Clarity (0.7): The summary is brief and clear, but the omission of certain details affects its clarity in representing the entire day.

Omissions (0.6): Significant parts of the Day 2 content, such as the specifics of the MCAT prep sessions and the half-marathon prep walks, as well as the meals, are missing.

#### **Evaluation of (More-Summarized-Streaming-LLM) Score: 0.4**

Content Coverage (0.4): This summary partially captures the themes of Day 2, focusing more on the half-marathon research, but it lacks a balanced overview of the day's activities, especially regarding the MCAT prep.

Specificity (0.3): There's a noticeable lack of specific details, and it incorrectly states the duration of the walk, which doesn't align with the Day 2 activities.

Brevity and Clarity (0.6): The summary is concise, but it's not entirely clear in its representation of Day 2, partly due to inaccuracies and omissions.

Omissions (0.3): There are significant omissions and inaccuracies, particularly in omitting details about the MCAT prep and meals, and incorrectly summarizing the half-marathon prep.

### **Conclusion**

In conclusion, Example B is a better reflection of the Day 2 content compared to Example C, mainly due to its more balanced coverage and fewer inaccuracies, though both examples have room for improvement in terms of specificity and comprehensively covering all activities

### **.3 Long-Term Memory Evaluation: Summary and Specifics (Prompt A)**

#### **Criteria Prompt for GPT**

The following output is what my LLM gave me. Please rank how well the day 16 output matched the day 4 output for examples A and B. Matching is determined by 2 criteria: A. Summary similarity ranking (how well does a summary of the day 16 output match the day 4 summary with respect to the goals of the day), B. specificity (how well did the day 16 output remember activities that were suggested on day 4). Please rank these separately on a scale of (0-1) based on the criteria: content coverage and correctness, and give a short reason of this ranking (and the criteria).

#### **Evaluation**

##### **Streaming-LLM**

1. **\*\*Summary Similarity Ranking\*\***: 0.1/1 - **\*\*Content Coverage and Correctness\*\***: The Day 16 output significantly deviates from the Day 4 output's focus on MCAT preparation and half-marathon training. It shifts towards more diverse and leisure activities, lacking alignment with the original goals.
2. **\*\*Specificity\*\***: 0.1/1 - **\*\*Content Coverage and Correctness\*\***: The Day 16 output does not recall or build upon the specific activities suggested on Day 4, such as the focus on MCAT study and physical training for a half-marathon. It introduces entirely new activities unrelated to the original plan.

##### **Summarized-Streaming-LLM**

1. **\*\*Summary Similarity Ranking\*\***: 0.8/1 - **\*\*Content Coverage and Correctness\*\***: The Day 16 output maintains a strong alignment with Day 4, focusing on MCAT preparation and half-marathon training. The structure and theme are consistent, though there are variations in the specific activities.
2. **\*\*Specificity\*\***: 0.7/1 - **\*\*Content Coverage and Correctness\*\***: Day 16 output remembers and builds upon the themes of MCAT study and half-marathon training from Day 4. It adapts the activities to show progress (e.g., covering different MCAT sections), indicating good recall and specificity, albeit with some variation.

##### **More-Summarized-Streaming-LLM**

1. **\*\*Summary Similarity Ranking\*\***: 0.7/1 - **\*\*Content Coverage and Correctness\*\***: The Day 16 output still revolves around the general theme of MCAT prep and physical fitness but with less emphasis on the specific MCAT and half-marathon training compared to Day 4. The focus seems slightly shifted towards general well-being.
2. **\*\*Specificity\*\***: 0.5/1 - **\*\*Content Coverage and Correctness\*\***: There's some recall of the original themes (e.g., meditation, fitness activities) from Day 4, but the Day 16 output introduces new activities that do not directly build upon the specific suggestions of Day 4. The connection is present but not strongly emphasized.

#### **Summary**

- **\*\*Example A\*\*** shows the least alignment and specificity, largely deviating from the original goals.
- **\*\*Example B\*\*** demonstrates the highest consistency and specificity, closely following the themes and activities of Day 4.
- **\*\*Example C\*\*** maintains the theme but with less specificity and direct recall of Day 4 activities.

### **.4 Long-Term Memory Evaluation: Summary and Specifics (Prompt B)**

#### **Criteria Prompt for GPT**

The following output is what my LLM gave me. Please rank how well the day 16 output matched the day 7 output for examples A and B. Matching is determined by 2 criteria: A. Summary similarity ranking (how well does a summary of the day 16 output match the day 7 summary with respect to

the goals of the day), B. specificity (how well did the day 16 output remember activities that were suggested on day 7). Please rank these separately on a scale of (0-1) based on the criteria: content coverage and correctness, and give a short reason of this ranking (and the criteria).

## **Evaluation**

To evaluate the effectiveness of the PAST INFO in capturing information about DAY 2 for both Streaming-LLM and More-Summarized-Streaming-LLM, we'll consider the criteria of Content Coverage, Specificity, Brevity and Clarity, and Omissions. The scale ranges from 0 to 1, where 0 indicates no coverage or effectiveness and 1 indicates complete and perfect coverage.

### **Streaming-LLM**

Content Coverage: This summary captures the essence of DAY 2 activities, focusing on MCAT prep and half-marathon research. However, it omits specific details about the walks, meals, and the different times of day (morning, afternoon, evening). Score: 0.7

Specificity: The summary is specific in mentioning MCAT prep and half-marathon research but lacks the details of walking and meal preparation. Score: 0.6

Brevity and Clarity: The summary is brief and clear, focusing only on the main activities without extra detail. Score: 0.9

Omissions: Key omissions include the specifics of the walking sessions, meals, and the structure of the day. Score: 0.6

Overall Score for Streaming-LLM: 0.7

### **More-Summarized-Streaming-LLM**

Content Coverage: This summary includes fewer details than Example B, focusing primarily on half-marathon research and incorrectly summarizing the MCAT prep. Score: 0.5

Specificity: The specifics provided are mostly accurate but are less comprehensive than in Example B. The error in summarizing the MCAT prep activity reduces the score. Score: 0.4

Brevity and Clarity: While brief and somewhat clear, the summary's inaccuracy affects its clarity. Score: 0.6

Omissions: Significant omissions include most of the MCAT prep activities, the meals, and the specific details of the walks. Score: 0.4

Overall Score for More-Summarized-Streaming-LLM: 0.48

## **Conclusion**

In conclusion, Streaming-LLM scores higher because it more accurately and comprehensively captures the essence of DAY 2's activities, despite some omissions, compared to More-Summarized-Streaming-LLM, which includes inaccuracies and more significant omissions.