

MT-OSC: Path for LLMs that Get Lost in Multi-Turn Conversation

Jyotika Singh, Fang Tu, Miguel Ballesteros, Weiyi Sun,
Sandip Ghoshal, Michelle Yuan, Yassine Benajiba, Sujith Ravi, Dan Roth
Oracle AI

Correspondence: jyotika.s.singh@oracle.com

Abstract

Large language models (LLMs) suffer significant performance degradation when user instructions and context are distributed over multiple conversational turns, yet multi-turn (MT) interactions dominate chat interfaces. The routine approach of appending full chat history to prompts rapidly exhausts context windows, leading to increased latency, higher computational costs, and diminishing returns as conversations extend. We introduce MT-OSC, a **One-off Sequential Condensation** framework that efficiently and automatically condenses chat history in the background without disrupting the user experience. MT-OSC employs a Condenser Agent that uses a few-shot inference-based Condenser and a lightweight Decider to selectively retain essential information, reducing token counts by up to 72% in 10-turn dialogues. Evaluated across 13 state-of-the-art LLMs and diverse multi-turn benchmarks, MT-OSC consistently narrows the multi-turn performance gap—yielding improved or preserved accuracy across datasets while remaining robust to distractors and irrelevant turns. Our results establish MT-OSC as a scalable solution for multi-turn chats, enabling richer context within constrained input spaces, reducing latency and operational cost, while balancing performance.

1 Introduction

Large language models (LLMs) have achieved remarkable results on single-turn benchmarks, yet their performance can deteriorate significantly when task instructions or critical context are fragmented across multiple conversational turns—sometimes as early as the second exchange (Laban et al., 2025; Kwan et al., 2024). Despite this, such multi-turn (MT) interactions are characteristic of real-world dialogue, where users naturally break up information, clarifications, and intent across several exchanges.

This fragmentation introduces two fundamental challenges: (1) Dispersed information across turns

makes it increasingly difficult for LLMs to accurately recall and integrate crucial details, often leading to inconsistencies and degraded performance compared to single-message queries (Laban et al., 2025); (2) The prevalent solution of concatenating the entire chat transcript consumes valuable context window and quickly becomes unsustainable, incurring excess latency, cost, and potential information loss as conversations grow.

Although ad-hoc summarization of prior turns has become common, especially for longer chats, these methods risk omitting essential context and are often insufficient for complex multi-turn reasoning. Prior work on chat compression (Gliwa et al., 2019; Bae et al., 2022; Zhang et al., 2022; Chen et al., 2025) and retrieval-augmented dialogues (Xu et al., 2022; Liu et al., 2023) have explored strategies to shrink context size while preserving relevance. Nonetheless, such approaches frequently depend on model fine-tuning, explicit context selection, or complex hierarchical schemes, limiting their scalability and ease of integration.

In contrast, we address a complementary challenge to task decomposition research where prior work improves results by splitting a complex request into multiple focused prompts (Wen et al., 2024; Sarangi et al., 2025; Prasad et al., 2024; Wang et al., 2025b). Here, we focus on different, realistic scenario where information is *fragmented* over time and must be robustly integrated to avoid compounding errors.

We propose MT-OSC, a **One-off Sequential Condensation** framework that systematically condenses chat history as conversations evolve. MT-OSC operates entirely in the background, requiring no model modifications or fine-tuning and avoiding user-visible latency. Through a combination of a few-shot Condenser and a lightweight, data-driven Decider, MT-OSC dynamically preserves salient context, sharply reducing token overhead and safeguarding against information loss. Our comprehen-

sive evaluation spanning 13 SOTA LLMs and 10 diverse, multi-turn datasets (spanning underspecified MT samples (Laban et al., 2025) in sharded BFCL, GSM8K, HumanEval, Spider, ToTTo, and Summary of Haystack, and episodic MT samples from MT-Eval recollection, Expansion, Follow-up, and Refinement benchmarks (Kwan et al., 2024)) demonstrates that MT-OSC robustly closes the multi-turn performance gap, cutting chat history token significantly and remains resilient even in the presence of synthetically added noise and irrelevant turns. In summary, our primary contributions are:

- A task-agnostic condensing framework (**MT-OSC**) that achieves up to **72% token reduction** in 10-turn (10 user-assistant exchanges) conversations **without disrupting user experience through asynchronous background execution**.
- MT-OSC features a **Condenser Agent** to carry robust few-shot condensation for effective multi-turn context preservation with a configurable decider to minimize information loss.
- Extensive experiments across 13 top LLMs and 10 diverse benchmarks confirm MT-OSC’s **broad applicability and robustness to distractions** and redundant dialogue turns.

2 Related Work

A central theme in multi-turn research is compressing chat history into compact, reusable state representations instead of naively appending full transcripts (Gliwa et al., 2019; Feng et al., 2022; Wang et al., 2025a). While there have been extensive work on model pre-training (Press et al., 2022; Su et al., 2023) and optimization for fine-tuning (Rafailov et al., 2024) to improve long context handling, the core challenge remains: preserving task-relevant context as conversations lengthen. Bae et al. (2022) tackles this using finetuned T5 summarization model to retain only essential dialogue. Zhang et al. (2022) leverage hierarchical encodings and attention for iterative refinement of conversation history. Chen et al. (2025) introduce compressive memory by integrating user information with conversation summaries; their approach relies on supervised fine-tuning and preference optimization. Jiang et al. (2024) proposed using a small language model to compute token perplexity and selectively remove low-perplexity tokens, also reordering relevant content to beginning or end to

address the *lost in the middle* (Liu et al., 2024a; Levy et al., 2024) issue for long-context scenarios; however, this method requires re-compression for each query, limiting the ability to cache context. In contrast, our approach is distinguished by its lack of reliance on hierarchical encodings, model fine-tuning, heuristic-based compression, or manual context selection. Instead, our method operates unobtrusively in the background, enabling easy integration into any LLM-based chat system without specialized architectures or operational overhead.

Long context prompt compression methods (Pan et al., 2024; Jiang et al., 2024) compress very long inputs via token/segment pruning (often importance/perplexity-based) and filtering/reordering. This differs from short-horizon multi-turn settings, where redundancy is limited and correctness can depend on small but decisive cues (e.g., negations, corrections, key entity mentions); token deletion can therefore alter intent or break coreference, and reordering is less meaningful given the causal chronology of dialogue. As a result, these methods are not directly comparable baselines for MT-OSC, but are better viewed as complementary prompt-compression tools for very long histories.

Another influential line of research, retrieval-augmented conversation (Xu et al., 2022; Liu et al., 2023; Aadhithya et al., 2024; Salama et al., 2025), selectively re-injects relevant dialogue snippets using retrieval-based methods to address context growth and improve grounding. While retrieval helps limit prompt size, it may miss nuances spanning multiple turns or temporal dependencies, and often increases system complexity or user latency via real-time retrieval (Chen et al., 2025). By contrast, our strategy offers a complementary alternative, maintaining a continuously updated, compact representation that eliminates the need for instantaneous look-up or large-scale context storage.

Recent work has focused on systematic evaluation of multi-turn context handling methods. MT-Eval (Kwan et al., 2024) benchmarks LLM abilities on dialogue recollection, expansion, refinement, and follow-up. Zhang et al. (2025) stress evaluation criteria such as conversational quality, multi-turn reasoning, and tool usage. (Gorle et al., 2025) presents information loss study as multi-turn conversations progress. Laban et al. (2025) present a sharding-based evaluation protocol where fully specified single-turn data is artificially split into underspecified multistep interactions to simulate

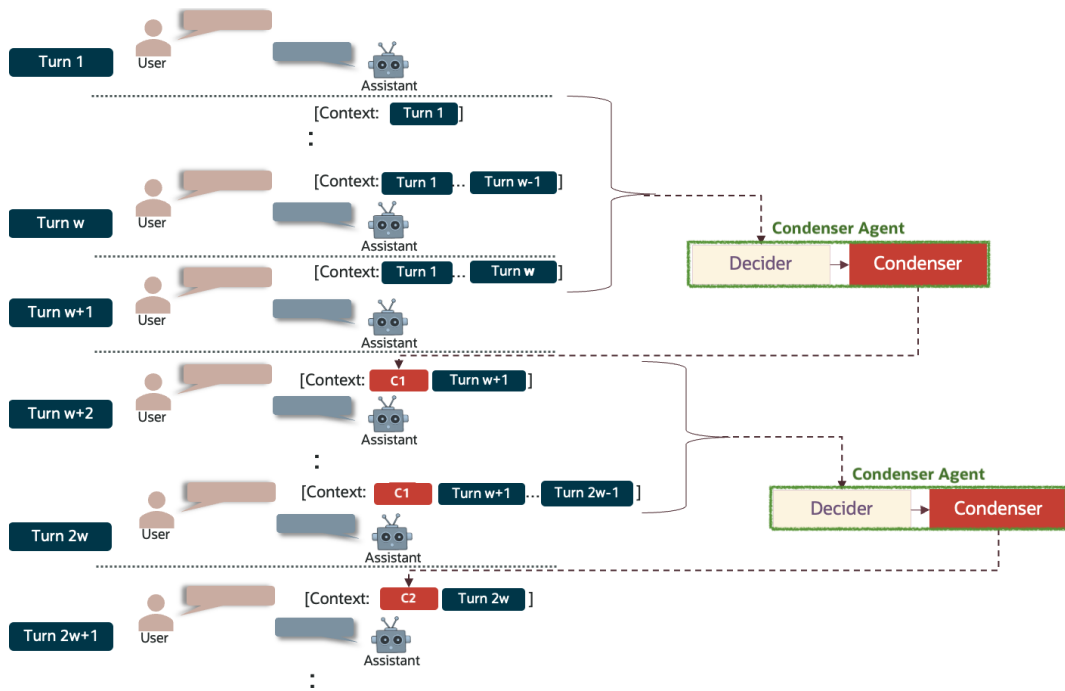


Figure 1: MT-OSC framework with one-off sequential condensation of chat history (w turns/exchanges at a time). This figure illustrates the chat flow when the Condenser Agent initiates condensation of prior history.

realistic chat dynamics. Baseline methods in (Lan et al., 2025) such as CONCAT and SNOWBALL, although useful for diagnosis, are infeasible for practical deployment: their token requirements scale linearly with chat length, and they often fail for episodic dialogues by repeatedly answering user’s older requests. Other proposed methods like RECAP intervene only at the final turn (unknown a priori), limiting their applicability for ongoing dialogues. Furthermore, recommendations such as "retry with more time" or "consolidate before retrying" require human intervention and lack systematic, automated mitigation strategies.

Overall, while prior work has elucidated the scope and severity of multi-turn performance decline, effective and generalizable solutions for automated multi-turn condensation remain relatively underexplored. MT-OSC fills this gap by providing a robust, scalable condensation process that streamlines multi-turn context integration without sacrificing accuracy, latency, or deployability. Appendix A shares extended details on related work.

3 Methods

MT-OSC Framework (illustrated in Figure 1) comprises a *Condenser agent* comprising a *Condenser* that condenses chat history and a *Decider* which determines the selective application of condensation; and a *one-off sequential* application strategy.

Condenser: This module leverages an LLM agent prompted in a few-shot style with explicit reasoning, tuned through curated exemplars to robustly condense chat history. We construct a diverse set of multi-turn chat examples to illustrate a spectrum of condensation policies: cases where key user text should be retained verbatim for future reference, scenarios requiring comprehensive consolidation, and situations where irrelevant turns may be safely excluded. Each hand-crafted example is paired with detailed rationales for information retention. This enables the Condenser to adapt dynamically to distinct dialogue patterns frequently observed in public multi-turn datasets, such as LMSYS-Chat1M (Zheng et al., 2024). These exemplars are independent of any of the data sources (discussed in Section 4) we use for our experiments in the paper which will further signify their generality in our results (Section 5).

The few-shot sample set (Figure 2) is central to maintaining nuanced context, as our ablation studies demonstrate that generic summarization often misses such subtleties (Section 5). Formally:

$$C_j = \text{Condense}(\{(u_i, a_i)\}_{i=0}^w) = (C_{ju}, C_{ja})$$

where C_j is the j -th condensed user-assistant pair and w denotes the number of turns processed per condensation operation. We use Llama-3.3-70B-Instruct as the underlying Condenser in our

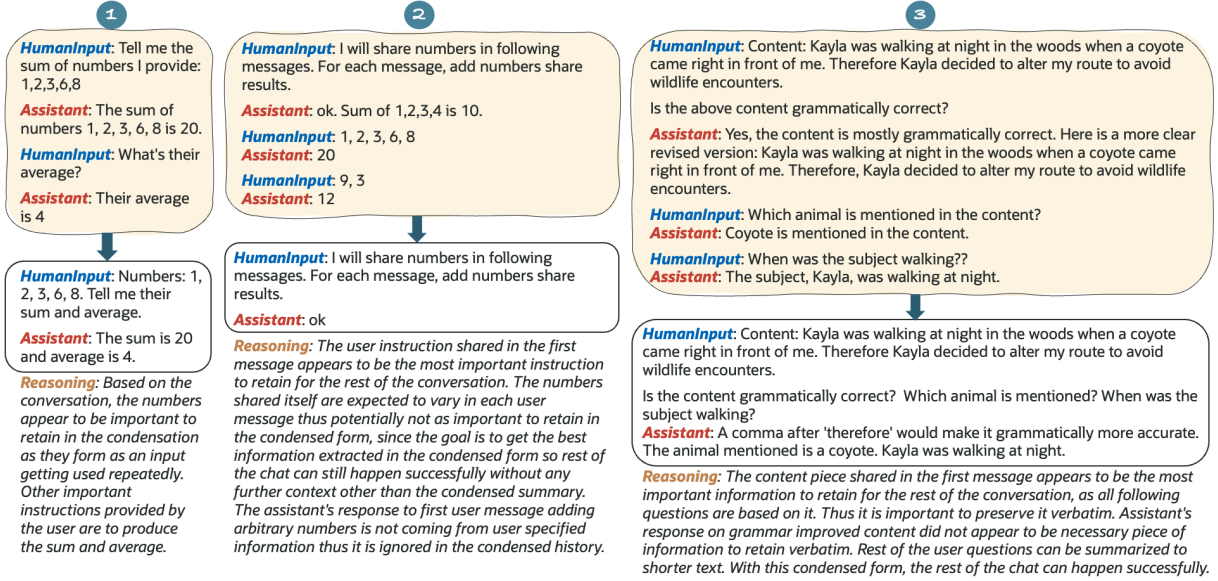


Figure 2: Examples of Multi-Turn Chat Condensing with Reasoning (original chat in yellow; condensed chat below.)

experiments; see Appendix B for prompt templates.

Decider: The Decider is a lightweight and configurable component that adaptively determines whether or not to apply condensation to a given conversation segment. Although condensation substantially reduces token load in many cases, certain dialogues where essential information is repeatedly referenced over several turns with dense context could risk information loss if condensed. To safeguard against this, the Decider employs a rule-based function that quantifies redundancy and novelty in assistant responses relative to prior user queries. Each user message U_i and assistant response A_i is represented as a set of normalized content words, following standard preprocessing (stop word removal, case normalization, and lemmatization). For each turn i , we extract unique, relevant words from A_i and remove those already mentioned in user messages up to turn i to identify new assistant terms. We further eliminate words that appear in later assistant turns, isolating content unique to each step and tracking overlap. If average overlap between assistant turns exceeds $\gamma\%$ and total user content is over τ tokens across 4 turns, condensation is withheld to avoid potential loss of repeatedly referenced information. This selective approach ensures the condenser maintains fidelity in information-dense dialogues. Mathematically:

$$Dw = \begin{cases} 1 & \text{if } \text{Overlap}(a_0 - aw) > \gamma \text{ and} \\ & \text{UserTokens}(H_t) > \tau \\ 0 & \text{otherwise} \end{cases}$$

where $\text{Overlap}(a_0, a_w)$ denotes the normalized word overlap between w assistant turns minus user shared content, γ is the average overlap threshold (e.g., 0.2), and τ is the user provided content token threshold in w turns (e.g., 1000). This approach provides a lightweight, configurable safeguard for information-rich or repetitive conversations.

In the main paper we share results on set values for Decider hyperparameters (γ 0.2, τ 1000). Appendix D.4 contains more details on sensitivity of the Decider parameters.

One-off Sequential Condensation: Relevant information is extracted from prior User and Assistant turns to generate a condensed chat history, which then replaces the raw history. The process condenses the conversation w turns at a time. At turn $w + 2$, the condensed history ($C1$) is used in place of the original chat history, while the most recent turn ($w + 1$) remains in its original form. The condensation repeats sequentially whenever another w turns have accumulated. ($C1$ (computed from the first w turns) plus the next $w - 1$ turns after turn w). This process repeats as the dialogue progresses, ensuring that only a compact and contextually salient form of history is included in subsequent model inputs.

Combined MT-OSC Operation is formalized as:

$$H_{w+2} = \begin{cases} \{(u_i, a_i)\}_{i=0}^{w+1} & (\text{if } D_w \text{ is False}) \\ \{(C_{ju}, C_{ja})\} \cup \{(u_k, a_k)\}_{k=t}^{w+1} & (\text{else}) \end{cases}$$

where t is the turn before which the last condensation C_j was performed, H_w is the chat history

up to turn w , D_w is the output of the Decider for the current window of size w , (u_i, a_i) is the user-assistant pair at turn i , C_j is the latest condensed representation of the current window, and w is the window size. See Appendix B for more details.

4 Datasets

We evaluate MT-OSC across 10 datasets, selected to encompass a wide spectrum of multi-turn chat scenarios. These datasets include both sharded static benchmarks and resembling naturally occurring dialogues, representing chats of up to 12 user turns—beyond the multi-turn range typically covered in prior work.

Sharded Datasets. We employ six prominent benchmarks (Laban et al., 2025)—GSM8K, BFCL-V3 Parallel, HumanEval (HEval), Spider, ToTTo, and Summary of Haystack (SoH)—in their sharded form. Each task, originally single-turn, is split into multiple turns to simulate underspecified, real-world multi-turn workflows, where instruction and context are fragmented across user utterances.

MT-EVAL Datasets. We further adopt four datasets drawn from MT-EVAL (Kwan et al., 2024), each reflecting a different facet of real conversational flow: recollection, refinement, expansion, and follow-up. Noting data scarcity in the recollection and expansion categories (each with only ten instances), we synthesize additional samples for these groups by generating new instances matched to the original style and distribution (see Appendix C.2).

Dataset Splits. We partition all data into development, validation, and test subsets with no instance overlap in downstream evaluation. Seven datasets are held out exclusively for testing; the remainder are split to support model development and hyperparameter tuning. Our primary results are reported on a test set of 752 multi-turn chat samples with 4–12 user turns. Complete breakdowns of corpus sizes and splits are in Appendix C.

Robustness Evaluation Data. The datasets are relatively clean without noise, distractions, or unnecessary context. To probe robustness, we inject simulated noise and redundancy between turns to mirror more realistic conversational messiness. These perturbations include: (i) **Repetition Injection**—randomly duplicating N turns in each chat; (ii) **Filler Injection**—randomly inserting vac-

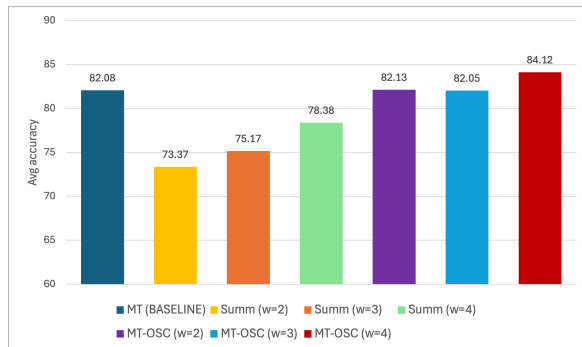


Figure 3: Comparison of performance aggregated across various datasets on simple summarization (Summ) using our one-off sequential application approach and our MT-OSC approach with window sizes 2, 3, 4.

uous filler turns (e.g., “Um,” “Hmm”); (iii) **Contextual Diversion**—inserting topically related but task-irrelevant utterances. These methods are inspired by prior attempts to stress-test context robustness (Levy et al., 2024; Mirzadeh et al., 2025), but are newly extended here to multi-turn settings (see Appendix C.3 for prompt and method details, C.3.1 for example of original vs. the three variants).

Evaluation Metrics. We use a suite of established metrics tailored to each dataset: accuracy for code and math (HumanEval, BFCL, GSM, Spider, Recollection+); LLM-based quality ratings (10-point scale) for open-ended dialogic tasks (Refinement, Follow-up, Expansion+); BLEU score for ToTTo; and a composite joint-score F1 for SoH. Where prior rule-based metrics proved inadequate (e.g., GSM final answer matching), we augment with LLM-judge reviews for correctness. Full details are in Appendix C.1.

Unless otherwise stated, our main experiments use Llama-3.3-70B-Instruct. For aggregate performance and token analyses, we restrict to accuracy-based datasets and provide per-task breakdowns as appropriate.

5 Results and Discussion

5.1 Performance vs. Baseline

We passed each dataset sample through the baseline approach of passing all prior turns into the model’s context (MT-baseline) and compare it with MT-OSC at different window sizes, averaged over three+ runs per sample. Figure 3 shows results across different window sizes, showing MT-OSC with an overall improved performance, with window size 4 showing the most improvement. See

Appendix D for details and parameter settings, including information on other baselines (FIFO) that we explored but do not report because their low performance (compared to the industry-standard MT-baseline, which is the default and most widely used approach) makes comparisons uninformative.

Table 1 contains the results broken down by dataset and shows that MT-OSC ($w=4$) achieves improved or largely maintained performance across datasets, while reducing the average chat history tokens. Given our exemplars are from an independent source than the datasets we present results on; this supports their generality in our method. Statistical analysis via Wilcoxon signed-rank test shows a *statistically significant reduction in token count* ($statistic = 28.0, p = 0.009$), while performance remains statistically indistinguishable ($statistic = 7.0, p = 0.118$) from the MT-baseline—demonstrating MT-OSC’s efficiency without compromising on performance.

	Performance		Chat history tokens	
	MT-baseline	MT-OSC*	MT-baseline	MT-OSC*
BFCL ^{shrd}	81.13%	86.79%	165	124
GSM ^{shrd}	83.45%	84.80%	1260	354
HEval ^{shrd}	74.67%	77.33%	539	272
Spider ^{shrd}	76.95%	79.44%	105	104
ToTTo ^{shrd}	0.19	0.19	2585	2585
SoH ^{shrd}	0.13	0.13	7495	7495
refine ^{mte}	5.39	5.35	1663	1342
follow ^{mte}	8.47	8.47	1379	1379
recoll ^{mte+}	94.22%	92.22%	1258	562
expan ^{mte+}	8.61	7.9	873	648

Table 1: Average performance and total chat history tokens (per turn) comparing MT-baseline and MT-OSC ($w=4$) across datasets. Metrics: accuracy (BFCL, GSM, HEval, Spider, recoll), 10-point rating (refine, follow, expan), BLEU (ToTTo), and joint score (SoH).

Context Loss Mitigation: The common reasons for drop in performance in multi-turn conversations is the risk of either failing to recall pertinent context or propagating incorrect assumptions—especially before users provide sufficient detail or clarification (Laban et al., 2025). This issue is particularly prominent in sharded datasets such as GSM, BFCL, HumanEval, and Spider, where initial conversational turns frequently lack adequate context, causing large language models to make premature

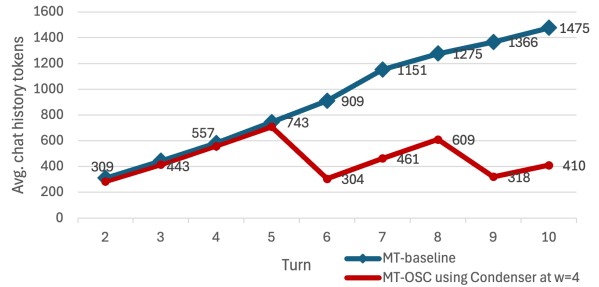


Figure 4: Chat history tokens growth in MT-baseline compared to using Condenser ($w = 4$) showing reduction in token consumption in the user-facing chat.

or unsupported inferences. As subsequent turns unfold and more information becomes available, models sometimes struggle to reconcile earlier assumptions, potentially leading to confusion or compounded errors. Notably, in these challenging scenarios, MT-OSC offers a clear advantage by condensing chat history to include only the most salient and contextually important information leading to improved accuracy and more consistent reasoning, as demonstrated by our quantitative results. For illustrative cases where this approach led to correction of earlier misinterpretations, we provide detailed examples in Appendix E (Figure 14).

5.2 Token Reduction

Figure 4 shows that *MT-OSC significantly reduces chat history tokens up to 72% for 10-turn chats* compared to passage of full chat history to the model’s context in MT-baseline, *bypassing linear growth in token consumption as the chat grows*.

Table 1 presents the results broken down by dataset and shows that MT-OSC ($w = 4$) reduces the average number of chat history tokens. The only datasets where token counts do not decrease are those in which most samples have fewer than $w + 2$ total turns (e.g., MT-EVAL-follow-up and Spider (with only 6 samples)), or where the Decider did not pass the samples through the Condenser component (e.g., ToTTo and SoH). Other datasets that utilized the Condenser show reductions in chat history tokens ranging from 19% to 72%, with an average reduction of 41%. These results include both samples with fewer than six turns and those with six or more turns. The true impact of the Condenser is observed in conversations with six or more turns, where chat history tokens are reduced by an average of 44.56% across those datasets, as shown in Figure 12 in Appendix D.3.

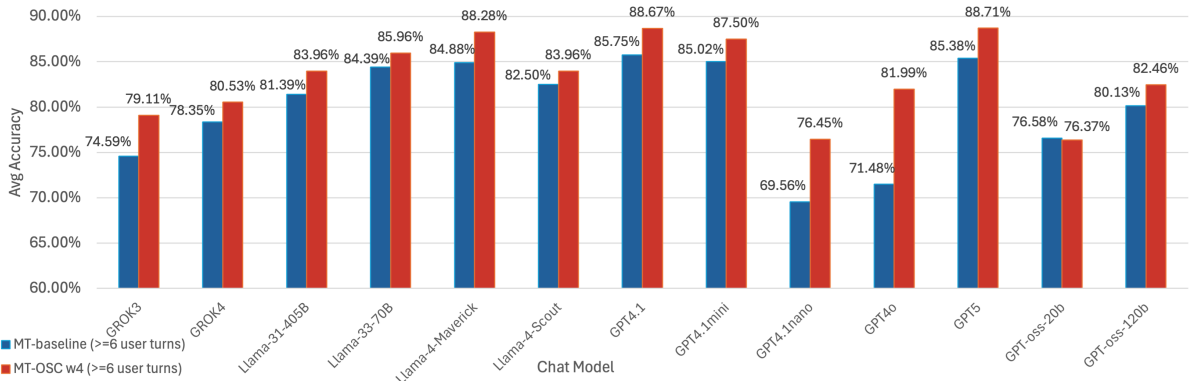


Figure 5: Model performance scores for various Chat Models for conversations with 6 or more turns across baseline multi-turn and MT-OSC with window size $w = 4$. See Table 5 for broken down details in tabular form.

Chat Efficiency for the End-User: MT-OSC reduces the chat history included in the main chat prompt, thereby decreasing the number of tokens and consequently lowering latency for the end user. Prompt length is a primary contributor to Time To First Token (TTFT), which measures how quickly users begin to see the model’s output (Agarwal et al., 2023). Empirical studies have shown a linear relationship between the number of prompt tokens and TTFT (Paruthi, 2024). *For chats with ≥ 6 turns, if chat history comprises 4,000 tokens, MT-OSC saves an average of 1,782 tokens per exchange* (reducing from 4,000 to 2,218 tokens), which translates to *approximately 1.1-1.2 seconds in latency reduction* based on findings in (Paruthi, 2024). This reduction provides a tangible improvement in end-user experience.

In real-world applications, the number of turns persisted in a chat may often be restricted to conserve computational resources, with a limit of around 10 turns for many exploratory dialogues (Marino, 2025; team, 2025). MT-OSC maintains low chat history token consumption, limiting history turns to ≤ 5 when $w=4$, thereby enabling efficient operation within these practical constraints.

Overall Solution Efficiency (User-Facing Chat + Background Process): Tokens are still consumed during the condensing operation, which is processed in the background and does not affect the user experience. This background processing nevertheless contributes to the overall token usage and associated costs. To quantify the balance between token savings and additional background consumption, Figure 6 presents the total number of tokens used—encompassing both real-time chat and background condenser operations—for the baseline multi-turn scenario and for MT-OSC Condenser. In

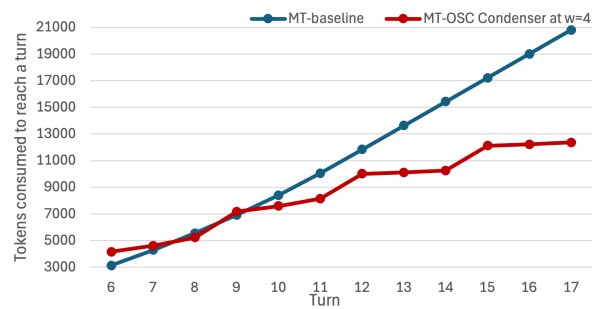


Figure 6: Total tokens consumed by chat history in the main chat + background condenser, showing overall cost benefit and overhead balancing as the chat grows.

comparison, *as conversations progress, the overall token consumption is a lot lower with condensation*, highlighting the efficiency of our approach.

Asynchronous Execution Advantage MT-OSC runs as a background process whose output is not required for the immediate next turn in the chat. Since its prompt length is typically comparable to or smaller than the main chat—often shorter due to additional system prompts and user typing delays—the condensation step usually completes while the next turn wraps. Its result is therefore available by the next-to-next turn. This enables a one-off, asynchronous condensation step that *adds no latency to the user experience*: the conversation proceeds normally while MT-OSC executes in the background, and its output is seamlessly used in one-off subsequent turns.

5.3 Performance Across LLMs

Generalization Across Chat LLMs We evaluated MT-OSC with a window size $w=4$ across 13 leading LLMs compared to MT-baseline, as summarized in Figure 5. *Consistently across mod-*

els, including the latest releases, MT-OSC outperformed the baseline in task accuracy. See Appendix D for detailed results by dataset.

Notably, GPT-5 achieved the highest performance under the MT-baseline among the tested models followed by GPT4o, yet MT-OSC further boosted its performance by 3.3%. Similarly, we notice improvement with MT-OSC across Llama and Grok models. These results demonstrate that MT-OSC delivers consistent gains in performance across a diverse range of top models across different LLM architectures.

Generalization Across Condenser LLMs In this paper main experiments, we report results using Llama-3.3-70B as the fixed Condenser LLM. To assess whether MT-OSC generalizes across Condenser models, we additionally reran a subset of datasets (Sharded BFCL and HumanEval) with alternative Condenser models. As shown in Table 6 (Appendix D.1), replacing the Condenser model with Llama-4-Maverick, GPT-4.1, or Gemini-2.5-Flash—while keeping the overall setup, prompts, and hyperparameters unchanged—yields consistent performance retention or improvement on the evaluated subset.

5.4 Ablation Study

To better understand the contribution of individual components in MT-OSC, we conduct two targeted ablation studies: 1) **Decider:** We evaluate the impact of the Decider by comparing full MT-OSC to a variant of the Condenser Agent with the Decider removed, measuring performance differences. 2) **Condenser vs. Simple Summarization:** With Decider disabled and $w = (2, 3, 4)$, we replace our Condenser module with a simple summarization prompt (see Appendix B for details), which provides pointed summarization instructions but omits the curated examples used in our Condenser.

Dataset	Performance		Chat Hist Tokens	
	w/ D	w/o D	w/ D	w/o D
ToTTo ^{shrd}	0.18	0.09	2585	1888
SoH ^{shrd}	0.13	0.08	7495	4086
Refin ^{mtev}	5.35	5.18	1342	593

Table 2: Performance and chat history tokens with and without Decider (D) across the datasets where Decider induces a performance or token count difference.

Decider Ablation: Table 2 summarizes the results on datasets that led to a performance difference without the Decider component. We find that the Decider component consistently maintains high chat success rates, especially on more challenging datasets where condensation may likely lose relevant information. *Ablation of the Decider generally leads to reduced performance on three datasets, highlighting its role in safeguarding against information loss during condensation.* Since $w=4$ shows an impact on chats with six or more turns, we compute the reduction in chat history tokens per dataset for chats with at least six turns. *With the Decider, chat history tokens are reduced by an average of 44.56%, and without the Decider, the reduction is 52.07%* (Figure 12 in Appendix D.3). Decider is a lightweight module and tunable based on the nature of the data and priority between performance and token length. An example from the Sharded ToTTo dataset where Decider skips condensing and prevents information loss is shared in Appendix D.3, and details on sensitivity of the Decider parameters are shared in Appendix D.4.

Condenser vs. Simple Summarization: Figure 3 shows that replacing the Condenser with a *simple summarizer decreases success rates across datasets and window sizes* demonstrating the effectiveness of our curated few-shot approach in maintaining better shortened chat histories and the limitations of the summary prompt baseline.

In our experiments, we observe that the summary prompt often omits important user instructions during summarization. In contrast, our condenser-based method provides critical examples, enabling the model to preserve the full text when necessary and to condense or omit less relevant information as appropriate. Notably, our approach achieves these improvements without reliance on a reasoning-specialized LLMs given the demonstrated performance using Llama-3.3-70B-Ins as the Condenser model. An example showing Condenser and simple summarizer outputs is shared in Appendix E.

These studies show that the Condenser is essential to MT-OSC’s ability to preserve and often enhance performance in multi-turn dialogue settings. The Decider, as a lightweight and tunable module, offers additional protection against performance degradation, especially on complex or information-dense datasets, by safeguarding against potential information loss during condensation.

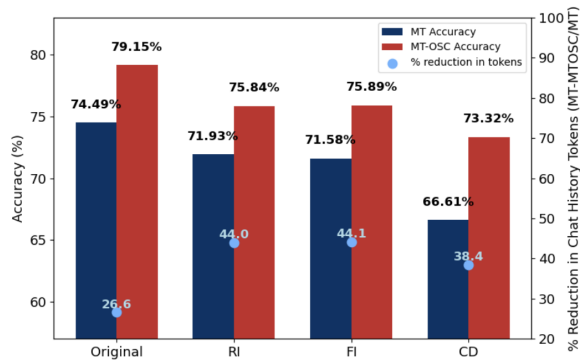


Figure 7: Comparison of performance between MT and MT-OSC, and reduction in chat history tokens, on the three different variants created for robustness testing (1. Repetition Infusion (RI); 2. Filler Injection (FI); 3. Contextual Diversion (CD)).

5.5 Robustness Testing

We evaluated the robustness of our approach using three simulated variants of the original datasets (as described in Section 4) with 1-2 added turns, applying both full multi-turn chat execution and MT-OSC. The comparative results are presented in Figure 7 aggregated across Grok-3, Llama-3.3-70B, Llama-4-Maverick, Llama-4-Scout, GPT4.1, and GPT4o and sharded BFCL, GSM, HEval, and Spider datasets. See Appendix D.2 for detailed results by dataset.

Our results show that the *inclusion of extra turns degrades multi-turn performance relative to the clean version of the data*, with a performance drop of about 7.88% for the contextual diversion variant and 2.5-3% for the other two variants (see Figure 7). This drop aligns with prior work, which reports that increased context length can impair task performance (Levy et al., 2024). Our results extend this observation to multi-turn scenarios.

The results show that *MT-OSC works just as well on these variants showing similar (about 4%) or even higher (up to 7% in Contextual Diversion variants) absolute gains in performance than on original/cleaner data while reducing chat history tokens*. Importantly, MT-OSC effectively mitigates the impact of unnecessary information by condensing the chat history to retain only the most relevant content, maintaining stable performance benefits even on noisier data.

To test **robustness against varying amounts of noise**, we also conducted an experiment to vary the number of extra turns added (1-5) for the contextual divergent variants. Figure 8 shows that increasing the number of distractor turns (and thus the con-

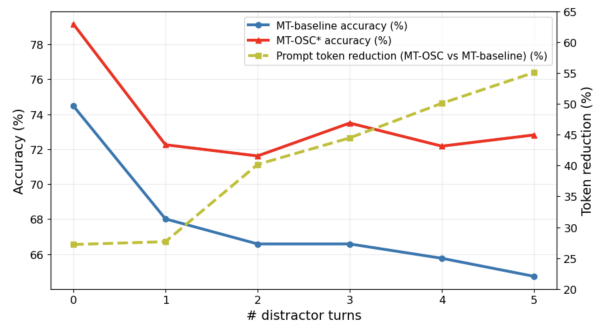


Figure 8: Performance of MT-baseline and MT-OSC under Contextual Diversion (CD) variants varied by number of distraction turns added.

versation length and user input tokens) degrades baseline performance and MT-OSC shows consistent improvements throughout. *Token reduction increases as the amount of distracting content grows*, showing that *MT-OSC successfully handles noise* and retains task-relevant information.

6 Conclusion

Multi-turn conversations present persistent challenges for LLMs, resulting in both degraded task performance and unsustainable increases in prompt length as dialogues scale. To address these limitations, we introduce MT-OSC, background condensation framework that sequentially condenses chat history as conversations grow. MT-OSC achieves significant reductions in chat history token counts (up to 72% fewer tokens for 10-turn dialogues) while either preserving or enhancing accuracy across a diverse set of benchmarks and state-of-the-art LLM architectures.

Our evaluation demonstrates that MT-OSC not only narrows the multi-turn performance gap but also provides robust gains in settings with conversational noise or distractors, enabling more efficient and reliable interactions. Importantly, MT-OSC operates entirely in the background: it does not require model modification, retraining, or user intervention, and it delivers tangible improvements in latency and efficiency for real-world deployments.

Overall, MT-OSC provides a practical, scalable solution to the multi-turn context bottleneck, enabling broader applications of conversational AI in chat systems.

Limitations

This study utilizes publicly available datasets that are designed for comparing single-turn and multi-turn conversational interactions. These datasets typ-

ically contain fewer than 10–12 user turns per conversation and primarily feature clean, single-topic dialogues with minimal agentic behavior—that is, they involve straightforward LLM calls without exploring the tool-use capabilities that confer agentic properties to LLMs. While we applied data augmentation to assess robustness, the underlying data represents limitation in linguistic complexity and topic diversity that characterize more diverse real-world conversational settings.

For **future work**, model training can explicitly incorporate more complex multi-turn interactions, enabling forthcoming models to better handle the challenges presented by extended conversations. Developing or incorporating datasets with longer, more diverse, and multi-topic conversations would provide a richer foundation for analysis and modeling. Further, leveraging data that reflects greater conversational complexity, including open-domain and agent-driven exchanges, could enhance the generalizability and applicability of future findings. Exploring methods such as topical decomposition may also provide valuable support for tackling the challenges posed by complex, multi-faceted dialogue interactions.

References

- Aadharsh Aadhithya, Sachin Kumar, and KP Soman. 2024. [Enhancing long-term memory using hierarchical aggregate tree for retrieval augmented generation](#). *arXiv e-prints*, pages arXiv–2406.
- Amit Agarwal, Hansa Meghwani, Hitesh Laxmichand Patel, Tao Sheng, Sujith Ravi, and Dan Roth. 2025. [Aligning LLMs for multilingual consistency in enterprise applications](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 117–137, Suzhou (China). Association for Computational Linguistics.
- Megha Agarwal, Asfandyar Qureshi, Nikhil Sardana, Linden Li, Julian Quevedo, and Daya Khudia. 2023. [Llm inference performance engineering: Best practices](#).
- Sanghwan Bae, Donghyun Kwak, Soyoung Kang, Min Young Lee, Sungdong Kim, Yui Jeong, Hyeri Kim, Sang-Woo Lee, Woomyoung Park, and Nako Sung. 2022. [Keep me updated! memory management in long-term conversations](#). In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3769–3787, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, and Wanli Ouyang. 2024. [MT-bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7421–7454, Bangkok, Thailand. Association for Computational Linguistics.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 34 others. 2021. [Evaluating large language models trained on code](#). *ArXiv*, abs/2107.03374.
- Nuo Chen, Hongguang Li, Jianhui Chang, Juhua Huang, Baoyuan Wang, and Jia Li. 2025. [Compress to impress: Unleashing the potential of compressive memory in real-world long-term conversations](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 755–773, Abu Dhabi, UAE. Association for Computational Linguistics.
- Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *ArXiv*, abs/2110.14168.
- Haodong Duan, Jueqi Wei, Chonghua Wang, Hongwei Liu, Yixiao Fang, Songyang Zhang, Dahua Lin, and Kai Chen. 2024. [BotChat: Evaluating LLMs’ capabilities of having multi-turn dialogues](#). In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3184–3200, Mexico City, Mexico. Association for Computational Linguistics.
- Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2022. [A survey on dialogue summarization: Recent advances and new frontiers](#). *arXiv preprint arXiv:2107.03175*.
- Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. [SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization](#). In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.
- Abhiram Rao Gorle, Amit Kumar Singh Yadav, and Tsachy Weissman. 2025. [Quantifying information gain and redundancy in multi-turn LLM conversations](#). In *First Workshop on Multi-Turn Interactions in Large Language Models*.
- Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. [LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression](#). In *Proceedings of the 62nd Annual Meeting*

- of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.
- Wai-Chung Kwan, Xingshan Zeng, Yuxin Jiang, Yufei Wang, Liangyou Li, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. 2024. [MT-eval: A multi-turn capabilities evaluation benchmark for large language models](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20153–20177, Miami, Florida, USA. Association for Computational Linguistics.
- Philippe Laban, Alexander Fabbri, Caiming Xiong, and Chien-Sheng Wu. 2024. [Summary of a haystack: A challenge to long-context LLMs and RAG systems](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9885–9903, Miami, Florida, USA. Association for Computational Linguistics.
- Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. 2025. [Llms get lost in multi-turn conversation](#). *Preprint*, arXiv:2505.06120.
- Mosh Levy, Alon Jacoby, and Yoav Goldberg. 2024. [Same task, more tokens: the impact of input length on the reasoning performance of large language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15339–15353, Bangkok, Thailand. Association for Computational Linguistics.
- Lei Liu, Xiaoyan Yang, Yue Shen, Binbin Hu, Zhiqiang Zhang, Jinjie Gu, and Guannan Zhang. 2023. [Think-in-memory: Recalling and post-thinking enable llms with long-term memory](#). *arXiv preprint arXiv:2311.08719*.
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024a. [Lost in the middle: How language models use long contexts](#). *Transactions of the Association for Computational Linguistics*, 12:157–173.
- Tianyang Liu, Fei Wang, and Muhao Chen. 2024b. [Re-thinking tabular data understanding with large language models](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 450–482, Mexico City, Mexico. Association for Computational Linguistics.
- Weiwen Liu, Xu Huang, Xingshan Zeng, xinlong hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong WANG, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Wang Xinzhi, Yong Liu, Yasheng Wang, and 8 others. 2025. [ToolACE: Winning the points of LLM function calling](#). In *The Thirteenth International Conference on Learning Representations*.
- Susie Marino. 2025. [35+ chatbot statistics you need to know for 2025](#). LocalIQ. Accessed: 29 Dec 2025.
- Hansa Meghwani, Amit Agarwal, Priyaranjan Pattnayak, Hitesh Laxmichand Patel, and Srikant Panda. 2025. [Hard negative mining for domain-specific retrieval in enterprise systems](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 1013–1026, Vienna, Austria. Association for Computational Linguistics.
- Seyed Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2025. [GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models](#). In *The Thirteenth International Conference on Learning Representations*.
- Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. [LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, Bangkok, Thailand. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: a method for automatic evaluation of machine translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. [ToTTo: A controlled table-to-text generation dataset](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online. Association for Computational Linguistics.
- Veraj Paruthi. 2024. [How input token count impacts the latency of ai chat tools](#). Glean. Accessed: 29 Dec 2025.
- Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025a. [The berkeley function calling leaderboard \(BFCL\): From tool use to agentic evaluation of large language models](#). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 48371–48392. PMLR.
- Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025b. [The berkeley function calling leaderboard \(BFCL\): From tool use to agentic evaluation of large language models](#). In *Forty-second International Conference on Machine Learning*.
- Priyaranjan Pattnayak, Amit Agarwal, Hansa Meghwani, Hitesh Laxmichand Patel, and Srikant Panda. 2025. [Hybrid AI for responsive multi-turn online](#)

- conversations with novel dynamic routing and feedback adaptation. In *Proceedings of the 4th International Workshop on Knowledge-Augmented Methods for Natural Language Processing*, pages 215–229, Albuquerque, New Mexico, USA. Association for Computational Linguistics.
- Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. 2024. *ADaPT: As-needed decomposition and planning with language models*. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4226–4252, Mexico City, Mexico. Association for Computational Linguistics.
- Ofir Press, Noah A. Smith, and Mike Lewis. 2022. *Train short, test long: Attention with linear biases enables input length extrapolation*. *Preprint*, arXiv:2108.12409.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. *Direct preference optimization: Your language model is secretly a reward model*. *Preprint*, arXiv:2305.18290.
- Rana Salama, Jason Cai, Michelle Yuan, Anna Currey, Monica Sunkara, Yi Zhang, and Yassine Benajiba. 2025. *MemInsight: Autonomous memory augmentation for LLM agents*. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 33136–33152, Suzhou, China. Association for Computational Linguistics.
- Sneheel Sarangi, Maha Elgarf, and Hanan Salam. 2025. *Decompose-ToM: Enhancing theory of mind reasoning in large language models through simulation and task decomposition*. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10228–10241, Abu Dhabi, UAE. Association for Computational Linguistics.
- J. Singh. 2021. *Social media analysis using natural language processing techniques*. In *Proceedings of the 20th Python in Science Conference, SciPy*, page 74–80. SciPy.
- Jyotika Singh. 2022. *pyaudioprocessing: Audio processing, feature extraction, and machine learning modeling*. In *Proceedings of the 21st Python in Science Conference, SciPy*, page 152–158. SciPy.
- Jyotika Singh. 2023. *Natural Language Processing in the Real World: Text Processing, Analytics, and Classification*. Chapman and Hall/CRC.
- Jyotika Singh, Rebecca Bilbro, Michael Avon, Scott Bowen, Dan Jolicoeur, and Serge Matta. 2021. *Method for optimizing media and marketing content using cross-platform video intelligence*. U.S. Patent. Issued March 16, 2021.
- Jyotika Singh, Weiyi Sun, Amit Agarwal, Viji Krishnamurthy, Yassine Benajiba, Sujith Ravi, and Dan Roth. 2025. *Can LLMs narrate tabular data? an evaluation framework for natural language representations of text-to-SQL system outputs*. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 883–902, Suzhou (China). Association for Computational Linguistics.
- Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. *Roformer: Enhanced transformer with rotary position embedding*. *Preprint*, arXiv:2104.09864.
- Goose team. 2025. *Smart context management*. Goose. Accessed: 29 Dec 2025.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. *Attention is all you need*. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.
- Qingyue Wang, Yanhe Fu, Yanan Cao, Shuai Wang, Zhiliang Tian, and Liang Ding. 2025a. *Recursively summarizing enables long-term dialogue memory in large language models*. *Neurocomput.*, 639(C).
- Yaoliang Wang, Zhiyong Wu, Junfeng Yao, and Jinsong Su. 2025b. *Tdag: A multi-agent framework based on dynamic task decomposition and agent generation*. *Neural Networks*, 185:107200.
- Jiixin Wen, Ruiqi Zhong, Pei Ke, Zhihong Shao, Hongning Wang, and Minlie Huang. 2024. *Learning task decomposition to assist humans in competitive programming*. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11700–11723, Bangkok, Thailand. Association for Computational Linguistics.
- Jing Xu, Arthur Szlam, and Jason Weston. 2022. *Beyond goldfish memory: Long-term open-domain conversation*. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5180–5197, Dublin, Ireland. Association for Computational Linguistics.
- Wenli Yang, Lilian Some, Michael Bain, and Byeong Kang. 2025. *A comprehensive survey on integrating large language models with knowledge-based methods*. *Knowledge-Based Systems*, 318:113503.
- Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. *Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task*. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.
- Michelle Yuan, Weiyi Sun, Amir H. Rezaeian, Jyotika Singh, Sandip Ghoshal, Yao-Ting Wang, Miguel Ballesteros, and Yassine Benajiba. 2026. *Barriers*

to discrete reasoning with transformers: A survey across depth, exactness, and bandwidth. *Preprint*, arXiv:2602.11175.

Chen Zhang, Xinyi Dai, Yaxiong Wu, Qu Yang, Yasheng Wang, Ruiming Tang, and Yong Liu. 2025. A survey on multi-turn interaction capabilities of large language models. *arXiv preprint arXiv:2501.09959*.

Tong Zhang, Yong Liu, Boyang Li, Zhiwei Zeng, Pengwei Wang, Yuan You, Chunyan Miao, and Lizhen Cui. 2022. History-aware hierarchical transformer for multi-session open-domain dialogue system. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3395–3407, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. 2024. *LMSYS-chat-1m: A large-scale real-world LLM conversation dataset*. In *The Twelfth International Conference on Learning Representations*.

A Extended related work

Chat systems in Natural Language Processing (NLP) (field comprising text and audio processing) have existed well before large language models (LLMs) became state of the art (Vaswani et al., 2017; Singh, 2022). NLP applications are widely used across industry verticals (Singh, 2023) and diverse data sources, including business and finance, social media, and even unstructured data (Yang et al., 2025; Singh et al., 2025; Liu et al., 2024b; Singh, 2021), spanning task-oriented, data-analysis-focused (Singh et al., 2021), and free-form settings, and now LLMs are heavily used across these data types and applications. While LLMs offer substantial advantages along with some limitations (Yuan et al., 2026), they have made real-world chat applications far more prevalent by reducing reliance on separate intent/entity detection models and hand-crafted rules to control behavior based on conversational signals. Still, reliance on single-turn benchmarks prevail and multi-turn chats underperform single-turn specified prompts.

Memory compression Chen et al. (2025) introduce the idea of compressive memory where they integrate conversation summaries with information about the user. They use Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO) to train a Llama-2 model to summarize and compress memories without dependence on external modules. Jiang et al. (2024) uses a small language model to calculate the perplexity of each token in the original prompt and then remove tokens with lower perplexities, and reorders relevant parts to the beginning and end to combat the "lost in the middle" problem for long-context scenarios. It requires re-compression for different questions, even with the same context, preventing caching of the context. In addition, aggressive compression can inadvertently remove key details, impacting accuracy for tasks requiring nuance.

Why long-context work is not competitive but complimentary Long-context compression methods such as LLMLingua-2 (Pan et al., 2024) and LongLLMLingua (Jiang et al., 2024) target a different regime of compressing very long inputs via token/segment pruning (often importance/perplexity-based) and, for LongLLMLingua, reordering/filtering. In short-horizon multi-turn setting, redundancy is limited and correctness often hinges on small but decisive state (e.g., nega-

tions, corrections like 'instead,' key entity mentions), so token-level deletion can disproportionately change intent or break coreference; reordering is also less meaningful when history is short and chronologically causal. Therefore, they are not an apples-to-apples comparison and not a suitable baseline for MT-OSC; rather, they are best viewed as prompt-compression tools that may be used in addition to MT-OSC for very long histories.

Multi-turn sharded datasets and hinted solution limitation Laban et al. (2025) formulates a unique evaluation method for underspecification in multi-turn conversations. They introduce a sharding process where they transform datasets with fully-specified instructions into sharded instructions that are revealed on subsequent turns. The CONCAT method proposed in the paper could perform better in underspecified settings but is impractical for real-world scenarios, as users are unlikely to resend all prior context with each message. If automated, it is also ineffective for episodic multi-turn conversations, where the model may struggle to distinguish between the current and previous tasks. Additionally, the approach significantly increases the number of tokens in the conversation, reducing overall efficiency. There are two other methods mentioned in the paper, RECAP and SNOWBALL. Although RECAP performs better than SNOWBALL, its intervention is conducted on the last turn of the conversation, which is not known a priori, and both have been shown to have worse performance than CONCAT. The two recommendations in the paper to mitigate information loss, "If time allows, try again." and "Consolidate before retrying.", are not principled approaches as they both require human evaluation and intervention.

B Method prompts and details

B.1 Condenser prompt

The prompt for *Condenser* is as follows. The examples used are shared in Figure 2.

The inference parameter settings were $temperature = 0.01$, $frequency_penalty = 1$, $max_completion_tokens = 10000$, and $top_p = 1$, and LLM was LLAMA-3.3-70B-INSTRUCT.

Condense the information from HumanInput and also share a concise summary of Assistant response to the human input. Make sure you don't miss any specific values and instructions provided by the human input that are relevant for the conversation. For the assistant response summary, make sure to keep any important points but keep it concise.

Also return the reasoning behind your condensation strategy. Return in JSON format "HumanInput": "<text>", "Assistant": "<text>", "Reasoning": "<text>". Only return the JSON with no additional text.

Examples:

Conversation History: <example-1>
<assistant-output-example-1>

Conversation History: <example-2>
<assistant-output-example-2>

Conversation History: <example-3>
<assistant-output-example-3>

Conversation History:

B.2 Simple summarization prompt

The prompt for the simple summarization is as follows.

Summarize the information from HumanInput and also share a concise summary of Assistant response to the human input. Make sure you don't miss any specific values and instructions provided by the human input. For the assistant response summary, make sure to keep any important points but keep it concise. Return in JSON format "HumanInput": "<text>", "Assistant": "<text>". Only return the JSON with no additional text.

The inference parameter settings were $temperature = 0.01$, $frequency_penalty = 1$, $max_completion_tokens = 10000$, and $top_p = 1$, and LLM was LLAMA-3.3-70B-INSTRUCT.

B.3 MT-OSC details

For the Decider, we used $\gamma 0.2$ and $\tau 1000$ tokens.

The one-off sequential condenser method (MT-OSC) for $w=4$ can be represented as follows in equation form.

Definitions:

- u_i : User message at turn i

- a_i : Assistant message at turn i
- $C_j = (C_{ju}, C_{ja})$: Output of the j -th condenser, summarized user-assistant pair
- w : Window size (number of user-assistant pairs condensed at a time)

Chat History Representation:

At turn 1: $H_1 = \emptyset$

At turn 2: $H_2 = \{(u_1, a_1)\}$

At turn 3: $H_3 = \{(u_1, a_1), (u_2, a_2)\}$

At turn 4: $H_4 = \{(u_1, a_1), (u_2, a_2), (u_3, a_3)\}$

When $|H_t| = w$ (e.g., $w = 4$), condensation is triggered as a background process:

$$C_1 = \text{Condense}(\{(u_1, a_1), (u_2, a_2), (u_3, a_3), (u_4, a_4)\}) = (C_{1u}, C_{1a})$$

Turn 5 continues while C_1 is generating:

At turn 5: $H_5 = \{(u_1, a_1), (u_2, a_2), (u_3, a_3), (u_4, a_4)\}$

After C_1 is generated:

At turn 6: $H_6 = \{(C_{1u}, C_{1a}), (u_5, a_5)\}$

At turn 7: $H_7 = \{(C_{1u}, C_{1a}), (u_5, a_5), (u_6, a_6)\}$

When H_t contains w new pairs, condensation is triggered again:

$$C_2 = \text{Condense}(\{(C_{1u}, C_{1a}), (u_5, a_5), (u_6, a_6), (u_7, a_7)\}) = (C_{2u}, C_{2a})$$

Turn 8 continues while C_2 is generating:

At turn 8: $H_8 = \{(C_{1u}, C_{1a}), (u_5, a_5), (u_6, a_6), (u_7, a_7)\}$

After C_2 is generated:

At turn 9: $H_9 = \{(C_{2u}, C_{2a}), (u_8, a_8)\}$
and so on

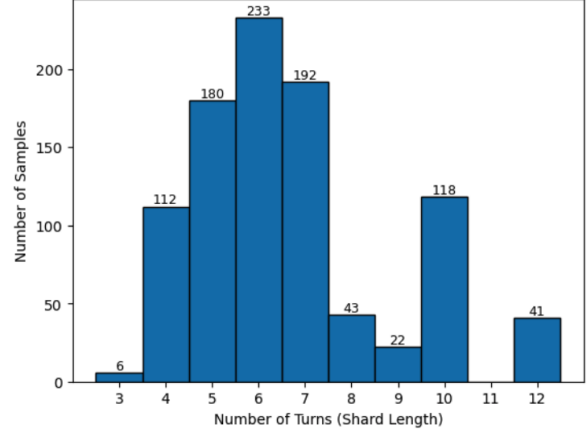


Figure 9: Histogram showing the number of samples for each maximum number of turns per sample in sharded + MTEval datasets.

C Dataset details

In general, there has been work done to study conversations across LLMs and applications (Patnayak et al., 2025; Meghwani et al., 2025; Agarwal et al., 2025), but not specific to multi-turn short-term memory. Below, we dig into this problem and closely related datasets specific to our paper’s multi-turn conversation scope in single chat sessions.

Datasets choice and details: While other multi-turn datasets exist in the literature, many contain only short conversations—for example, ToolAce (Liu et al., 2025) is limited to a maximum of four turns, and the more recent BFCL multi-turn and multi-step function calling dataset (Patil et al., 2025a) has very few examples exceeding five turns. The datasets we selected for our experiments contained up to 12 turns. Figure 9 shows the number of samples by number of turns for the datasets used for our experiments (sharded and MTEval datasets combined).

Table 3 shows a breakdown of datasets used, their source, sizes, splits between development, validation, and test sets. All samples in the dataset are less than or equal to 12 turns, with a vast majority of them being ≤ 10 turns. The total size of our development set is 50 samples, validation set is 72 samples, and test set (on which we report all the results in the paper) is 752 samples.

The development set was primarily used for manual inspection and for analyzing the impact of different experimental techniques and prompt variations. As a result, the development subset is smaller in size relative to the validation and test subsets. To

Dataset	Source	Success Metric	Size	Split (Dev, Val, Test)
Sharded BFCL	(Laban et al., 2025)	Accuracy [Exact match (Action matching)]	105	(30.5%, 19%, 50.5%)
Sharded GSM8K	(Laban et al., 2025)	Accuracy [Exact match (Final Answer)]	103	(-, 36%, 64%)
Sharded HumanEval	(Laban et al., 2025)	Accuracy [Exact match (Code test passing)]	45	(-, -, 100%)
Sharded Spider	(Laban et al., 2025)	Accuracy [Exact match (SQL execution output match)]	107	(-, -, 100%)
Sharded ToTTo	(Laban et al., 2025)	BLEU Score	120	(15%, 12.5%, 72.5%)
Sharded Summary of Haystack	(Laban et al., 2025)	LLMaaS ("Joint Score") to measure info coverage & attribution F1	94	(-, -, 100%)
MT-EVAL refinement	(Kwan et al., 2024)	LLMaaS for quality rating out of 10	40	(-, -, 100%)
MT-EVAL follow-up	(Kwan et al., 2024)	LLMaaS for quality rating out of 10	60	(-, -, 100%)
MT-EVAL recollection+	(Kwan et al., 2024)	Accuracy [Exact match (Final Answer)]	100	(-, -, 100%)
MT-EVAL expansion+	(Kwan et al., 2024)	LLMaaS for quality rating out of 10	100	(-, -, 100%)

Table 3: Dataset name, source, success metric for evaluation, size, and split. The sharded datasets were synthetically generated and manual inspection and validation by the authors of their respective sources cited above. The MT-EVAL datasets also went through human-in-the-loop creation, validation, and filtering process where the initial data generation utilized GPT-4.

ensure unbiased evaluation, we strictly excluded any overlap between the datasets used in the development phase and those used in validation and testing. In particular, all four MT-EVAL datasets were reserved exclusively for the test set. Among the six sharded datasets, some were allocated entirely to the test and/or validation sets, while others were divided across the development, validation, and test sets.

C.1 Evaluation details

Evaluation metrics per dataset are shared in Table 3.

For all datasets, we adhere as closely as possible to the original evaluation methods, making only minimal necessary assumptions. There has been more work on evaluation of multi-turn conversations using LLM-as-a-Judge (Duan et al., 2024), however those are free-form chats rather as task-oriented and hence not applicable to our work.

Sharded versions (Laban et al., 2025) of HumanEval (Chen et al., 2021), Spider (Yu et al., 2018), BFCL-V3 Parallel (Patil et al., 2025b), and GSM8K (Cobbe et al., 2021) are assessed based on

binary correctness—either through execution (code in HumanEval and SQL queries in Spider) or by checking semantic equivalence to a reference answer (API calls in BFCL and numerical responses in GSM). Each sample consists of multiple conversational turns, with each final answer attempt evaluated for correctness. If a correct answer is produced at any point during the conversation, the sample is considered a success.

Sharded versions of ToTTo (Parikh et al., 2020) and Summary of Haystack (Laban et al., 2024) datasets resemble refinement tasks evaluated on a continuous scale from 0 to 1. ToTTo uses the BLEU metric (Papineni et al., 2002), while Summary of Haystack is scored using a custom LLM-as-a-judge metric called Joint Score, designed to assess the summary’s information coverage and attribution accuracy. For both tasks, we evaluate the model’s response at the final turn of the conversation, as it is expected that earlier turns will progressively refine the answer leading up to this point for these datasets.

MT-Eval datasets (inspired by LMSYS-Chat-1M

(Zheng et al., 2024) and MT-Bench (Bai et al., 2024)) for Refinement, Expansion, and Follow-up are passed through an LLM judge to rate answer for each turn in a sample, which is then averaged to get an average score per sample in the dataset. MT-Eval Recollection dataset uses exact match for category output by the model based on ground-truth categories.

Sharded GSM8K evaluation: Our evaluation approach for sharded GSM8K dataset diverges from the rule-based method used in the source paper, which we found is prone to false negatives—particularly when the numerical answer is not positioned at the end of the response string. For example, the method marks “for the 5 items, the price is \$78” as correct, but “the price is \$78 for the 5 items” as incorrect, despite their semantic equivalence. Manual inspection of mismatches revealed multiple cases where identical or correct answers failed to match due to these limitations. For our final reported results, we passed the responses not matched by the rule-based methods to be subsequently assessed by an LLM-as-a-Judge using LLAMA-3.3-70B-INSTRUCT with *temperature* = 0, *top_p* = 1, and *max_tokens* = 4096. Manual inspection revealed resolution of false negatives using this 2-tier evaluation approach for this dataset. The judge prompt used is as follows:

I am sharing a user Q and a reference answer for that question. Then I will show you a model generated answer.
Return True if the model generated answer is correct and the final answer it contains matches the answer in the reference text. Otherwise, return False.

User Q: {}

Reference answer: {}

Model generated answer: {}

C.2 MT-EVAL Synthesis

Two out of the four MT-EVAL datasets, namely **Recollection** and **Expansion**, originally contained only 10 samples each. To increase this number to 100 samples per dataset, we used the following methods:

MT-EVAL Recollection Dataset Each sample consists of a user instruction followed by document

text in all subsequent turns. The dataset contains 100 unique documents in total, with 10 documents (one per turn) used in each sample. To generate additional samples, we randomly selected 10 different documents from the pool of 100 for each new sample, while keeping the top-level user instruction and task unchanged.

MT-EVAL Expansion Dataset Each sample begins with a document in the first turn, followed by 10 different questions related to that document in subsequent turns. These questions depend only on the initial document and are not order-sensitive. To create variants, we shuffled the order of the 10 questions for each sample, producing 9 new variants per original sample. This process, combined with the 10 original samples, resulted in a total of 100 samples for this dataset.

C.3 Generating distracting turns

We observe that most publicly available multi-turn datasets are relatively clean with no distractions or noise. To further assess the robustness of our experiments, we introduced simulated redundancy and task-irrelevant information between turns within the above datasets. This modification allowed us to evaluate performance in less curated, more realistic conversational settings, thereby ensuring our results generalize beyond cleaner, idealized data. Specifically, we introduced the following types of redundancy and irrelevant information into the multi-turn conversations:

1. **Repetition Infusion:** Repeated random turns within the multi-turn dialogues - for each multi-turn conversation, we randomly select N turns in the middle (not the first turn or the last turn) and repeat them once in the same position. N is set to be a portion of the total number of turns in the conversation.
2. **Filler Injection:** Inserted random turns containing only meaningless filler words, such as 'Um.', 'Uh.', 'Well.', 'Anyway.', 'Ok.', 'Hmm.', etc. - for each multi-turn conversation, we randomly select N turns in the middle (exclude the first turn) and add a randomly selected filler word as extra turn right before them.
3. **Contextual Diversion:** For each multi-turn chat, we randomly select N turns from the middle (excluding the first and last) and add

new, contextually relevant turns that aren't essential for task completion. If a selected turn isn't blank, we generate an extra turn based on its content and place it immediately after. Appendix C.3 shows the prompt we use to generate the extra distracting turns.

We used the model GPT4o1-200B-128 to generate the distracting turns for Contextual Diversion with inference parameter settings as follows: $temperature = 1.0$, $top_p = 0.75$, $max_tokens = 4096$.

Can you come up with related but distracting information as new entry in user inputs, for example:

Input:

Check for availability of meeting rooms for next week.

Output:

Meeting rooms are often in high demand, especially during peak business hours. It's a good idea to consider alternative locations or virtual meeting options in case the preferred rooms are fully booked. Additionally, checking the amenities available in each room, such as video conferencing equipment or whiteboards, can help ensure that the meeting runs smoothly.

Make sure the added information is not helping make the instructions more clear.

Please respond only with the output without any extra explanations or text.

Here is the input to modify:

C.3.1 Samples for Robustness Tests

Figure 10 contains examples for data samples and how they were augmented for robustness tests for *Repetition Infusion*, *Filler Injection*, and *Contextual Diversion*. The light color turns show the augmentations over existing data.

D Results

Across inferences against different models used for analysis, the parameters were set to the following settings: $temperature = 0.01$, $frequency_penalty = 1$ (for LLMs that support this inference parameter),

	Llama-3.3-70B	GPT 4.1
MT-baseline	81.13%	61.54%
MT-OSC*	86.79%	69.23%
FIFO	80.77% ↓	51.92% ↓

Table 4: FIFO compared to MT-baseline and MT-OSC for managing chat history shows that FIFO performs worse than the baseline (MT-baseline) approach on the Sharded BFCL dataset. *=ours

$max_completion_tokens = 10000$, and $top_p = 1$.

Figure 11 shows performance across of datasets using raw chat turns in model's context (MT (baseline)), using one-off sequential approach using simple summarizer prompt, and using MT-OSC. Results are across window sizes 2, 3, and 4. The results show largely maintained or improved performance with MT-OSC.

FIFO We also experimented with a first-in-first-out (FIFO) context strategy; however, in our task-oriented datasets, relevant information often appears throughout early turns, so FIFO dropped critical context and resulted in substantially lower accuracy, and we therefore omit it from reporting in our extensive evaluations. Table 4 shows FIFO results on our Sharded BFCL test set compared to MT-baseline and MT-OSC for chat models Llama-3.3-70B and GPT 4.1. Since it underperforms the industry-standard default MT-baseline, we do not include it in the main experiments. We set the FIFO limit to 4 turns: when the chat history exceeds 4 turns, the earliest turn is discarded. This matches MT-OSC with window size 4, where the chat history for any given turn contains at most four prior turns.

D.1 Results across 13 LLMs

Table 5 shows results for datasets across 13 LLMs for MT-baseline and MT-OSC.

Results across Condenser models Table 6 shows results broken down by dataset (sharded BFCL and HumanEval subsets) and chat model (Llama-4-Maverick, GPT 4.1, GPT 4o), for different condenser model choices (Llama-3.3-70B(default), Llama-4-Maverick, GPT4.1, Gemini 2.5 Flash) compared to MT-baseline for each chat model.

- "Find the prime factorization"
- "Start with the prime factorization of the number 4567"
- "Also, find the prime factorization of the number 7890"
- "People sometimes enjoy creating decorative pieces featuring prime factorization patterns, which can be a fun way to showcase mathematical curiosities. At the same time, it might be helpful to ensure you have the latest hardware if you plan on doing frequent factorization tasks, though this doesn't necessarily influence the result itself."
- "Ensure the results are returned in a 'dictionary' style"

Contextual Diversion

- "Find the prime factorization"
- "Start with the prime factorization of the number 4567"
- "umm"
- "Also, find the prime factorization of the number 7890"
- "ok"
- "Ensure the results are returned in a 'dictionary' style"

Filler Injection

- "Find the prime factorization"
- "Start with the prime factorization of the number 4567"
- "Start with the prime factorization of the number 4567"
- "Also, find the prime factorization of the number 7890"
- "Ensure the results are returned in a 'dictionary' style"

Repetition Infusion

Figure 10: These examples shows the original turns from a conversation (in darker color in each example) and the perturbed version with new random turns (shown in the lighter color in each example) added to the conversation.



Figure 11: Comparison of performance across datasets of simple summarization (Summ) using the one-off sequential application approach (window sizes 2, 3, and 4) and the MT-OSC (window sizes 2, 3, and 4). Chat model is Llama-3.3-70B-Instruct.

	GSM <i>shrd</i>	BFCL <i>shrd</i>	HEval <i>shrd</i>	Recoll <i>mte+</i>	Spider <i>shrd*</i>	Expan <i>mte+</i>	Refin <i>mte</i>	Avg <i>acc%</i>
Metric	accuracy %					rating out of 10		
GROK3								
MT	0.82	0.67	0.51	0.96	0.5	9.01	6.26	0.75
MTOSC	0.86	0.71	0.62	0.95	0.5	8.38	6.22	0.79
GROK4								
MT	0.84	0.64	0.84	0.80	0.5	9.05	5.92	0.78
MTOSC	0.86	0.69	0.80	0.86	0.83	8.37	6.19	0.81
Llama-3.1-405B								
MT	0.82	0.82	0.69	0.92	0.67	8.65	5.52	0.81
MTOSC	0.86	0.84	0.74	0.92	0.5	7.98	5.51	0.84
Llama-3.3-70B								
MT	0.8	0.84	0.80	0.94	0.5	8.61	5.39	0.84
MTOSC	0.8	0.87	0.84	0.92	0.88	7.9	5.18	0.86
Llama-4-Maverick								
MT	0.76	0.84	0.83	0.90	0.5	8.58	5.36	0.85
MTOSC	0.78	0.90	0.91	0.91	0.67	7.82	5.43	0.88
Llama-4-Scout								
MT	0.84	0.88	0.67	0.92	0.17	8.48	5.22	0.83
MTOSC	0.86	0.88	0.70	0.91	0.83	7.65	5.34	0.84
GPT 4.1								
MT	0.92	0.67	0.83	0.97	0.83	9.08	6.15	0.86
MTOSC	0.87	0.80	0.90	0.96	0.67	8.41	6.26	0.89
GPT 4.1 mini								
MT	0.89	0.67	0.86	0.97	0.5	8.97	6.04	0.85
MTOSC	0.83	0.79	0.90	0.96	0.5	8.32	6.07	0.88
GPT 4.1 nano								
MT	0.70	0.51	0.70	0.86	0.67	8.14	5.90	0.70
MTOSC	0.77	0.59	0.82	0.87	0.67	7.72	5.81	0.77
GPT 4o								
MT	0.83	0.56	0.61	0.87	0.83	8.96	5.93	0.72
MTOSC	0.84	0.69	0.79	0.95	0.83	8.32	5.69	0.82
GPT 5								
MT	0.90	0.64	0.96	0.91	0.5	8.81	6.66	0.85
MTOSC	0.92	0.73	1.0	0.89	0.5	8.15	6.45	0.89
GPToss-20B								
MT	0.86	0.59	0.69	0.93	0.83	7.79	5.47	0.77
MT-OSC	0.84	0.61	0.70	0.91	0.67	7.24	5.85	0.76
GPToss-120B								
MT	0.92	0.49	0.86	0.93	0.5	8.51	6.15	0.80
MT-OSC	0.94	0.66	0.82	0.89	0.67	7.88	6.21	0.83
Avg. by dataset								
MT	0.85	0.67	0.76	0.91	0.58	8.67	5.84	0.80
MT-OSC	0.86	0.74	0.81	0.92	0.67	8.01	5.86	0.83

Table 5: Results across datasets and chat models for samples with 6 turns or more using MT-baseline and MT-OSC. (*Spider has a very small sample size of 6 for ≥ 6 turns. Average accuracy per LLM doesn't include Spider due to this reason.)

BFCL ^{shrd}				
		Llama4-	GPT4.1	GPT4o
		mav		
Condenser				
N/A	(MT- baseline)	82.72	82.72	60.49
Llama-3.3-70B		91.36	90.12	79.01
Llama-4-Mav		92.59	85.19	77.78
GPT 4.1		85.19	85.19	88.89
Gemini-2.5-flash		82.72	80.25	72.84

HumanEval ^{shrd}				
		Llama4-	GPT4.1	GPT4o
		mav		
Condenser				
N/A	(MT- baseline)	83.61	68.85	55.74
Llama-3.3-70B		91.36	90.12	79.01
Llama-4-Mav		92.59	85.19	77.78
GPT 4.1		85.19	85.19	88.89
Gemini-2.5-flash		82.72	80.25	72.84

Table 6: Across Condenser Models, performance in (%) of MT-OSC compared to MT-baseline for data samples with at least 6 turns or more in sharded BFCL and HumanEval sets.

D.2 Robustness test results

Table 7 compares the performance between MT-baseline and MT-OSC on top of perturbed dataset samples. Results show that MT-OSC consistency performs on perturbed data as it does on original samples of the same data.

D.3 Decider analysis

Figure 12 shows that chat history tokens are reduced across different datasets when using MT-OSC, both with and without the Decider component, for conversations with six or more turns.

With the Decider, chat history tokens are reduced by an average of 32.19%. Without the Decider, the reduction averages 45.58% across all datasets (even the ones where no sample goes through the condenser in the w/ Decider case).

For datasets where some samples are condensed in w/ Decider case (BFCL, GSM, HumanEval, Refinement, Recollection+, and Expansion+), with Decider reduction averages 44.56% and without Decider reduction averages 52.07%. Note we didn’t include Spider as it has only 6 samples with ≥ 6 turns so we excluded it in the calculation due

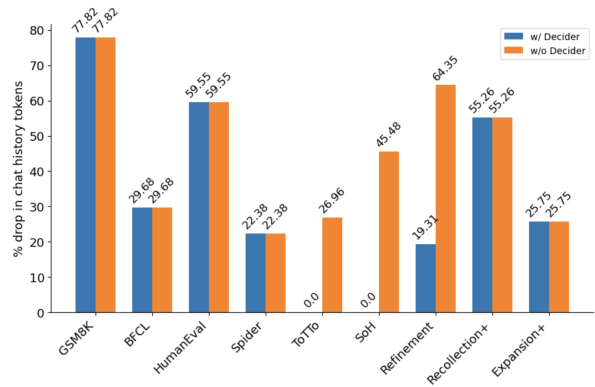


Figure 12: Percentage reduction in chat history tokens in MT-OSC over MT-baseline across datasets with and without Decider component.

to very low number of samples.

Example To complement the Decider ablation results, we share a Sharded ToTTo dataset example (using chat model Llama-3.3-70B) illustrating how default condensation without Decider gate may lose salient multi-turn emphasis signals. In this setting, the final table description depends on information provided across all past turns, including a highlighted-cell view and additional metadata.

Example user-provided conversation turns:

Turn 1: "I am providing you a table. You must produce a short one-sentence description of the table. The description should be at most 30 words.<data>"

Turn 2: "Here’s 10 examples of descriptions of other tables. You can use it as references to understand how to describe the table: <data>"

Turn 3: "I am now giving you a version of the table with some cells highlights."

Turn 4: "Here’s additional metadata about the table. The table_webpage_url is"

Turn 5: "*Here’s additional metadata about the table. The table_page_title is 2012 in AFC."

Turn 6: "Here’s additional metadata about the table. The table_section_title is Events list."

In this conversation, earlier turns center more around “AFC 3”, while turns (4–6) provide page and section context that should be incorporated without overriding the highlight focus.

MT-OSC final output without Decider (always condense): “The table lists events on the 2012 in AFC Wikipedia page under Events list.”

MT-OSC final output with Decider (in this case - skip condensation): “The table highlights AFC 3 event details in the Events list section of the 2012

	1. Repetition Infusion		2. Filler Injection		3. Contextual Diversion	
	MT	MT-OSC	MT	MT-OSC	MT	MT-OSC
GSM ^{shrd}	85.80	86.63	84.77	87.86	82.92	84.77
BFCL ^{shrd}	63.45	70.64	64.02	69.89	59.77	67.24
HumanEval ^{shrd}	70.81	75.89	74.40	79.17	66.67	79.63
Spider ^{shrd}	67.68	70.20	63.13	66.67	57.07	61.62
refinement ^{mte}	5.67	5.66	5.7	5.64	5.01	5.00

Table 7: Comparison of performance between MT and MT-OSC on the three different variants created for robustness testing (1. Repetition Infusion; 2. Filler Injection; 3. Contextual Diversion).

in AFC Wikipedia page."

This example shows a typical Decider benefit: aggressive condensation may at time, for certain chat models, produce a less specific final description by losing or down-weighting earlier emphasis cues in comparison to cues shared in newer turns; the Decider helps avoid this potential failure mode.

D.4 Decider parameter sensitivity analysis

Sensitivity to decider parameters (γ, τ). We sweep $\gamma \in \{0.1, 0.2, 0.3, 0.4\}$ and $\tau \in \{500, 1000, 2000, 3000, 4000\}$ on 10 datasets. For each setting we report (i) the number of samples routed to compression versus not, and (ii) the resulting end-to-end *success* score (reported in each dataset’s native metric). Figure 13 shows the results. Overall, *5/10 datasets are insensitive* to (γ, τ) in our sweep (GSM, BFCL, HumanEval, Spider, Recollection+): both the routing decisions and success are unchanged. Sensitivity is concentrated in *Totto*, where increasing τ generally increases the number of eligible samples compressed and changes the success score; *Refinement* has a similar trend as *ToTTo* but the change in success score is much smaller. *Expansion+* show very small differences; *Summary of Haystack* is almost constant with only a minor change at $\gamma = 0.4$; *Follow-up* is constant since it has ≤ 6 turns, ultimately not getting compressed due to turn limit (MT-OSC at $w=4$ applies to a chat ≥ 6 turns).

These results indicate that the decider hyperparameters primarily control compression aggressiveness on a subset of datasets, while some datasets remain robust, a design allowing for compressing but avoiding risk of information loss which some of these datasets samples portray.

E Condenser and Summarizer examples

Figure 14 presents a sample from the GSM8K^{shrd} dataset in which the MT-baseline run results in

incorrect assumptions and an incorrect final answer. In contrast, MT-OSC successfully addresses these issues and produces the correct solution.

The below example show a sample chat where assistant makes assumptions not provided by the user. The simple summary fails to separate user shared context from assistant shared context. The MT-OSC Condenser module makes a better separation. Keep the context separate is useful so future turns of the conversation are not confused by potential incorrect assumptions made by the assistant model earlier in the chat.

Example original chat: This example shows first four turns on a chat sample from GSM8K^{shrd}.

- **USER:** *Q: how far is Elvis from his house?*
- **ASSISTANT:** *To answer this question, I need more information. The question doesn't provide enough context or details about Elvis's location or the location of his house. Could you please provide more information, such as: Where is Elvis currently located? Where is his house located? Is there a specific distance or direction mentioned? With more context, I can try to help you solve the problem and estimate the distance between Elvis and his house.*
- **USER:** *Q: Elvis begins his journey from his house.*
- **ASSISTANT:** *With this new information, we can start fresh. Since Elvis begins his journey from his house, that means he is currently at his house. So, to answer the original question: "How far is Elvis from his house?" The answer would be: Elvis is 0 miles (or 0 kilometers) away from his house, since he is starting from there.*
- **USER:** *Q: he drives west for 5 hours.*
- **ASSISTANT:** *New development! Since Elvis drives west for 5 hours, we need to consider the distance he covers during this time. However, we don't know his exact speed. Let's assume a moderate speed of 60 miles per hour (mph)*

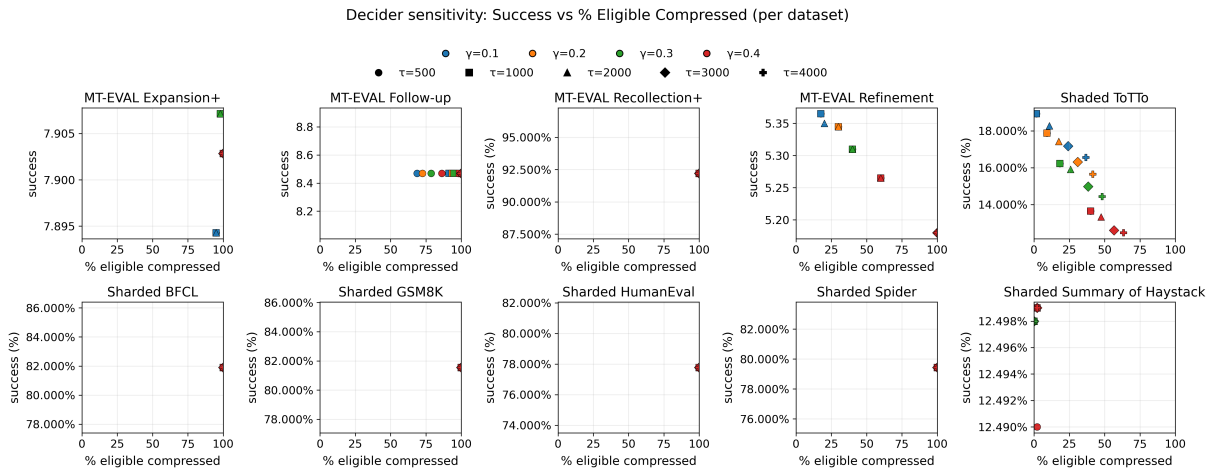


Figure 13: Sensitivity of the decider to γ and τ across datasets. Each panel corresponds to one dataset. Points show, for each (γ, τ) setting, the end-to-end task success score (y-axis; reported in the dataset's native units) versus the number of samples that were routed to compression (x-axis). Color indicates γ and marker shape indicates τ . Flat panels indicate insensitivity to (γ, τ) , while sloped panels highlight datasets where increased compression changes downstream success.

Single-turn (ST) sample

Elvis starts driving from his house and travels west for 5 hours. Then he turns around and travels east for 8 hours. If he was driving at an average speed of 18mph for both parts of the journey, how far is he from his house now?

MT-OSC chat

how far is Elvis from his house? He begins his journey from his house and drives west for 5 hours.

after that, he turns around to change direction.

then, he drives east for 8 hours.

his average speed was 18 mph during the entire trip.

.... Final distance from house:
 $**35** = **3 \times 18** =$
 $**54 miles**$

Multi-turn (MT) chat

how far is Elvis from his house?

Elvis begins his journey from his house.

he drives west for 5 hours.

...Let's use 65 mph as a rough estimate. Distance = Speed \times Time = 65 mph \times 5 hours = 325 miles...

after that, he turns around to change direction.

then, he drives east for 8 hours.

....(8 \times 65mph=520)... 520 miles - 325 miles= 195 miles....

his average speed was 18 mph during the entire trip.

<tries to update calculation but gets confused and ends up using 195 miles in the calculation.>

Figure 14: Example showing MT-baseline vs MT-OSC, where MT-OSC handles incorrect assumptions made by the model earlier in a chat.

for simplicity. This is just an estimate, as the actual speed might vary.
 $\text{Distance} = \text{Speed} \times \text{Time} = 60 \text{ mph} \times 5 \text{ hours} = 300 \text{ miles}$
So, Elvis drives 300 miles west from his house.
To answer the original question: "How far is Elvis from his house?"
The answer would be: Elvis is approximately 300 miles away from his house, in a westerly direction.

- **USER:** 'Q: after that, he turns around to change direction.
A:

- **ASSISTANT:** Plot twist!
Since Elvis turns around to change direction, that means he is now heading in the opposite direction, which is east.
He was previously 300 miles west of his house. Now, he starts driving east, but we don't know for how long or how far.
The distance between Elvis and his house remains the same for now: 300 miles. However, as he starts driving east, this distance will begin to decrease.
We'll need more information about his speed and the time he drives east to determine the new distance between Elvis and his house.

Simpler summarizer output: This example shows the output of the chat shared above when passed through the simple summarizer.

- **USER:** The conversation starts with a question about the distance between Elvis and his house. It is then clarified that Elvis begins his journey from his house, so he is 0 miles away. He then drives west for 5 hours, covering an estimated distance of 300 miles at a moderate speed of 60 mph. After that, he turns around to change direction, heading east, but the time and distance driven east are not specified.

- **ASSISTANT:** The assistant responds by initially requesting more context about Elvis's location. After each update, the assistant recalculates the distance between Elvis and his house: first it's 0 miles when he starts at home, then approximately 300 miles after driving west for 5 hours at 60 mph, and finally this distance begins to decrease as he heads back east but the exact new distance is unknown without more information on his speed and time driven east.

MT-OSC Condenser output: This example shows the output of the Condenser component for the original chat shared above.

- **USER:** how far is Elvis from his house? He begins his journey from his house, drives west for 5 hours and then turns around to change direction.

- **ASSISTANT:** Elvis is initially 0 miles away from his house. After driving 300 miles west, he is approximately 300 miles away. When he turns around, the distance remains 300 miles until he starts driving east.