TOKMEM: TOKENIZED PROCEDURAL MEMORY FOR LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models rely heavily on prompts to specify tasks, recall knowledge and guide reasoning. However, this reliance is inefficient as prompts must be re-read at each step, scale poorly across tasks, and lack mechanisms for modular reuse. In this paper, we aim to store and recall seen procedures efficiently. We introduce TokMem, a tokenized procedural memory that stores recurring procedures as compact, trainable embeddings. Each memory token encodes both an address to a procedure and a control signal that steers generation, enabling targeted behavior with constant-size overhead. To support continual adaptation, TokMem keeps the backbone model frozen, allowing new procedures to be added without interfering with existing ones. We evaluate TokMem on 1,000 tasks for atomic recall and multi-step function-calling for compositional recall, where it consistently outperforms retrieval-augmented generation while avoiding repeated context overhead, and fine-tuning with far fewer parameters. These results establish TokMem as a scalable and modular alternative to prompt engineering and fine-tuning, offering an explicit procedural memory for LLMs.[1]

1 INTRODUCTION

Large language models (LLMs) have become the foundation of modern natural language processing, powering a wide range of applications in text understanding, generation, and coding (Brown et al., 2020; Grattafiori et al., 2024). Prompting is a widely adopted way to steer LLM behavior, where in-context learning enables adaptation to new tasks without parameter updates (Brown et al., 2020). Consequently, prompt and context engineering has emerged as a dominant mechanism for specifying tasks, obtaining relevant information, and guiding multi-step reasoning or tool invocation (Wei et al., 2022b; Yao et al., 2023; Sahoo et al., 2025).

Despite its success, this reliance on long prompts is inherently inefficient. Constructing and maintaining prompts are labor-intensive and difficult to scale across many tasks (Liu et al., 2023). At inference, long prompts increase computational cost because the attention mechanism scales quadratically with sequence length (Vaswani et al., 2017), and they reduce the effective context window available for inputs and outputs, often leading to truncation and loss of details (Liu et al., 2024a). These limitations make it difficult to manage expanding tasks and to execute procedures efficiently.

To address these issues, recent approaches offload prompts into retrieval-based memory. Retrieval-augmented generation (RAG) (Lewis et al., 2020) and memory systems such as MemGPT(Packer et al., 2023) fetch and reinsert documents or conversational state at inference time. While retrieving in-context learning demonstrations (Wei et al., 2022b) can provide procedural cues that guide the model's behavior, the mechanism still largely aligns with declarative memory in cognitive science: knowledge remains as explicit text that must be repeatedly interpreted. This creates two challenges: (1) retrieved content still occupies the context window, reintroducing quadratic compute and truncation pressure, and (2) frequently used procedures are repeatedly re-read as text rather than compiled into compact, reusable procedures, missing the compression opportunity suggested by minimum description length principles (Grünwald, 2007).

We propose **Tok**enized **Mem**ory (**TokMem**), an explicit form of procedural memory that encodes recurring procedures as compact, trainable tokens while keeping the backbone frozen. Here, a *proce-*

[1] Our anonymized code is available here.

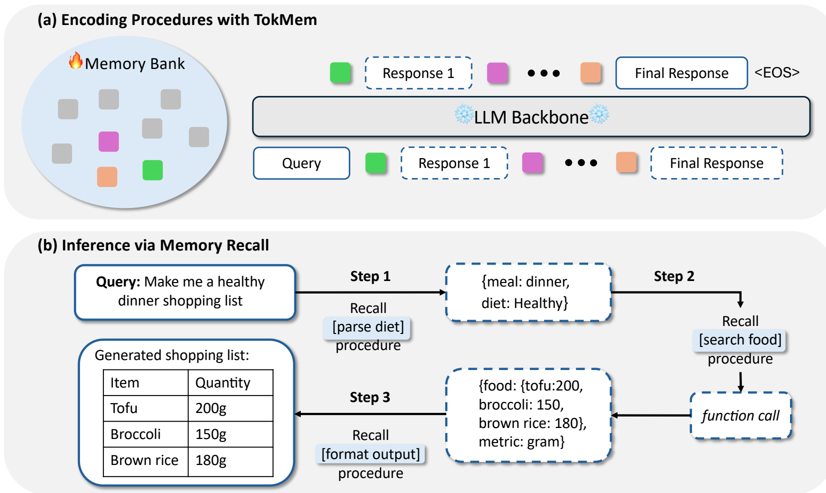

Figure 1: Overview of TokMem. (a) New memory (colored) tokens are interleaved with text sequences, learning with next-token-prediction while the LLM backbone remains frozen. (b) An example of inference, a query recalls and chains memory tokens (parse, search, format), enabling multi-step procedural behavior without long prompts.

dure means a reusable, context-response mapping that encodes a specific task behavior, inspired by physiological research Anderson & Lebiere (1998). Compared with factual knowledge, procedural knowledge (such as riding a bicycle) is often more nuanced and sophisticated.

In our TokMem, each memory token serves both as an address to a procedure and as a control signal that steer generation, enabling targeted behavior with constant-size overhead. Specifically, rather than front-loading procedures as long prompts, TokMem integrates memory tokens directly into the generation process. As shown in Figure 1b, memory tokens can be invoked and chained across stages: after producing one response segment, the model retrieves the next relevant token, which conditions the next stage, enabling the composition of multi-step behaviors such as parsing, searching, and formatting.

A key advantage of TokMem is that its memory tokens are parameter-isolated from the backbone. This design ensures that the learned procedural knowledge is fully stored in dedicated tokens, allowing new procedures to be added without interfering with existing ones. TokMem thus naturally supports continual learning, where the model can accumulate procedural skills over time while preserving stability. This capability mirrors human procedural memory, where skills are gradually acquired through practice and later invoked by contextual cues (Anderson & Lebiere, 1998). In this way, TokMem enables both efficient learning and continual expansion of procedural knowledge.

We evaluate TokMem in two complementary settings. In the atomic memory recall setting, each task from Super-Natural Instructions (Wang et al., 2022a) is treated as a distinct procedure, TokMem stores and retrieves 1,000 such procedures efficiently, without catastrophic forgetting. In the compositional memory recall setting based on function-calling tasks (Liu et al., 2024b), each tool invocation is modeled as an atomic procedure, and solving a query requires chaining multiple procedures together. TokMem supports this process by composing memory tokens, enabling the model to assemble procedures into coherent multi-step behaviors. Across both settings and multiple LLM backbones, TokMem consistently outperforms retrieval-based baselines and surpasses parametric fine-tuning while using far fewer trainable parameters.

2 METHOD

We begin by reviewing how Transformer-based LLMs process input sequences and then describe how TokMem departs from existing approaches to enable procedural memory.

2.1 TEXTUALIZED CONTEXT ENGINEERING

A Transformer (Vaswani et al., 2017) processes a sequence of tokens $(a_1, \ldots, a_n) \in \mathbb{N}^n$, where each a_i is an integer representing the index of a token (usually sub-words). The model retrieves the corresponding embedding vector from the embedding layer and produces an input sequence $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \in \mathbb{R}^{n \times d}$, which is then consumed to predict the next token in sequence.

Recent advances in prompting can be viewed as *textualized context engineering*, where the goal is to carefully choose input tokens that steer the model toward improved behavior. For example, chain-of-thought prompting (Wei et al., 2022a) augments input with intermediate reasoning steps to strengthen logical inference. Retrieval-based methods such as RAG (Lewis et al., 2020) and memory-augmented approaches like MemGPT (Packer et al., 2023) provide relevant information in text form by retrieving external memory. While effective, these approaches are costly: they exhaust limited model context window, and significantly increase compute due to the quadratic complexity of self-attention.

2.2 TOKMEM: PROCEDURAL MEMORY AS A TOKEN

Our key idea is that frequently reused procedures can be effectively "compressed" and stored by encoding them into an internalized memory token, bypassing repeated textual specification. Consider l memory tokens. They are added to the vocabulary as special tokens, each represented by an embedding. Thus, they form a memory bank of l special embeddings:

$$M = \begin{bmatrix} \boldsymbol{m}_1^\top \\ \vdots \\ \boldsymbol{m}_l^\top \end{bmatrix} \in \mathbb{R}^{l \times d}, \qquad \boldsymbol{m}_i \in \mathbb{R}^d. \tag{1}$$

Each $\boldsymbol{m}_i$ is a trainable vector with no direct textual translation and represents a unique procedure. For simplicity, we label each memory token $\boldsymbol{m}_i$ to have a special index $a_{m_i} \in \mathbb{N}$.

To connect these tokens with training, we first describe a single training instance. We define a *procedure–response* pair, where a procedure is invoked by a memory token $\boldsymbol{m}_i$, and the corresponding response is a sequence of textual tokens $(r_{i1}, \ldots, r_{in}) \in \mathbb{N}^n$ that encodes the information implied by the procedure. For example, the response could contain arguments for tool calling or certain pattern of text following the procedure.

In training, a procedure-response pair is represented by the concatenation of a procedural memory token a_{m_i} and the embeddings of $(r_{i1}, \ldots, r_{in})$ together. Each training instance may contain multiple turns of procedure-response pairs, modeling tasks that require multi-step reasoning or composition for a query q. Formally, the sequentialized training sequence has the layout

$$\boldsymbol{a} = \big(q_1, \ldots, q_k,\ \underbrace{a_{m_i}, a_{r_{i1}}, a_{r_{i2}}, \ldots,}_{\text{procedure-response pair}}\ \underbrace{a_{m_j}, a_{r_{j1}}, a_{r_{j2}}, \ldots,}_{\text{procedure-response pair}}\ \ldots\big). \tag{2}$$

We adopt the standard next-token prediction loss:

$$\mathcal{L}(\boldsymbol{a}; M) = -\sum_{i>k} \log \Pr(a_i \mid \boldsymbol{a}_{<i}; M). \tag{3}$$

During optimization, the memory embeddings $(\boldsymbol{m}_1, \ldots, \boldsymbol{m}_l)$ are trainable and shared in the input embedding layer and LM head, whereas the the pre-trained text token embeddings in the input embedding layer and LM head, as well as the backbone remains frozen. For the training corpus, each memory embedding is exposed to varied queries and responses, allowing it to learn the underlying procedure representation. We visualize our training process in Figure 1a.

2.3 INFERENCE WITH MEMORY TOKENS

At inference, TokMem recalls procedures through memory routing and conditional response generation, where *routing* refers to the selection of the appropriate memory token for a given query. Specifically, for a query $q = (q_1, \ldots, q_k)$, the model predicts a distribution over memory tokens from its final hidden state h_k:

$$P(a_{m_i} | q) \propto \exp(\text{logit}(m_i | h_k)), \tag{4}$$

We choose the memory token with the highest probability and append it to the sequence as $(q_1, \ldots, q_k, a_{m_i})$. Then, the model generates the response autoregressively. For queries that require calling multiple procedures, after generating one response segment, the model may predict another memory token and generate the next response, as seen in Figure 1b. Notably, when a query does not correspond to any learned procedure, the logits of all memory tokens may remain low, and the model naturally defaults to generating regular text tokens.

In summary, the model decides whether to recall memory tokens (and how many) based on its training, as shown in Eqn. (2).

2.4 Stabilizing New Memories

TokMem allows new procedures to be added incrementally to the memory bank, mimicking how humans continually form new procedural memories without disrupting existing skills (Anderson & Lebiere, 1998; Squire, 2009). This design enables practical deployment scenarios, where an LLM can steadily accumulate routines across domains and tasks rather than retraining from scratch.

However, adding new tokens poses stability challenges. If all procedural memories are introduced at once, the model risks overfitting to spurious patterns. Conversely, when new embeddings are added gradually, they often develop inflated norms that dominate routing logits and suppress older memories. To address this, we introduce renormalization, which is a lightweight post-update calibration to the memory bank $M \in \mathbb{R}^{l \times d}$.

Let A and I denote the indices of the active (new) and inactive (existing) procedural memories, respectively. We estimate the prevailing scale from the inactive set:

$$\bar{n}_I \;=\; \mathrm{mean}_{j \in I} \big\| \boldsymbol{m}_j \big\|_2, \tag{5}$$

and rescale each active active embedding as

$$\boldsymbol{m}_i \;\leftarrow\; \boldsymbol{m}_i \cdot \frac{\bar{n}_I}{\|\boldsymbol{m}_i\|_2 + \varepsilon}, \qquad i \in A. \tag{6}$$

This operation preserves the directions of newly added embeddings while aligning their magnitudes to the established scale of the memory bank, ensuring smooth integration without overwhelming the routing dynamics. The computational overhead is negligible, scaling as $O(|A|d)$.

3 Experiments

We evaluate TokMem in two complementary scenarios. In the *atomic memory recall* setting, each task from the Super-Natural Instructions dataset (Wang et al., 2022a) is framed as a standalone procedure, where a query directly maps to the desired response. In the *compositional memory recall* setting that involves calling multiple procedures, evaluated on the function-calling dataset (Liu et al., 2024b), where invoking a tool is treated as a procedure and solving a query requires composing several function calls. Experiments are conducted on Qwen (Qwen & et al., 2025) and Llama (Grattafiori et al., 2024) model families, ranging from the 0.5B-parameter Qwen to the 8B-parameter Llama model.

3.1 Experimental Setup

Baselines. Across both settings, we compare TokMem with textualized context engineering, retrieval-augmented memory, and parameter-efficient fine-tuning.

- **Base**: In the atomic setting, the model answers queries without demonstrations, providing a non-parametric lower bound highlighting the need to recall task knowledge.
- **ICL**: In the compositional setting, we augment input with all tool descriptions and prepend two compositional procedure–response demonstrations, representing a context engineering baseline.
- **RAG**: We retrieve relevant demonstrations or tool usages with Sentence-BERT (Reimers & Gurevych, 2019) and prepend them to the query, following memory-augmented generation (Packer et al., 2023; Chhikara et al., 2025; Xu et al., 2025).

- **Fine-tuning**: We use low-rank adapters (Hu et al., 2022) inserted into the query and value projections of the transformer, updating millions of parameters depending on model sizes.[2] This serves as a parametric form of procedural memory, but is prone to forgetting as new tasks are introduced.
- **Replay Memory**: To mitigate catastrophic forgetting in fine-tuning, we follow the idea of experience replay (Mnih et al., 2015) by maintaining a buffer of previously seen tasks or tools and mixing them with the current training data.

Training Details. All methods are implemented in HuggingFace Transformers and trained on a single NVIDIA A6000 GPU with 48GB memory using mixed-precision (bfloat16) training. The backbone models remain frozen; for fine-tuning, only the adapter weights (rank $r = 8$) are updated, while for TokMem, only the embeddings of the newly added procedure IDs are trainable. Tokenizer vocabulary is expanded with these procedure IDs, and their embeddings are initialized by averaging the pretrained embeddings (Hewitt, 2021). For Replay Memory, we mix 20% of replayed samples with the current batch, using a buffer of 500 examples refreshed every 10 tasks in the atomic setting and $1{,}000$ examples updated each round in the compositional setting.

We optimize with AdamW using a learning rate 5×10^{-5} for fine-tuning and 5×10^{-3} for TokMem; weight decay is 10^{-2} for fine-tuning and zero for TokMem. Training runs for one epoch with batch size 4 and maximum sequence length 1024, using teacher forcing and applying the loss only to memory-token and response positions.

Evaluation methods. We evaluate TokMem from two perspectives: (1) Memory token routing accuracy, which measures whether the correct memory tokens are selected, and (2) Tasks performance, which measures the generation performance for the task (such as Rouge-L and F1), detailed in the following experiments.

3.2 Atomic Memory Recall

Dataset Details. We evaluate on the Super-Natural Instructions (SNI) dataset (Wang et al., 2022a), which provides diverse QA-style natural language tasks. Here, each task is treated as an individual procedure: a query directly invokes the learned procedure to produce the desired response. We sample $1{,}000$ English tasks, each task contains 500 training and 50 test examples. To reflect how memories are typically acquired over time, we introduce tasks sequentially during training rather than all at once, but training samples of each task are shuffled. We scale the number of tasks from 10 up to $1{,}000$, and record checkpoints after training on $\{10, 50, 200, 500, 1{,}000\}$ tasks. This resembles incremental domain adaptation (Asghar et al., 2020), where at each checkpoint, the performance is evaluated across all previously seen tasks. Additional tasks details are provided in Appendix D.1.

We follow Wang et al. (2022a) and use Rouge-L (Lin, 2004) to evaluate generation quality. For the methods with memory routing (RAG and TokMem), we additionally report accuracy, reflecting whether the correct procedure was selected and applied.

Decoupled Memory Embeddings. We also include an ablation variant where memory tokens are decoupled to an address token and a steering token. This decoupling separates the roles of a memory token and also increase the capacity of TokMem. We refer to this variant as TokMem with decoupled embeddings (TokMem+DC), with further details provided in Appendix A.

Results and Findings. TokMem provides the most consistent and scalable performance across models and task scales. As shown in Table 1, non-parametric methods such as Base shows stable but fail to achieve competitive performance. RAG performs reasonably well on when memory is not heavy but quickly degrade as the number of task memory increases, indicating its sensitivity to retriever quality. Parametric methods such as fine-tuning achieve stronger initial accuracy but suffer from forgetting as tasks accumulate; replay memory alleviates this issue but still falls short of TokMem. By contrast, we see that TokMem maintains high accuracy with minimal performance drop when acquiring new task memories, achieving the best average results across all settings.

[2] In our preliminary study, we found that adding training parameters by applying low-rank adapters to more projections did not improve performance while greatly increasing computational cost. We therefore follow Hu et al. (2022) and apply LoRA to the query and value projections only.

Table 1: Atomic recall performance on SNI, reported with Rouge-L. TokMem consistently outperforms fine-tuning and RAG across models and scales, maintaining strong results even at 1,000 tasks.

Model	Method	Number of Tasks 10	50	200	500	1000	Avg.
Qwen 2.5 0.5B	*Base*	33.9	39.0	38.8	39.1	38.5	37.9
	RAG	50.4	43.2	38.8	36.2	34.7	40.7
	Fine-Tuing	52.4	48.0	40.6	41.7	43.2	45.2
	Replay Memory	52.4	49.5	47.2	47.7	46.7	48.7
	TokMem	52.8	51.3	49.3	50.2	50.0	50.7
	TokMem+DC	53.8	50.5	50.2	50.9	50.0	**51.1**
Llama 3.2 3B	*Base*	16.6	19.9	20.0	18.7	18.2	18.7
	RAG	60.0	48.7	45.8	42.3	39.9	47.3
	Fine-Tuing	67.1	59.1	59.5	58.4	57.9	60.4
	Replay Memory	67.1	61.1	60.6	61.4	60.0	62.0
	TokMem	68.0	62.3	61.2	61.5	61.5	**62.9**
	TokMem+DC	68.8	62.5	58.7	61.7	61.1	62.6
Llama 3.1 8B	*Base*	27.2	27.8	30.4	29.6	29.5	28.9
	RAG	63.8	53.9	49.1	45.3	42.6	50.9
	Fine-Tuing	75.8	64.3	63.2	58.7	61.6	64.7
	Replay Memory	75.8	65.2	64.5	63.4	63.6	66.5
	TokMem	75.4	65.5	65.1	64.4	64.8	**67.0**
	TokMem+DC	75.6	65.8	63.7	64.2	64.4	66.7

Table 2: Task routing accuracy. TokMem achieves near-perfect routing accuracy at 1,000 tasks, far exceeding RAG retriever whose accuracy falls below 80%.

Model	Method	Number of Tasks 10	50	200	500	1000
Sentence-Bert	*RAG*	99.6	92.6	88.7	83.2	79.7
Qwen 2.5 0.5B	*TokMem*	99.4	98.6	97.4	96.9	94.7
	TokMem+DC	**99.4**	**99.2**	**98.4**	**97.2**	**96.1**
Llama 3.2 3B	*TokMem*	**100.0**	**99.9**	**98.3**	**97.1**	**96.1**
	TokMem+DC	99.8	99.3	97.2	96.2	95.4
Llama 3.1 8B	*TokMem*	**99.8**	**99.6**	**98.9**	**97.7**	**97.5**
	TokMem+DC	99.7	99.4	97.8	97.2	97.2

The decoupled variant (TokMem+DC) yields modest gains for the smaller Qwen 0.5B model but no improvement for larger Llama models, and in some cases it underperforms TokMem when scaling to many tasks. Overall, although TokMem+DC is a tempting variant, it does not provide additional benefits. We advocate for the simple yet effective TokMem (without DC).

Table 2 further highlights TokMem's robustness in memory routing. Its accuracy remains above 94% even at 1,000 tasks with the smallest 0.5B model, significantly outperforming the Sentence-BERT retriever used in RAG, whose accuracy drops below 80% when have the stress to route for 1,000 tasks. This high-fidelity memory routing enables TokMem to sustain strong performance without relying on external retrieval mechanisms or fine-tuning, demonstrating its advantage in continual and large-scale task acquisition.

Analysis of Training Efficiency. We compare the training sample efficiency of LoRA fine-tuning and TokMem on the first 10 tasks from SNI, a mixture setup that removes the impact of forgetting. We set the adapter rank to $r = 1$, which helps prevent overfitting and aligns its parameter scale with TokMem. The result in Figure 2 shows that TokMem consistently achieves higher performance than fine-tuning across all sample budgets, with the greatest advantage appearing in the low-data regime. The decoupled variant (TokMem+DC) offers small but consistent improvements, particularly when more samples are available. Overall, these results highlight TokMem's ability to learn new procedures effectively with limited data, making it a both parameter and data efficient approach to memory acquisition.

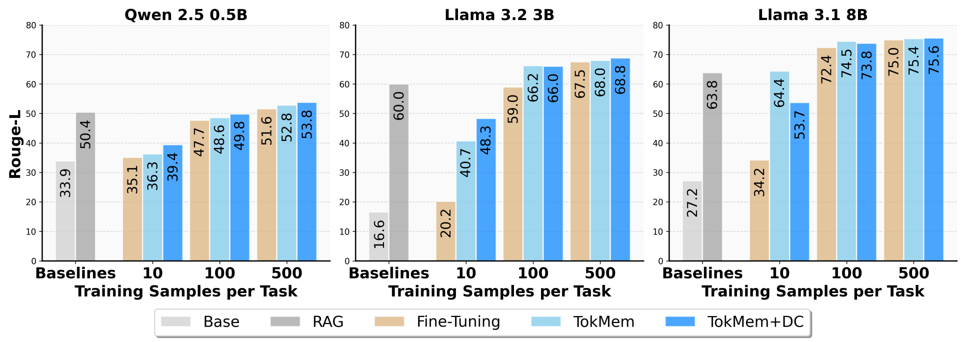

Figure 2: Sample efficiency on a 10-task mixture from SNI. TokMem consistently outperforms fine-tuning in the low-data regime. TokMem can surpass RAG with only 10 training samples, demonstrating strong few-shot learning capability.

Table 3: Compositional tool-use performance on APIGen. TokMem achieves strong tool selection and argument F1 across multiple calls, outperforming ICL and RAG with lower input-augmentation complexity, and surpassing fine-tuning with far fewer trainable parameters.

			Tool Selection				**Argument**			
Model	**Method**	**#Params**	2 calls	3 calls	4 calls	**Avg.**	2 calls	3 calls	4 calls	**Avg.**
Llama 3.2 1B	*ICL*	–	27.6	11.1	10.5	16.4	0.6	0.7	0.0	0.4
	RAG	–	29.5	10.8	10.5	16.9	7.2	1.0	0.0	2.7
	Fine-Tuing	0.85M	10.4	9.5	7.0	9.0	77.3	72.6	55.8	68.6
	TokMem (w/o adapt)	0.10M	86.8	80.9	90.8	86.2	68.9	61.1	73.0	67.7
	TokMem (w/ adapt)	0.10M	98.4	98.0	98.9	**98.4**	84.3	84.3	87.8	**85.5**
Llama 3.2 3B	*ICL*	–	66.8	59.2	59.6	61.9	42.2	42.3	38.8	44.1
	RAG	–	78.1	71.2	69.3	72.8	54.8	53.1	62.7	56.9
	Fine-Tuing	2.29M	98.7	98.1	96.8	97.9	87.9	86.6	82.9	85.8
	TokMem (w/o adapt)	0.15M	82.6	79.3	67.2	76.4	65.4	57.2	50.2	57.6
	TokMem (w/ adapt)	0.15M	99.2	98.2	100.0	**99.2**	85.9	86.7	88.3	**86.3**
Llama 3.1 8B	*ICL*	–	79.7	72.9	75.4	76.0	51.5	52.6	57.3	53.8
	RAG	–	79.6	75.3	93.0	82.6	53.3	57.1	69.2	59.9
	Fine-Tuing	3.41M	98.8	97.2	98.2	98.1	87.7	86.8	88.2	87.6
	TokMem (w/o adapt)	0.20M	84.9	82.0	81.6	82.8	65.8	56.7	65.9	62.8
	TokMem (w/ adapt)	0.20M	99.4	97.9	100.0	**99.1**	88.1	86.5	93.4	**89.3**

3.3 COMPOSITIONAL MEMORY RECALL

Dataset Details. We construct a benchmark from the APIGen dataset (Liu et al., 2024b) by sampling the 50 frequently used tools. Here, each tool invocation is treated as an atomic procedure, and solving a query requires composing multiple such procedures. We synthesize 5,000 training queries and 500 test queries, both capped at four calls. Details of the dataset can be found in Appendix D.2

We report performance using two F1 metrics: (i) Tool Prediction F1, which measures whether the correct tools are invoked; and (ii) Argument Generation F1, which evaluates the correctness of function call arguments. For robustness to semantic equivalence, both gold and predicted outputs are normalized into Abstract Syntax Trees before scoring (Patil et al., 2025).

Adaptation for Compositionality. We found that TokMem benefits from a brief adaptation phase that exposes the backbone with the compositional structures of memory tokens. Concretely, we fine-tune the backbone on a held-out auxiliary tool set using the same LoRA fine-tuning setup as the baseline. The adapted weights are then merged, after which the backbone remains frozen for memory acquisition and evaluation (see Appendix B.1 for details). Note that the auxiliary tool set is different from the evaluation set, so it does not break our frozen-backbone setup.

The proposed adaptation initialization teaches the model on how to interleave responses with memory tokens. We thus consider it a part of the TokMem approach for compositional tasks and use TokMem with adaptation as the default configuration. We also analyze the effect of this adaptation

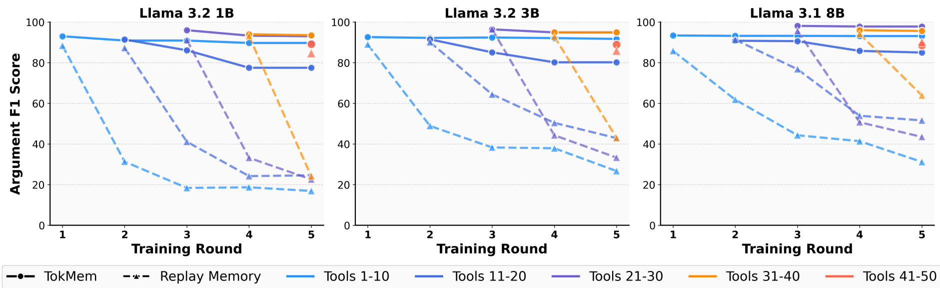

Figure 3: Forgetting analysis in continual adaptation. As new tools are introduced, fine-tuning with replay memory suffers sharp drops on earlier tasks, while TokMem maintains stable performance. Larger models show stronger retention due to greater capacity.

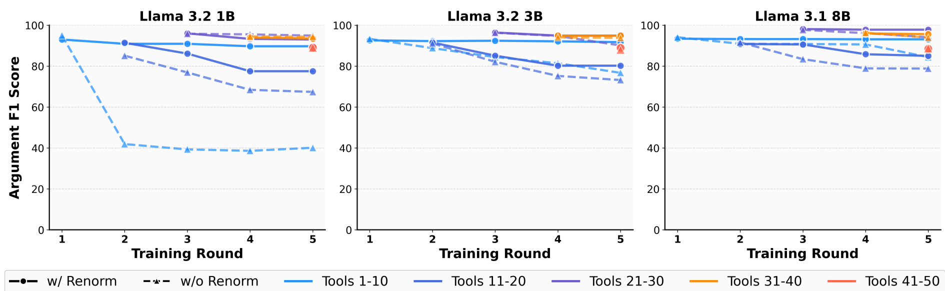

Figure 4: Effect of renormalization on TokMem. Without renormalization, new tokens dominate and older ones are forgotten, particularly in smaller models with limited embedding capacity. Renormalization effectively mitigates this by balancing norms across tokens.

phase on the fine-tuning baseline in Appendix B.2. We find that it degrades baseline performance and we excludes it for a stronger baseline.

Results and Findings. Table 3 shows that TokMem achieves the strongest overall performance. Without adaptation for compositionality, it outperforms RAG while avoiding the added complexity from an external retrieval mechanism. Non-parametric baselines such as ICL and RAG perform poorly on both tool prediction and argument generation, particularly with the smaller Llama 1B model, likely due to its weak instruction-following ability.

Compared with parametric baselines, TokMem consistently matches or surpasses LoRA fine-tuning while requiring an order of magnitude fewer trainable parameters. For example, on the Llama 8B model, LoRA requires 3.41M training parameters while TokMem only needs 0.2M to achieves higher performance.

Notably, TokMem exhibits stronger interpretability between tool selection and argument generation, with improvements in the former translating directly into the latter. By contrast, LoRA fine-tuning shows weaker alignment. For example, as seen with the 1B model, it often generates plausible arguments even when tool selection is incorrect, indicating that its argument generation is not properly grounded in the chosen tools.

TokMem also shows significantly better compositional ability than fine-tuning, successfully executing queries with more function calls than it has seen during training. Detailed results are provided in Appendix B.4.

Analysis of Forgetting. We compare TokMem with replay memory in a continual adaptation setting, where tools are introduced sequentially over five training rounds (e.g., tools 1–10 in the first round, 11–20 in the second, and so forth). As shown in Figure 3, we see that replay memory struggles to prevent catastrophic forgetting as new tools are introduced. By contrast, TokMem maintains higher performance across tool groups, with only mild declines that primarily reflect the growing

Table 4: Comparison of TokMem vs. prefix tuning on memorizing text from the *Fanfics* dataset. TokMem converges faster and achieves lower perplexity than prefix tuning, particularly with few memory tokens.

	1024 tokens		2048 tokens		4096 tokens	
Method	Steps@90%Best ↓	PPL ↓	Steps@90%Best ↓	PPL ↓	Steps@90%Best ↓	PPL ↓
Prefix tuning-1	1700	3.81	2300	8.77	2200	14.32
TokMem-1	**1200**	**3.28**	**1400**	**7.07**	**1700**	**12.27**
Prefix tuning-2	**500**	1.13	1700	3.51	1800	8.38
TokMem-2	600	**1.09**	**1300**	**2.75**	**1700**	**7.21**
Prefix tuning-5	300	1.07	500	1.17	1400	2.39
TokMem-5	**200**	**1.03**	**500**	**1.15**	**1400**	**1.91**

number of tools. Larger models exhibit better retention for both approaches, likely due to their expanded parameter capacity, which reduces the risk of interference with previously learned tools.

We further investigate the effect of the renormalization step introduced in Section 2.4 on newly added memory tokens, whose norms may otherwise dominate older tokens in the softmax (analyzed in Appendix B.5). As seen in Figure 4, TokMem without renormalization shows noticeable forgetting especially when the size of the model is small. However, larger models are more robust to forgetting even without renormalization, again due to their greater embedding capacity. Overall, renormalization improves TokMem's resistance to forgetting by balancing routing between both new and old memory tokens. Additional analysis on the benefits of keeping the backbone frozen for continual memory acquisition is provided in Appendix B.3.

3.4 Analysis on Memory Placement

An important design choice in TokMem is the placement of memory tokens within the input sequence, which directly influences how the backbone model attends to and integrates procedural knowledge. While TokMem introduces a memory routing mechanism for generating tokens, its effectiveness also depends on this placement strategy. In the absence of routing, TokMem reduces to a prompt-tuning method (Li & Liang, 2021; Lester et al., 2021) with learnable embeddings, but it distinctively adopts an *infix* placement: `query` ⊕ `MEM` ⊕ `response`. Our experiments indicate that this infix design allows memory tokens to be activated after the context has been encoded, enabling context-aware conditioning and natural composition of multiple procedures.

However, it is unclear whether this memory placement is strictly better than the more common *prefix* formulation: `MEM` ⊕ `query` ⊕ `response` used in prior prompt-tuning work, where prefix tokens influence generation without having observed the query. To study the impact of placement, we compare prefix and infix placements under matched token budgets in the single-task setting.

Setup. We compare TokMem with infix memory placement against prefix tuning by stress-testing the capacity of memory tokens (Sastre & Rosá, 2025) using the recent *Fanfics* dataset collected after the pretraining of LLMs (Kuratov et al., 2025). We fix the sequence length to 128 tokens and vary the batch size from 8 to 32, compressing batches of 1024 to 4096 response tokens into 1 to 5 memory tokens. For each sequence, we prepend a randomly generated query that serves only as a marker to distinguish the two placements, while the actual target to be learned remains the response.

We measure learning speed using Steps@90%Best, defined as the number of training steps (evaluated every 100 steps) required to reach 90% of the best perplexity. Results are averaged over five runs. Additional experiments evaluating generalization on a math reasoning dataset are provided in Appendix C.

Results. Table 4 shows that TokMem consistently achieves lower perplexity and often converges faster than prefix tuning. With a single token, TokMem reaches 90% of the best perplexity roughly 30% sooner than prefix tuning, indicating that conditioning memory after the query helps the model learn more efficiently. Interestingly, when more tokens are available (e.g., five tokens), the performance gap narrows. This suggests that prior work (Li & Liang, 2021), which typically uses dozens or even hundreds of tokens, may have underestimated the importance of memory placement in low-token regimes, where each token must compress more procedural information.

4 Related Work

Equipping LLMs with memory has been explored through multiple directions. Most existing approaches emphasize declarative memory, where the objective is to store and retrieve explicit information such as facts or conversation history (Packer et al., 2023; Chhikara et al., 2025; Zhong et al., 2024). In contrast, parameter-based approaches internalize task-specific behaviors within model parameters, resembling procedural memory. TokMem builds on this latter view while emphasizing modularity and compositionality.

Text-based External Memory. A common approach is to externalize memory as textual content retrieved at inference time. Retrieval-augmented generation (RAG)(Lewis et al., 2020) and its variants(Guu et al., 2020; Karpukhin et al., 2020; Borgeaud et al., 2022; Khandelwal et al., 2020) attach relevant textual chunks during inference, while RET-LLM (Modarressi et al., 2023) encodes knowledge as symbolic triplets. Building on these ideas, more recent systems such as MemGPT (Packer et al., 2023), Mem0 (Chhikara et al., 2025), and A-Mem (Xu et al., 2025) extend these ideas to conversational settings through hierarchical or summarization-based memory states. While effective for factual recall, these approaches are not optimized for procedural control and often incur significant inference-time overhead due to the re-read of textual memory.

Parameter-based Memory. Another line of work encodes memory directly into model parameters. Fine-tuning and multitask instruction tuning (Wei et al., 2022a; Sanh et al., 2021), as well as parameter-efficient variants such as LoRA (Hu et al., 2022) allow models to acquire new procedures, but task knowledge is entangled. Mixture-of-LoRAs (Feng et al., 2024) address the entanglement issue but the mixtures are typically invoked independently and are not designed for memory composition. MemoryLLM (Zhong et al., 2024) introduces latent memory pools but remian entangled. Prompt-based methods such as prompt tuning (Lester et al., 2021; Wu et al., 2025) store knowledge implicitly as global embeddings without selective routing, and L2P (Wang et al., 2022b) introduces modular prompt pools but still relies on an external controller to determine which prompts are retrieved. Prompt compression methods (Mu et al., 2023; Chevalier et al., 2023) compress prompts into one-size-fits-all representations, which may distort prompt information. ToolGen (Wang et al., 2025) compresses tools into virtual tokens but focuses on post-training the backbone through multi-stage fine-tuning. By contrast, TokMem keeps the backbone frozen, and introduces discrete memory units that can be added or composed without retraining, supporting continual adaptation.

Compositional Memory. A complementary direction explores how models compose skills from simpler building blocks. Chain-of-thought prompting (Wei et al., 2022b) and tool-augmented reasoning frameworks such as Toolformer (Schick et al., 2023) enable multi-step reasoning, but rely on textual instructions that must be re-interpreted at each step. Modular parameter methods (Rosenbaum et al., 2018; Pfeiffer et al., 2021) create specialized adapters that can be recombined, but composition requires parameter merging or heuristic routing. TokMem differs by representing procedures as discrete tokens that can be chained directly in context, enabling lightweight parameter-isolated composition.

5 Conclusion

We introduced Tokenized Memory (TokMem), a parameter-efficient framework that encodes procedural memory as compact tokens. TokMem enables selective recall and compositional use of procedures without modifying backbone parameters, achieving strong performance across multitask and tool-augmented reasoning benchmarks.

Future directions are discussed in Appendix E, including reinforcement learning for stronger compositional generalization, and personalization through user-specific memory banks. These extensions pave the way for scalable, compact, and user-adaptive memory systems in large language models.

REFERENCES

John R. Anderson and Christian Lebiere. *The Atomic Components of Thought*. Lawrence Erlbaum Associates, 1998.

Nabiha Asghar, Lili Mou, Kira A. Selby, Kevin D. Pantasdo, Pascal Poupart, and Xin Jiang. Progressive memory banks for incremental domain adaptation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BkepbpNFwr.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*, 2022. URL https://arxiv.org/abs/2112.04426.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020. URL https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3829–3846. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.emnlp-main.232/.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory, 2025. URL https://arxiv.org/abs/2504.19413.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Yu Han, and Hao Wang. Mixture-of-LoRAs: An efficient multitask tuning for large language models. *arXiv preprint arXiv:2403.03432*, 2024. URL https://arxiv.org/abs/2403.03432.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, and et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Peter D Grünwald. *The minimum description length principle*. 2007. URL https://direct.mit.edu/books/monograph/3813/The-Minimum-Description-Length-Principle.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Retrieval augmented language model pre-training. In *International Conference on Machine Learning*, pp. 3929–3938, 2020. URL http://proceedings.mlr.press/v119/guu20a.html.

John Hewitt. Initializing new word embeddings for pretrained language models, 2021. URL https://nlp.stanford.edu/~johnhew//vocab-expansion.html.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 6769–6781, 2020. URL https://aclanthology.org/2020.emnlp-main.550/.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HklBjCEKvH.

Yuri Kuratov, Mikhail Arkhipov, Aydar Bulatov, and Mikhail Burtsev. Cramming 1568 tokens into a single vector and back again: Exploring the limits of embedding space capacity. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19323–19339. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.acl-long.948/.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, 2021. URL https://aclanthology.org/2021.emnlp-main.243/.

Patrick Lewis, Ethan Perez, Aleksandar Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33: 9459–9474, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 4582–4597, 2021. URL https://aclanthology.org/2021.acl-long.353/.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. pp. 74–81. Association for Computational Linguistics, 2004. URL https://aclanthology.org/W04-1013/.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, pp. 157–173, 2024a. URL https://aclanthology.org/2024.tacl-1.9/.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9), 2023. URL https://doi.org/10.1145/3560815.

Yujia Liu, Jiacheng Zhang, Ziming Wang, Xiaohan Li, Jing Li, and Hao Wang. APIGen: Automated API code generation for function-calling capabilities in large language models, 2024b. URL https://arxiv.org/abs/2406.18518.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, pp. 529–533, 2015. URL https://www.nature.com/articles/nature14236.

Ali Modarressi, Ayyoob Imani, Mohsen Fayyaz, and Hinrich Schütze. RET-LLM: Towards a general read-write memory for large language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 15558–15571, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6a4cd50db0cad92c4c8d9e6ee01ac8c6-Abstract-Conference.html.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=2DtxPCL3T5.

Charles Packer, Vivian Fang, Shishir Gururaj Patil, Kevin Lin, Sarah Wooders, and Joseph E Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023. URL https://arxiv.org/abs/2310.08560.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=2GmDdhBdDk.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapter-Fusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 487–503. Association for Computational Linguistics, 2021. URL https://aclanthology.org/2021.eacl-main.39/.

Qwen and et al. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2019. URL https://arxiv.org/abs/1908.10084.

Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. Routing networks: Adaptive selection of non-linear functions for multi-task learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ry8dvM-R-.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A systematic survey of prompt engineering in large language models: Techniques and applications, 2025. URL https://arxiv.org/abs/2402.07927.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021. URL https://arxiv.org/abs/2110.08207.

Ignacio Sastre and Aiala Rosá. Memory tokens: Large language models can generate reversible sentence embeddings. In *Proceedings of the First Workshop on Large Language Model Memorization*, pp. 183–189. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.l2m2-1.14/.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=Yacmpz84TH.

Larry R Squire. Memory and brain systems: 1969–2009. *Journal of Neuroscience*, 29:12711–12716, 2009. URL https://www.jneurosci.org/content/29/41/12711.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008, 2017. URL https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Renxi Wang, Xudong Han, Lei Ji, Shu Wang, Timothy Baldwin, and Haonan Li. Toolgen: Unified tool retrieval and calling via generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=XLMAMmowdY.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, et al. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 5085–5109, 2022a. URL https://aclanthology.org/2022.emnlp-main.340/.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 139–149, 2022b. URL https://openaccess.thecvf.com/content/CVPR2022/papers/Wang_Learning_To_Prompt_for_Continual_Learning_CVPR_2022_paper.pdf.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=gEZrGCozdqR.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022b. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Zijun Wu, Yongchang Hao, and Lili Mou. Ulpt: Prompt tuning with ultra-low-dimensional optimization, 2025. URL https://arxiv.org/abs/2502.04501.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025. URL https://arxiv.org/pdf/2502.12110.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Yu Zhong, Longyue Wang, Jiajun Liu, Guangdong Chen, Minjun Wu, Qifan Zhou, Zerui Wang, Xianzhi Wang, et al. MemoryLLM: Towards self-updatable large language models. *arXiv preprint arXiv:2402.04624*, 2024. URL https://arxiv.org/abs/2402.04624.

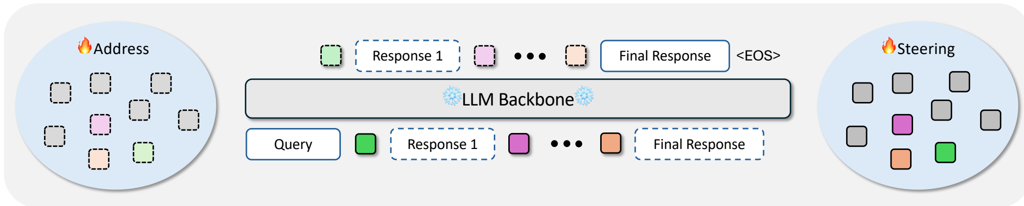

Figure 5: Overview of Decoupled TokMem embeddings, which learns separate memory matrices for address of memories and generation steering.

A Decoupled Embedding for TokMem

In the standard TokMem formulation, each memory token embedding $\boldsymbol{m}_i \in \mathbb{R}^d$ is shared shared across two roles: (1) addressing for memory routing and (2) steering for generation. We consider a decoupled (DC) variant that separates these functions into two embedding matrices:

$$M^{\text{addr}} = (\boldsymbol{u}_1; \ldots; \boldsymbol{u}_l) \in \mathbb{R}^{l \times d}, \qquad M^{\text{steer}} = (\boldsymbol{s}_1; \ldots; \boldsymbol{s}_l) \in \mathbb{R}^{l \times d}. \tag{7}$$

Here, M^{addr} provides *address embeddings* at the output layer. When a memory token is predicted, the model produces a distribution over indices i according to M^{addr}. The chosen index i is then used to retrieve the corresponding steering embedding $\boldsymbol{s}_i$ from M^{steer}, which is injected into the input sequence and influences subsequent generation.

Training follows the standard next-token prediction objective, analogous to Equation 3:

$$\mathcal{L}(\boldsymbol{a}; M^{\text{addr}}, M^{\text{steer}}) = -\sum_{i>k} \log \Pr(a_i \mid \boldsymbol{a}_{<i}; M^{\text{addr}}, M^{\text{steer}}). \tag{8}$$

where k denotes the query length. During optimization, only M^{addr} and M^{steer} are updated; the backbone remains frozen. In addition, the renormalization treatment introduced in Section 2.2 is only applied to the address embeddings in M^{addr}.

This decoupled formulation provides a clean separation of functionality: routing is handled via M^{addr}, while steering is controlled by M^{steer}. While conceptually clean, our experiments do not show consistent improvements over the coupled formulation, particularly on larger models, where embedding capacity is sufficient to jointly support both roles.

B Details for Compositional Memory Recall

B.1 Details of Adaptation Phase

In compositional scenarios, the model should not only recall individual procedures but also compose them to solve multi-step queries. To prepare TokMem for such use, we construct a held-out auxiliary training set of 50 tools (5,000 samples) following Section 3.3. The backbone is then fine-tuned for one epoch on this set using LoRA, jointly with the temporary memory embeddings, before the adapted weights are merged and frozen.

The intuition for this adaptation phase to let the LLM learn to align its routing and generation behavior with compositional memory recall. After adaptation, the temporary embeddings are discarded, while the adapted backbone is retained for inference with new tasks. This procedure provides a general inductive bias for modular composition, enabling it to generalize to new tools and procedures without further retraining.

Algorithm 1 summarizes this lightweight procedure. Temporary memory embeddings are inserted into the input sequence, the loss is optimized jointly over memory and response tokens, and once the backbone has adapted, the temporary memory bank is discarded.

Algorithm 1 Adaptation Phase for Compositional Memory Recall

Require: Pretrained backbone f_{θ_0}, adaptation traces $\mathcal{D}_{\text{adapt}}$ from held-out procedures
1: Initialize backbone $\theta \leftarrow \theta_0$ and temporary memory embeddings $\mathcal{M}$
2: Set learning rates η_θ and $\eta_\mathcal{M}$
3: **for** each minibatch in $\mathcal{D}_{\text{adapt}}$ **do**
4: Insert $\mathcal{M}$ into sequence; forward pass with f_θ
5: Compute loss $\mathcal{L}$ on memory and response tokens
6: $\theta \leftarrow \theta - \eta_\theta \nabla_\theta \mathcal{L}$
7: $\mathcal{M} \leftarrow \mathcal{M} - \eta_\mathcal{M} \nabla_\mathcal{M} \mathcal{L}$
8: **end for**
9: Discard temporary memory $\mathcal{M}$ and freeze backbone θ
10: **return** adapted backbone f_θ

Table 5: Comparison of standard fine-tuning vs. fine-tuning with an adaptation phase.

Model	Configuration	Argument F1			
		2 calls	3 calls	4 calls	**Avg.**
Llama 1B	*Fine-tuning*	77.3	72.6	55.8	68.6
	+ *adapt*	72.8	54.6	42.1	56.5
	+ *adapt & all linear*	74.1	66.7	68.4	**69.7**
Llama 3B	*Fine-tuning*	87.9	86.6	82.9	**85.8**
	+ *adapt*	78.9	69.5	63.2	70.5
	+ *adapt & all linear*	79.7	72.3	89.5	80.5
Llama 8B	*Fine-tuning*	87.7	86.8	88.2	**87.6**
	+ *adapt*	77.6	66.3	84.2	76.0
	+ *adapt & all linear*	84.5	78.3	89.5	84.1

B.2 Analysis of Fine-tuning with Adaptation Phase

TokMem employs an adaptation phase where it is fine-tuned on a held-out auxiliary tools before training for new tools. In Section 3.3, our fine-tuning baseline dose not include this phase. To validate the fairness of our fine-tuning baseline, we investigate whether it could also benefit from this phase by training sequentially on the held-out auxiliary tools and the target tools.

As shown in Table 5, introducing the adaptation phase generally degrades performance compared with the standard fine-tuning baseline. This might be because of the interference between the disjoint held-out auxiliary and target tool sets with completely different functions and parameters. By contrast, TokMem avoids this interference issue by "abandoning" the auxiliary memory tokens after the adaptation phase.

Although adding the training capacity by finetuning with all linear layers alleviate this issue, we see that it still cannot outperform our original setup. This confirms that fine-tuning without adaptation is a stronger baseline for comparison.

B.3 Analysis of Unfreezing LLM Backbone for TokMem

We further examine the importance of freezing the backbone when adding new tool memories, reflecting real-world usage where procedural knowledge grows incrementally over time. This setting contrasts with recent approaches (Wang et al., 2025) that compress tool usage into virtual tokens by post-training the backbone. While such methods improve retrieval efficiency at scale, they rely on modifying backbone parameters, which hinders continual adaptability and risks overwriting prior knowledge.

As shown in Figure 6, unfreezing the backbone during TokMem adaptation leads to severe forgetting of previously learned tools, consistent with catastrophic interference in continual learning. By contrast, freezing the backbone preserves prior capabilities while allowing new tool memories to be incorporated without loss, highlighting TokMem's advantage for incremental adaptation. Notably,

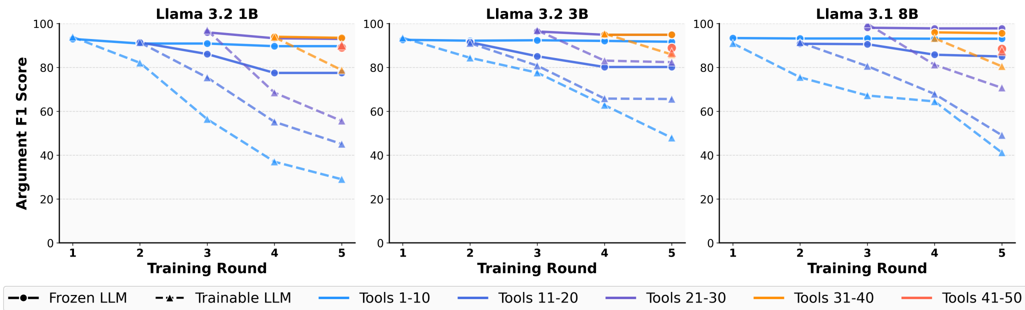

Figure 6: Comparison between Freezing and unfreezing the backbone. Allowing the backbone to update when adding new tool memories causes severe forgetting. Freezing preserves prior tools while enabling new ones.

Table 6: TokMem generalizes to longer tool-call chains at test time, significantly outperforming fine-tuning in zero-shot multi-step settings (training with 1 call and test with 2-4 calls).

		Test Time			
Train Maximum Calls	**Method**	2 calls	3 calls	4 calls	**Avg.**
1-call	*Fine-tuning*	34.9	21.3	14.1	23.4
	TokMem	60.3	54.3	48.9	**54.5** **(+31.1)**
2-call	*Fine-tuning*	86.2	78.8	64.8	76.6
	TokMem	82.0	81.8	82.3	**82.0** (+5.4)
3-call	*Fine-tuning*	86.9	85.5	80.3	84.2
	TokMem	86.8	84.0	84.7	**85.2** (+1.0)
4-call	*Fine-tuning*	87.9	86.6	82.9	85.8
	TokMem	85.9	86.7	88.3	**86.3** (+0.5)

unfreezing offers no meaningful performance gains after the initial training round, suggesting that TokMem strikes an effective balance between performance and continual adaptation.

B.4 Compositional Generalization

We observe that TokMem provides clear advantages in compositional generalization over fine-tuning. Table 6 reports Argument F1 when the Llama 3B model is evaluated on queries requiring more function calls than those observed during training.

Notably, when trained solely on single-call data, TokMem achieves much stronger performance than fine-tuning when evaluating on 2 to 4 calls test data. This demonstrates that memory tokens trained for atomic procedures can be effectively composed at test time, enabling strong zero-shot generalization to multi-step behavior.

As the training regime is expanded to include more calls (e.g., up to 3 or 4), the performance gap narrows, but TokMem remains competitive or slightly ahead across all configurations. These results suggest that TokMem naturally supports compositionality, enabling flexible chaining of learned procedures without requiring task-specific fine-tuning.

B.5 Analysis of Norm Inflation for Newer Memories

We analyze the L2 norm of the learned memory embeddings when tools are introduced sequentially using Llama 3.2 3B. We follow the setup in Figure 4 by training TokMem with 5 rounds, adding 10 tools per round without our renormalization treatment. Figure 7 shows that newly added memory tokens gradually develop their L2 norms, which leads to competition with existing frozen tokens for the softmax operation for memory routing.

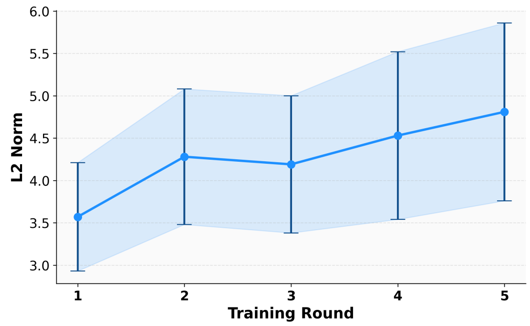

Figure 7: L2 norm of newly added memory tokens. Each round introduce 10 new tool memories. Error bar indicate standard deviation across the 10 tokens added in each round.

Table 7: Comparison of prefix tuning vs. TokMem condition embedding on GSM8K with two different size of Llama models. TokMem achieves higher compliance with required output formats and stronger exact-match accuracy than prefix tuning, especially in low-data regimes.

		Llama 3.2 1B		**Llama 3.2 3B**	
Data%	**Method**	Compliance↑	EM ↑	Compliance↑	EM ↑
20%	*Prefix tuning*	0.0	0.0	45.9	33.1
	TokMem	**98.0**	**37.7**	**94.6**	**65.6**
100%	*Prefix tuning*	82.8	30.0	97.2	64.1
	TokMem	**97.4**	**39.1**	**98.2**	**66.9**

C Additional Analysis on Memory Placement

We have stress-tested the effect of memory token placement (prefix vs. infix) with randomly generated queries and with varying length of memory tokens in Section 3.4. We now turn to the GSM8K math reasoning dataset (Cobbe et al., 2021), to evaluate generalization and training efficiency.

Our experiments run on Llama 3.2 1B and 3B as backbone models and compare prefix tuning against TokMem under two training setups: using only 20% of the training set that represents a low-data regime, or the full dataset. We report two evaluation metrics.

- **Compliance** measures whether the model follows the required answer format, i.e., producing the final answer after the delimiter “####”. This metric isolates the recall of procedural memory from the reasoning abilities already present in the backbone models.
- **Exact Match (EM)** measures the correctness of the final answer after standard normalization (e.g., removing commas or extraneous symbols).

As shown in Table 7, TokMem significantly outperforms prefix tuning, particularly in the low-data setting. With only 20% of the data, prefix-tuning fails to provide meaningful results, yielding zero compliance and EM on the 1B model and underperforming on the 3B model. By contrast, TokMem achieves near-perfect compliance and substantially higher EM scores across both backbones. When trained on the full dataset, prefix tuning improves considerably, yet TokMem continues to deliver stronger compliance and higher EM, underscoring its superior data efficiency and more reliable procedural control.

D Details of Datasets

D.1 Details of Super-Natural Instruction

We sample 1,000 English tasks from the SNI dataset, where each task is labeled with a task ID and a short descriptive name. The full list of sampled tasks is provided in Table 8. Tasks are introduced to the model sequentially in ascending order of their IDs (e.g., the model first sees task 1, then task 2, and so on).

After training on the first k tasks, we save a checkpoint and evaluate performance on the test sets of all k tasks encountered so far. This simulates a continual learning setup where the model is expected to acquire new procedures while retaining previously learned ones. Once the model has been trained on all $1{,}000$ tasks, it should be able to perform all of them without forgetting earlier tasks.

D.2 Details of Function Calling dataset

For evaluating compositional memory recall, we sample 50 tools from the APIGen dataset (Liu et al., 2024b). The list of tools and their corresponding descriptions is provided in Table 8.

For each tool, we collect 50 query–call pairs, some of which may involve multiple calls to the same tool. This yields a total of $50 \times 50 = 2{,}500$ samples representing the non-compositional use of tools. To avoid data leakage, we split these samples into training and test sets with a $9{:}1$ ratio.

On top of this, we synthesize complex queries by combining calls across different tools. These multi-step queries require the model to invoke multiple tools in sequence. We cap the number of synthesized samples at $5{,}000$ for training and 500 for testing.

E Future Work

Our experiments are conducted on the research-oriented SNI and APIGen datasets, which allow controlled analysis of atomic and compositional recall. While these settings demonstrate the feasibility and effectiveness of tokenized procedural memory without backbone training, they do not fully capture the diversity of real-world procedures.

In particular, richer forms of composition, such as interleaving function calls from APIGen with NLP tasks from SNI, as illustrated in Figure 1b, and multi-turn interactions remain unexplored, but could be supported with curated datasets. Overall, advancing TokMem toward practical deployment will require more realistic benchmarks or user-driven data collection pipelines that better reflect open-domain procedural knowledge.

Additional future directions include incorporating reinforcement learning to improve generalization for complex compositional structures, and enabling personalization by allowing users to attach their own memory banks while keeping the backbone frozen. Together, these extensions pave the way for scalable, compact, and user-adaptive memory systems in large language models.

F Use of Large Language Models

We used ChatGPT as a general-purpose assistant to improve the writing of our paper, including grammar, readability, and clarity. Additionally, we used it to search for related work, which we then manually verified. All ideas, analyses, and conclusions presented in this paper are our own, and we take full responsibility for the content.

Table 8: Details of the sampled tools from the APIGen dataset, including their names and descriptions.

ID	Tool	Description
1	auto_complete	Fetch auto-complete suggestions for a given query using the Wayfair API.
2	binary_addition	Adds two binary numbers and returns the result as a binary string.
3	binary_search	Performs binary search on a sorted list to find the index of a target value.
4	cagr	Calculates the Compound Annual Growth Rate (CAGR) of an investment.
5	calculate_factorial	Calculates the factorial of a non-negative integer.
6	calculate_grade	Calculates the weighted average grade based on scores and their corresponding weights.
7	calculate_median	Calculates the median of a list of numbers.
8	can_attend_all_meetings	Determines if a person can attend all meetings given a list of meeting time intervals.
9	cosine_similarity	Calculates the cosine similarity between two vectors.
10	count_bits	Counts the number of set bits (1's) in the binary representation of a number.
11	create_histogram	Create a histogram based on provided data.
12	directions_between_2_locations	Fetches the route information between two geographical locations including distance, duration, and steps.
13	fibonacci	Calculates the nth Fibonacci number.
14	final_velocity	Calculates the final velocity of an object given its initial velocity, acceleration, and time.
15	find_equilibrium_index	Finds the equilibrium index of a list, where the sum of elements on the left is equal to the sum of elements on the right.
16	find_first_non_repeating_char	Finds the first non-repeating character in a string.
17	find_longest_word	Finds the longest word in a list of words.
18	find_max_subarray_sum	Finds the maximum sum of a contiguous subarray within a list of integers.
19	find_minimum_rotated_sorted_array	Finds the minimum element in a rotated sorted array.
20	flatten_list	Flattens a nested list into a single-level list.
21	format_date	Converts a date string from one format to another.
22	generate_password	Generates a random password of specified length and character types.
23	generate_random_string	Generates a random string of specified length and character types.
24	get_city_from_zipcode	Retrieves the city name for a given ZIP code using the Ziptastic API.
25	get_pokemon_move_info	Retrieves information about a Pokémon's move using the PokéAPI.
26	get_product	Fetches product details from an API using the given product ID.
27	get_products_in_category	Fetches products in a specified category from the demo project's catalog.
28	greatest_common_divisor	Computes the greatest common divisor (GCD) of two non-negative integers.
29	integrate	Calculate the area under a curve for a specified function between two x values.
30	investment_profit	Calculates the profit from an investment based on the initial amount, annual return rate, and time.
31	is_anagram_phrase	Checks if two phrases are anagrams of each other, ignoring whitespace and punctuation.
32	is_leap_year	Checks if a year is a leap year.
33	is_palindrome	Checks if a string is a palindrome.
34	is_power	Checks if a number is a power of a given base.
35	is_rotation	Checks if one string is a rotation of another string.
36	is_valid_ip_address	Checks if a string is a valid IP address (IPv4).
37	is_valid_palindrome	Checks if a string is a valid palindrome, considering only alphanumeric characters and ignoring case.
38	is_valid_sudoku	Checks if a 9x9 Sudoku board is valid.
39	monthly_mortgage_payment	Calculates the monthly mortgage payment based on the loan amount, annual interest rate, and loan term.
40	note_duration	Calculates the duration between two musical notes based on their frequencies and the tempo.
41	place_safeway_order	Order specified items from a Safeway location.
42	polygon_area_shoelace	Calculates the area of a polygon using the shoelace formula.
43	potential_energy	Calculates the electrostatic potential energy given the charge and voltage.
44	project_population	Projects the population size after a specified number of years.
45	reverse_string	Reverses the characters in a string.
46	solve_quadratic	Computes the roots of a quadratic equation given its coefficients.
47	trapezoidal_integration	Calculates the definite integral of a function using the trapezoidal rule.
48	whois	Fetch the WhoIS lookup data for a given domain using the specified Toolbench RapidAPI key.
49	whole_foods_order	Places an order at Whole Foods.
50	wire_resistance	Calculates the resistance of a wire based on its length, cross-sectional area, and material resistivity.

Task Name
task001_quoref_question_generation
task004_mctaco_answer_generation_event_duration
task007_mctaco_answer_generation_transient_stationary
task010_mctaco_answer_generation_event_ordering
task013_mctaco_answer_generation_absolute_timepoint
task017_mctaco_wrong_answer_generation_frequency
task020_mctaco_span_based_question
task023_cosmosqa_question_generation
task026_drop_question_generation
task029_winogrande_full_object
task032_winogrande_question_generation_person
task035_winogrande_question_modification_person
task038_qasc_combined_fact
task041_qasc_answer_generation
task044_essential_terms_identifying_essential_words
task047_miscellaneous_answering_science_questions
task050_multirc_answerability
task054_multirc_write_correct_answer
task058_multirc_question_answering
task061_ropes_answer_generation
task065_timetravel_consistent_sentence_classification
task069_abductivenli_classification
task072_abductivenli_answer_generation
task076_splash_correcting_sql_mistake
task079_conala_concat_strings
task083_babi_t1_single_supporting_fact_answer_generation
task087_new_operator_addsub_arithmetic
task091_all_elements_from_index_i_to_j
task094_conala_calculate_mean
task098_conala_list_intersection
task101_reverse_and_concatenate_all_elements_from_index_i_to_j
task104_semeval_2019_task10_closed_vocabulary_mathematical_answer_generation
task1087_two_number_sum
task108_contextualabusedetection_classification
task111_asset_sentence_simplification
task113_count_frequency_of_letter
task1148_maximum_ascii_value
task1150_delete_max_min
task1153_bard_analogical_reasoning_affordance
task1156_bard_analogical_reasoning_tools
task1159_bard_analogical_reasoning_containers
task1162_coda19_title_classification
task1167_penn_treebank_coarse_pos_tagging
task1186_nne_hrngo_classification
task118_semeval_2019_task10_open_vocabulary_mathematical_answer_generation
task1192_food_flavor_profile
task1195_disflqa_disfluent_to_fluent_conversion
task1198_atomic_classification_owant
task1200_atomic_classification_xeffect
task1203_atomic_classification_xreact
task1207_atomic_classification_atlocation
task120_zest_text_modification
task1213_atomic_classification_desires
task1216_atomic_classification_causes
task122_conala_list_index_addition
task126_scan_structured_text_generation_command_action_all
task1284_hrngo_informativeness_classification
task1287_glue_qqp_paraphrasing
task128_scan_structured_text_generation_command_action_short
task1294_wiki_qa_answer_verification
task1297_qasc_question_answering
task130_scan_structured_text_generation_command_action_long
task1312_amazonreview_polarity_classification
task1315_find_range_array
task1319_country_by_barcode_prefix
task1321_country_continent
task1327_qa_zre_answer_generation_from_question
task1331_reverse_array
task1336_peixian_equity_evaluation_corpus_gender_classifier
task133_winowhy_reason_plausibility_detection
task1343_amazon_us_reviews_rating
task1346_glue_cola_grammatical_correctness_classification
task1354_sent_comp_classification
task1358_xlsum_title_generation
task1360_numer_sense_multiple_choice_qa_generation
task1366_healthfact_classification
task136_winowhy_knowledge_categorization
task1379_quarel_incorrect_answer_generation
task1382_quarel_write_correct_answer
task1385_anli_r1_entailment
task1388_cb_entailment
task1390_wscfixed_coreference
task1393_superglue_copa_text_completion
task1399_obqa_answer_generation
task1403_check_validity_date_mmddyyyy
task1406_kth_smallest_element
task1409_dart_text_generation
task1413_dart_object_identification
task1418_bless_semantic_relation_classification
task1420_mathqa_general
task1423_mathqa_geometry
task1426_country_independence_year
task1429_evalution_semantic_relation_classification
task1434_head_qa_classification
task1439_doqa_cooking_isanswerable
task1441_doqa_movies_answer_generation
task1444_round_power_of_two
task1447_drug_extraction_ade
task1451_drug_dose_extraction
task145_afs_argument_similarity_death_penalty
task1480_gene_extraction_jnlpba_dataset
task1483_chemical_extraction_chemprot_dataset
task1486_cell_extraction_anem_dataset
task1489_sarcasmdetection_tweet_classification
task1498_24hour_to_12hour_clock
task1500_dstc3_classification
task1503_hatexplain_classification
task1506_celebrity_minimal_dob_span
task1509_evalution_antonyms
task1515_imppres_longtextgeneration
task1518_limit_answer_generation
task1520_qa_srl_answer_generation
task1531_daily_dialog_type_classification
task1534_daily_dialog_question_classification
task153_tomqa_find_location_hard_clean
task1548_wiqa_binary_classification
task1551_every_ith_element_from_kth_element
task1555_scitail_answer_generation
task1558_jfleg_incorrect_answer_generation
task1560_blimp_binary_classification
task1566_propara_structured_text_generation
task1572_samsum_summary
task1575_amazon_reviews_multi_sentiment_classification
task1579_gigaword_incorrect_summarization
task1581_eqasc-perturbed_answer_generation
task1585_root09_hypernym_generation
task1589_scifact_classification
task1592_yahoo_answers_topics_classfication
task1595_event2mind_text_generation_1
task1598_nyc_long_text_generation
task1600_smcalflow_sentence_generation
task1603_smcalflow_sentence_generation
task1606_ethos_text_classification
task1609_xquad_en_question_generation
task1613_sick_given_category_generate_sentence
task1618_cc_alligned_classify_tel_eng
task1623_disfl_qa_disfluent_question_classification
task162_count_words_starting_with_letter
task1638_doqa2.1_movies_text_summarization
task1645_medical_question_pair_dataset_text_classification
task1657_gooaq_question_generation
task1660_super_glue_question_generation
task1665_trainglecopa_question_generation
task167_strategyqa_question_generation
task1703_ljspeech_textmodification
task1706_ljspeech_classification
task1713_convai3_sentence_generation
task1721_civil_comments_obscenity_classification
task1724_civil_comments_insult_classification
task1727_wiqa_what_is_the_effect
task1730_personachat_choose_next
task177_para-nmt_paraphrasing
task180_intervention_extraction
task183_rhyme_generation
task186_snli_contradiction_to_entailment_text_modification
task189_snli_neutral_to_contradiction_text_modification
task192_hotpotqa_sentence_generation
task195_sentiment140_classification
task198_mnli_domain_classification
task201_mnli_neutral_classification
task204_mnli_same_genre_classification
task207_max_element_lists
task210_logic2text_structured_text_generation
task213_rocstories_correct_ending_classification
task216_rocstories_correct_answer_generation
task219_rocstories_title_answer_generation
task222_rocstories_two_chioce_slotting_classification
task226_english_language_answer_relevance_classification
task229_arc_answer_generation_hard
task232_iirc_link_number_classification
task235_iirc_question_from_subtext_answer_generation
task238_iirc_answer_from_passage_answer_generation
task241_tweetqa_classification
task244_count_elements_in_set_union
task248_dream_classification
task265_paper_reviews_language_identification
task269_csrg_counterfactual_story_generation
task274_overruling_legal_classification
task277_stereoset_sentence_generation_stereotype
task280_stereoset_classification_stereotype_type
task283_dream_incorrect_answer_generation
task286_olid_offense_judgment
task289_gigaword_summarization
task293_storycommonsense_emotion_text_generation
task296_storycloze_correct_end_classification
task300_storycloze_order_generation
task303_record_incorrect_answer_generation
task306_jeopardy_answer_generation_double
task309_race_answer_generation
task314_europarl_sv-en_classification
task317_crows-pairs_classification_stereotype_type
task320_stereoset_classification_race
task323_jigsaw_classification_sexually_explicit
task326_jigsaw_classification_obscene
task329_gap_classification
task332_tellmewhy_answer_generation
task337_hateeval_classification_individual_en
task341_winomt_classification_gender_anti
task344_hybridqa_answer_generation
task347_hybridqa_incorrect_answer_generation
task350_winomt_classification_gender_identifiability_pro
task353_casino_classification_negotiation_elicit_pref
task357_casino_classification_negotiation_small_talk
task360_spolin_yesand_response_generation
task363_sst2_polarity_classification
task366_synthetic_return_primes
task370_synthetic_remove_divisible_by_3
task373_synthetic_round_tens_place
task376_reverse_order_of_words
task379_agnews_topic_classification
task382_hybridqa_answer_generation
task385_socialiqa_incorrect_answer_generation
task388_torque_token_classification
task392_inverse_causal_relationship
task399_semeval_2018_task1_tweet_sadness_detection
task402_grailqa_paraphrase_generation
task418_persent_title_generation
task421_persent_sentence_sentiment_classification
task427_hindienglish_corpora_hi-en_language_identification
task430_senteval_subject_count
task437_alt_en_ja_answer_generation
task442_com_qa_paraphrase_question_generation
task453_swag_answer_generation
task456_matres_intention_classification
task459_matres_static_classification
task462_qasper_classification
task471_haspart_answer_generation
task477_cls_english_dvd_classification
task489_mwsc_question_generation
task493_review_polarity_classification
task496_semeval_answer_generation
task499_extract_and_add_all_numbers_from_list
task502_scruples_anecdotes_whoiswrong_verification
task505_count_all_numerical_elements_in_list
task508_scruples_dilemmas_more_ethical_isidentifiable
task511_reddit_tifu_long_text_summarization
task514_argument_consequence_classification
task517_emo_classify_emotion_of_dialogue
task520_aquamuse_answer_given_in_passage
task523_find_if_numbers_or_alphabets_are_more_in_list
task550_discofuse_sentence_generation
task564_discofuse_classification
task567_circa_text_generation
task570_recipe_nlg_ner_generation
task573_air_dialogue_classification
task576_curiosity_dialogs_answer_generation
task579_socialiqa_classification
task582_naturalquestion_answer_generation
task585_preposition_classification
task588_amazonfood_rating_classification
task591_sciq_answer_generation
task594_sciq_question_generation
task597_cuad_answer_generation
task600_find_the_longest_common_substring_in_two_strings
task605_find_the_longest_common_subsequence_in_two_lists
task608_sbic_sexual_offense_binary_classification
task611_mutual_multi_turn_dialogue
task615_moviesqa_answer_generation
task618_amazonreview_summary_text_generation
task621_ohsumed_yes_no_numerical_answer_generation
task624_ohsumed_question_answering
task627_xlwic_word_with_same_meaning_sentence_generation
task630_dbpedia_14_classification
task633_dbpedia_14_answer_generation
task636_extract_and_sort_unique_alphabets_in_a_list
task639_multi_woz_user_utterance_generation
task643_refresd_classification
task648_answer_generation
task667_mmmlu_answer_generation_business_ethics
task670_ambigqa_question_generation
task672_nummersense
task676_ollie_relationship_answer_generation
task679_hope_edi_english_text_classification
task683_online_privacy_policy_text_purpose_answer_generation
task686_mmmlu_answer_generation_college_biology
task690_mmmlu_answer_generation_college_medicine
task693_mmmlu_answer_generation_conceptual_physics
task696_mmmlu_answer_generation_elementary_mathematics
task699_mmmlu_answer_generation_high_school_biology
task702_mmmlu_answer_generation_high_school_european_history
task705_mmmlu_answer_generation_high_school_macroeconomics
task708_mmmlu_answer_generation_high_school_physics
task711_mmmlu_answer_generation_high_school_us_history
task714_mmmlu_answer_generation_human_sexuality
task717_mmmlu_answer_generation_logical_fallacies
task721_mmmlu_answer_generation_medical_genetics
task724_mmmlu_answer_generation_moral_scenarios
task727_mmmlu_answer_generation_prehistory
task730_mmmlu_answer_generation_professional_medicine
task733_mmmlu_answer_generation_security_studies
task736_mmmlu_answer_generation_virology
task739_lhoestq_question_generation
task743_eurlex_summarization
task746_yelp_restaurant_review_classification
task749_glucose_reverse_cause_emotion_detection
task752_svamp_multiplication_question_answering
task755_find_longest_substring_and_replace_its_sorted_lowercase_version_in_both_lists
task758_msr_sqa_question_answer_generation
task766_craigslist_bargains_classification
task769_qed_summarization
task820_protoqa_answer_generation
task827_copa_commonsense_reasoning
task832_poleval2019_mt_classification
task835_mathdataset_answer_generation
task845_pubmedqa_question_generation
task848_pubmedqa_classification
task851_synthetic_multiply_evens
task854_hippocorpus_classification
task857_inquisitive_question_generation
task860_prost_mcq_generation
task862_asdiv_multidiv_question_answering
task865_mawps_addsub_question_answering
task868_cfq_mcd1_explanation_to_sql
task870_msmarco_answer_generation
task879_schema_guided_dstc8_classification
task886_quail_question_generation
task889_goemotions_classification
task892_gap_reverse_coreference_resolution
task897_freebase_qa_topic_question_generation
task900_freebase_qa_category_classification
task903_deceptive_opinion_spam_classification
task906_dialogre_identify_names
task910_bianet_classification
task918_coqa_answer_generation
task922_event2mind_word_generation
task925_coached_conv_pref_classifier
task928_yelp_positive_to_negative_style_transfer
task932_dailydialog_classification
task935_defeasible_nli_atomic_classification
task955_wiki_auto_style_transfer
task958_e2e_nlg_text_generation_parse
task964_librispeech_asr_text_auto_completion
task970_sherliic_causal_relationship
task976_pib_indian_language_identification

Task Name
task002_quoref_answer_generation
task005_mctaco_wrong_answer_generation_event_duration
task008_mctaco_wrong_answer_generation_transient_stationary
task011_mctaco_wrong_answer_generation_event_ordering
task015_mctaco_question_generation_frequency
task018_mctaco_temporal_reasoning_presence
task021_mctaco_grammatical_logical
task024_cosmosqa_answer_generation
task027_drop_answer_type_generation
task030_winogrande_full_person
task033_winogrande_answer_generation
task036_qasc_topic_word_to_generate_related_fact
task039_qasc_find_overlapping_words
task042_qasc_incorrect_option_generation
task045_miscellaneous_sentence_paraphrasing
task048_multirc_question_generation
task051_multirc_correct_answer_single_sentence
task056_multirc_classify_correct_answer
task059_ropes_story_generation
task063_first_i_elements
task067_abductivenli_answer_generation
task070_abductivenli_incorrect_classification
task073_commonsenseqa_answer_generation
task077_splash_explanation_to_sql
task080_piqa_answer_generation
task084_babi_t1_single_supporting_fact_identify_relevant_fact
task088_identify_typo_verification
task092_check_prime_classification
task096_conala_list_index_subtraction
task099_reverse_elements_between_index_i_and_j
task102_commongen_sentence_generation
task106_scruples_ethical_judgment
task1088_array_of_products
task109_smsspamcollection_spamsmsdetection
task112_asset_simple_sentence_identification
task1146_country_capital
task1149_item_check_edible
task1151_swap_max_min
task1154_bard_analogical_reasoning_travel
task1157_bard_analogical_reasoning_rooms_for_containers
task115_help_advice_classification
task1163_coda19_section_classification
task1168_brown_coarse_pos_tagging
task1187_politifact_classification
task1190_add_integer_to_list
task1193_food_course_classification
task1196_atomic_classification_oeffect
task1199_atomic_classification_xattr
task1201_atomic_classification_xintent
task1204_atomic_classification_hinderedby
task1208_atomic_classification_xreason
task1211_atomic_classification_hassubevent
task1214_atomic_classification_xwant
task1217_atomic_answer_generation
task123_conala_sort_dictionary
task127_scan_long_text_generation_action_command_all
task1285_kpa_keypoint_matching
task1288_glue_mrpc_paraphrasing
task1291_multi_news_summarization
task1295_adversarial_qa_question_answering
task129_scan_long_text_generation_action_command_short
task1310_amazonreview_rating_classification
task1313_amazonreview_polarity_classification
task1316_remove_duplicates_string
task131_scan_long_text_generation_action_command_long
task1325_qa_zre_question_generation_on_subject_relation
task1328_qa_zre_relation_generation_from_question
task1332_check_leap_year
task1338_peixian_equity_evaluation_corpus_sentiment_classifier
task1340_msr_text_compression_compression
task1344_glue_entailment_classification
task1347_glue_sts-b_similarity_classification
task1355_sent_comp_summarization
task1359_numer_sense_answer_generation
task1361_movierationales_classification
task1368_healthfact_sentence_generation
task1370_newscomm_classification
task137_detoxifying-lms_classification_toxicity
task1383_quarel_write_incorrect_answer
task1386_anli_r2_entailment
task1389_hellaswag_completion
task1391_winogrande_easy_answer_generation
task1394_meta_woz_task_classification
task139_detoxifying-lms_classification_topicality
task1404_date_conversion
task1407_dart_question_generation
task140_detoxifying-lms_classification_style
task1415_youtube_caption_corrections_grammar_correction
task1419_mathqa_gain
task1421_mathqa_other
task1424_mathqa_probability
task1427_country_region_in_world
task142_odd-man-out_classification_no_category
task1437_doqa_cooking_question_generation
task143_odd-man-out_classification_generate_category
task1442_doqa_movies_isanswerable
task1445_closest_integers
task1448_disease_entity_extraction_ncbi_dataset
task1452_location_entity_extraction_btc_corpus
task146_afs_argument_similarity_gun_control
task1481_gene_extraction_bc2gm_dataset
task1484_gene_extraction_linnaeus_dataset
task1487_organism_substance_extraction_anem_dataset
task148_afs_argument_quality_gay_marriage
task1499_dstc3_summarization
task1501_dstc3_answer_generation
task1504_hatexplain_answer_generation
task1507_boolean_temporal_reasoning
task150_afs_argument_quality_gun_control
task1516_imppres_naturallanguageinference
task1519_qa_srl_question_generation
task1529_scitail1.1_classification
task1532_daily_dialog_emotion_classification
task1535_daily_dialog_uniqueness_classification
task1540_parsed_pdfs_summarization
task1549_wiqa_answer_generation_missing_step
task1552_scitail_question_generation
task1556_scitail_passage_generation
task1559_blimp_binary_classification
task1562_zest_text_modification
task1567_propara_question_generation
task1573_samsum_classification
task1576_amazon_reviews_multi_english_language_classification
task157_count_vowels_and_consonants
task1582_bless_hypernym_generation
task1586_scifact_title_generation
task158_count_frequency_of_words
task1593_yahoo_answers_topics_classification
task1596_event2mind_text_generation_2
task1599_smcalflow_classification
task1601_webquestions_answer_generation
task1604_ethos_text_classification
task1607_ethos_text_classification
task160_replace_letter_in_a_sentence
task1614_sick_text_modify
task161_count_words_containing_letter
task1624_disfl_qa_question_yesno_classification
task1630_openpi_classification
task1639_doqa2.1_travel_text_summarization
task164_mcscript_question_answering_text
task1659_title_generation
task1661_super_glue_classification
task166_clariq_sentence_generation
task168_strategyqa_question_decomposition
task1704_ljspeech_textmodification
task170_hotpotqa_answer_generation
task1714_convai3_sentence_generation
task1722_civil_comments_threat_classification
task1725_civil_comments_severtoxicity_classification
task1728_web_nlg_data_to_text
task1731_quartz_question_answering
task178_quartz_question_answering
task181_outcome_extraction
task184_break_generate_question
task187_snli_entailment_to_contradiction_text_modification
task190_snli_classification
task193_duorc_question_generation
task196_sentiment140_answer_generation
task199_mnli_classification
task202_mnli_contradiction_classification
task205_remove_even_elements
task208_combinations_of_list
task211_logic2text_classification
task214_rocstories_incorrect_ending_classification
task217_rocstories_ordering_answer_generation
task220_rocstories_title_classification
task223_quartz_explanation_generation
task227_clariq_classification
task230_iirc_passage_classification
task233_iirc_link_exists_classification
task236_iirc_question_from_passage_answer_generation
task239_tweetqa_answer_generation
task242_tweetqa_classification
task245_check_presence_in_set_intersection
task249_enhanced_wsc_pronoun_disambiguation
task267_concatenate_and_reverse_all_elements_from_index_i_to_j
task270_csrg_counterfactual_context_generation
task275_enhanced_wsc_paraphrase_generation
task278_stereoset_sentence_generation_antistereotype
task281_points_of_correspondence
task284_imdb_classification
task287_casehold_legal_incorrect_answer_generation
task290_tellmewhy_question_answerability
task294_storycommonsense_motiv_text_generation
task297_storycloze_incorrect_end_classification
task301_record_question_generation
task304_numeric_fused_head_resolution
task307_jeopardy_answer_generation_final
task310_race_classification
task315_europarl_sv-en_language_identification
task318_stereoset_classification_gender
task321_stereoset_classification_religion
task324_jigsaw_classification_disagree
task327_jigsaw_classification_toxic
task330_gap_answer_generation
task333_hateeval_classification_hate_en
task339_record_answer_generation
task342_winomt_classification_profession_pro
task345_hybridqa_answer_generation
task348_squad2.0_unanswerable_question_generation
task351_winomt_classification_gender_identifiability_anti
task355_casino_classification_negotiation_other_need
task358_casino_classification_negotiation_uv_part
task361_spolin_yesand_prompt_response_classification
task364_regard_social_impact_classification
task368_synthetic_even_or_odd_calculation
task371_synthetic_product_of_list
task374_synthetic_pos_or_neg_calculation
task377_remove_words_of_given_length
task380_boolq_yes_no_question
task383_matres_classification
task386_semeval_2018_task3_irony_detection
task389_torque_generate_temporal_question
task397_semeval_2018_task1_tweet_anger_detection
task400_paws_paraphrase_classification
task404_grailqa_paraphrase_validation
task419_persent_answer_generation
task423_persent_document_sentiment_verification
task428_senteval_inversion
task431_senteval_object_count
task440_eng_guj_parallel_corpus_gu-en_classification
task443_com_qa_ans_question_generation
task454_swag_incorrect_answer_generation
task457_matres_conditional_classification
task460_qasper_answer_generation
task469_mrqa_answer_generation
task472_haspart_classification
task478_cls_english_music_classification
task491_mwsc_answer_generation
task494_review_polarity_answer_generation
task497_extract_all_numbers_from_list_in_order
task500_scruples_anecdotes_title_generation
task503_scruples_anecdotes_isanswerable
task506_position_of_all_alphabetical_elements_in_list
task509_collate_of_all_alphabetical_and_numerical_elements_in_list_separately
task512_twitter_emotion_classification
task515_senteval_odd_word_out
task518_emo_different_dialogue_emotions
task521_trivia_question_classification
task532_europarl_en-es_classification
task562_alt_language_identification
task565_circa_answer_generation
task568_circa_question_generation
task571_recipe_nlg_ner_generation
task574_air_dialogue_sentence_generation
task577_curiosity_dialogs_classification
task580_socialiqa_answer_generation
task583_udeps_eng_coarse_pos_tagging
task586_amazonfood_polarity_classification
task589_amazonfood_summary_text_generation
task592_sciq_incorrect_answer_generation
task595_mocha_answer_generation
task598_cuad_answer_generation
task602_wikitext-103_answer_generation
task606_sum_of_all_numbers_in_list_between_positions_i_and_j
task609_sbic_potentially_offense_binary_classification
task613_politifact_text_generation
task616_cola_classification
task619_ohsumed_abstract_title_generation
task622_replace_alphabets_in_a_list_by_their_position_in_english_alphabet
task625_xlwic_true_or_false_answer_generation
task628_xlwic_word_with_different_meaning_sentence_generation
task631_dbpedia_14_incorrect_answer_generation
task634_allegro_reviews_classification
task637_extract_and_sort_unique_digits_in_a_list
task641_esnli_classification
task645_summarization
task649_race_blank_question_generation
task668_extreme_abstract_summarization
task671_ambigqa_text_generation
task673_google_wellformed_query_classification
task677_ollie_sentence_answer_generation
task681_hope_edi_malayalam_text_classification
task684_online_privacy_policy_text_information_type_generation
task687_mmmlu_answer_generation_college_chemistry
task691_mmmlu_answer_generation_college_physics
task694_mmmlu_answer_generation_econometrics
task697_mmmlu_answer_generation_formal_logic
task700_mmmlu_answer_generation_high_school_chemistry
task703_mmmlu_answer_generation_high_school_geography
task706_mmmlu_answer_generation_high_school_mathematics
task709_mmmlu_answer_generation_high_school_psychology
task712_mmmlu_answer_generation_high_school_world_history
task715_mmmlu_answer_generation_international_law
task718_mmmlu_answer_generation_machine_learning
task722_mmmlu_answer_generation_random_topic
task725_mmmlu_answer_generation_nutrition
task728_mmmlu_answer_generation_professional_accounting
task731_mmmlu_answer_generation_professional_psychology
task734_mmmlu_answer_generation_sociology
task737_mmmlu_answer_generation_world_religions
task740_lhoestq_answer_generation_quantity
task744_eurlex_classification
task747_glucose_cause_emotion_detection
task750_aqua_multiple_choice_answering
task753_svamp_addition_question_answering
task756_find_longert_substring_and_return_all_unique_alphabets_in_it
task759_msr_sqa_incorrect_answer_generation
task767_craigslist_bargains_classification
task770_pawsx_english_text_modification
task821_protoqa_question_generation
task828_copa_commonsense_cause_effect
task833_poem_sentiment_classification
task843_financial_phrasebank_classification
task846_pubmedqa_classification
task849_pubmedqa_answer_generation
task852_synthetic_multiply_odds
task855_conv_ai_2_classification
task858_inquisitive_span_detection
task861_asdiv_addsub_question_answering
task863_asdiv_multiop_question_answering
task866_mawps_multidiv_question_answering
task868_mawps_singleop_question_answering
task871_msmarco_question_generation
task880_schema_guided_dstc8_classification
task887_quail_answer_generation
task890_gcwd_classification
task893_gap_fill_the_blank_coreference_resolution
task898_freebase_qa_answer_generation
task901_freebase_qa_category_question_generation
task904_hate_speech_offensive_classification
task907_dialogre_identify_relationships
task912_bianet_classification
task919_coqa_incorrect_answer_generation
task923_event2mind_classifier
task926_coached_conv_pref_word_generation
task929_products_reviews_classification
task933_wiki_auto_style_transfer
task936_defeasible_nli_snli_classification
task956_leetcode_420_strong_password_check
task959_e2e_nlg_text_generation_identify
task966_ruletaker_fact_checking_based_on_given_context
task974_prachathai67k_sentiment_classification

Task Name
task003_mctaco_question_generation_event_duration
task006_mctaco_question_generation_transient_stationary
task009_mctaco_question_generation_event_ordering
task012_mctaco_question_generation_absolute_timepoint
task016_mctaco_answer_generation_frequency
task019_mctaco_temporal_reasoning_category
task022_cosmosqa_passage_inappropriate_binary
task025_cosmosqa_incorrect_answer_generation
task028_drop_answer_generation
task031_winogrande_question_generation_object
task034_winogrande_question_modification_object
task037_qasc_generate_related_fact
task040_qasc_question_generation
task043_essential_terms_answering_incomplete_questions
task046_miscellaneous_question_typing
task049_multirc_questions_needed_to_answer
task053_multirc_correct_bad_question
task057_multirc_classify_incorrect_answer
task060_ropes_question_generation
task064_all_elements_except_first_i
task068_abductivenli_incorrect_answer_generation
task071_abductivenli_answer_generation
task075_squad1.1_answer_generation
task078_all_elements_except_last_i
task081_piqa_wrong_answer_generation
task085_unnatural_addsub_arithmetic
task089_swap_words_verification
task093_conala_normalize_lists
task097_conala_remove_duplicates
task100_concatenate_all_elements_from_index_i_to_j
task103_facts2story_long_text_generation
task107_splash_question_to_sql
task1089_check_monotonic_array
task110_logic2text_sentence_generation
task1135_xcsr_en_commonsense_mc_classification
task1147_country_currency
task114_is_the_given_word_longest
task1152_bard_analogical_reasoning_causation
task1155_bard_analogical_reasoning_trash_or_treasure
task1158_bard_analogical_reasoning_manipulating_items
task1161_coda19_title_generation
task1164_coda19_section_correction_classification
task116_com2sense_commonsense_reasoning
task1188_count_max_freq_char
task1191_food_veg_nonveg
task1194_kth_largest_element
task1197_atomic_classification_oreact
task119_semeval_2019_task10_geometric_mathematical_answer_generation
task1202_atomic_classification_xneed
task1205_atomic_classification_isafter
task1209_atomic_classification_objectuse
task1212_atomic_classification_hasproperty
task1215_atomic_classification_capableof
task121_zest_text_modification
task124_conala_pair_averages
task1283_hrngo_quality_classification
task1286_openbookqa_question_answering
task1289_trec_classification
task1293_kilt_tasks_hotpotqa_question_answering
task1296_wiki_hop_question_answering
task1309_amazonreview_summary_classification
task1311_amazonreview_rating_classification
task1314_country_abbreviation
task1317_country_calling_code
task1320_country_domain_tld
task1326_qa_zre_question_generation_from_answer
task132_dais_text_modification
task1333_check_validity_date_ddmmyyyy
task1339_peixian_equity_evaluation_corpus_text_completion
task1342_amazon_us_reviews_title
task1345_glue_qqp_question_paraprashing
task134_winowhy_reason_generation
task1357_xlsum_summary_generation
task135_winowhy_wrong_reason_generation
task1364_hans_answer_generation
task1369_healthfact_sentence_generation
task1378_quarel_correct_answer_generation
task1380_quarel_correct_option_generation
task1384_deal_or_no_dialog_classification
task1387_anli_r3_entailment
task138_detoxifying-lms_classification_fluency
task1392_superglue_multirc_answer_verification
task1398_obqa_question_generation
task1401_obqa_sentence_generation
task1405_find_median
task1408_dart_similarity_classification
task1410_dart_relationship_extraction
task1416_youtube_caption_corrections_incorrect_grammar_classification
task141_odd-man-out_classification_category
task1422_mathqa_physics
task1425_country_iso_numeric
task1428_country_surface_area
task1431_head_qa_answer_generation
task1438_doqa_cooking_answer_generation
task1440_doqa_movies_question_generation
task1443_string_to_number
task1446_farthest_integers
task144_subjqa_question_answering
task1453_person_entity_extraction_btc_corpus
task1479_organization_entity_extraction_btc_corpus
task1482_gene_extraction_chemprot_dataset
task1485_organ_extraction_anem_dataset
task1488_sarcasmdetection_headline_classification
task1495_adverse_drug_event_classification
task149_afs_argument_quality_death_penalty
task1502_hatexplain_classification
task1505_root09_semantic_relation_classification
task1508_wordnet_antonyms
task1510_evalution_relation_extraction
task1517_limit_classfication
task151_tomqa_find_location_easy_clean
task152_tomqa_find_location_easy_noise
task1533_daily_dialog_formal_classification
task1536_daily_dialog_happiness_classification
task1541_agnews_classification
task154_tomqa_find_location_hard_noise
task1554_scitail_classification
task1557_jfleg_answer_generation
task155_count_nouns_verbs
task1565_triviaqa_classification
task156_codah_classification_adversarial
task1574_amazon_reviews_multi_language_identification
task1577_amazon_reviews_multi_japanese_language_classification
task1580_eqasc-perturbed_question_generation
task1583_bless_meronym_classification
task1587_scifact_classification
task1590_diplomacy_text_generation
task1594_yahoo_answers_topics_question_generation
task1597_nyc_slot_filling
task159_check_frequency_of_words_in_sentence_pair
task1602_webquestion_question_genreation
task1605_ethos_text_classification
task1608_xquad_en_answer_generation
task1612_sick_label_classification
task1615_sick_tclassify_b_relation_a
task1622_disfl_qa_text_modication
task1625_disfl_qa_asnwer_generation
task1637_doqa2.1_cooking_text_summarization
task1640_aqa1.0_answerable_unanswerable_question_classification
task1656_gooaq_answer_generation
task165_mcscript_question_answering_commonsense
task1664_winobias_text_generation
task1678_mathqa_answer_selection
task169_strategyqa_sentence_generation
task1705_ljspeech_classification
task1712_poki_classification
task1720_civil_comments_toxicity_classification
task1723_civil_comments_sexuallyexplicit_classification
task1726_mathqa_correct_answer_generation
task1729_personachat_generate_next
task176_break_decompose_questions
task179_participant_extraction
task182_duorc_question_generation
task185_snli_contradiction_to_neutral_text_modification
task188_snli_neutral_to_entailment_text_modification
task191_hotpotqa_question_generation
task194_duorc_answer_generation
task197_mnli_domain_answer_generation
task200_mnli_entailment_classification
task203_mnli_sentence_generation
task206_collatz_conjecture
task209_stancedetection_classification
task212_logic2text_classification
task215_rocstories_incorrect_answer_generation
task218_rocstories_swap_order_answer_generation
task221_rocstories_two_choice_classification
task225_english_language_answer_generation
task228_arc_answer_generation_easy
task231_iirc_link_classification
task234_iirc_passage_line_answer_generation
task237_iirc_answer_from_subtext_answer_generation
task240_tweetqa_question_generation
task243_count_elements_in_set_intersection
task247_dream_answer_generation
task264_paper_reviews_accept_or_reject_classification
task268_casehold_legal_answer_generation
task273_europarl_classification
task276_enhanced_wsc_classification
task279_stereoset_classification_stereotype
task282_scruples_event_time
task285_imdb_answer_generation
task288_gigaword_summarization
task292_storycommonsense_character_text_generation
task295_semeval_2020_task4_commonsense_reasoning
task298_storycloze_correct_end_classification
task302_record_classification
task305_jeopardy_answer_generation_normal
task308_jeopardy_answer_generation_all
task311_race_question_generation
task316_crows-pairs_classification_stereotype
task319_stereoset_classification_profession
task322_jigsaw_classification_threat
task325_jigsaw_classification_identity_attack
task328_jigsaw_classification_insult
task331_gap_incorrect_answer_generation
task335_hateeval_classification_aggresive_en
task340_winomt_classification_gender_pro
task343_winomt_classification_profession_anti
task346_hybridqa_classification
task349_squad2.0_answerable_unanswerable_question_classification
task352_coda-19_classification
task356_casino_classification_negotiation_self_need
task359_casino_classification_negotiation_vouch_fair
task362_spolin_yesand_prompt_response_sub_classification
task365_synthetic_remove_vowels
task369_synthetic_remove_odds
task372_synthetic_palindrome_numbers
task375_classify_type_of_sentence_in_debate
task378_reverse_words_of_given_length
task381_boolq_question_generation
task384_socialiqa_question_classification
task387_semeval_2018_task3_irony_classification
task391_causal_relationship
task398_semeval_2018_task1_tweet_joy_detection
task401_numeric_fused_head_reference
task405_narrativeqa_question_generation
task420_persent_document_sentiment_classification
task426_hindienglish_corpora_hi-en_classification
task429_senteval_tense
task434_alt_en_hi_answer_generation
task441_eng_guj_parallel_corpus_gu-en_language_identification
task447_opus_paracrawl_classification
task455_swag_context_generation
task458_matres_negation_classification
task461_qasper_question_generation
task470_mrqa_question_generation
task475_yelp_polarity_classification
task488_extract_all_alphabetical_elements_from_list_in_order
task492_mwsc_incorrect_answer_generation
task495_semeval_headline_classification
task498_scruples_anecdotes_whoiswrong_classification
task501_scruples_anecdotes_post_type_verification
task504_count_all_alphabetical_elements_in_list
task507_position_of_all_numerical_elements_in_list
task510_reddit_tifu_title_summarization
task513_argument_stance_classification
task516_senteval_conjoints_inversion
task519_aquamuse_question_generation
task522_news_editorial_summary
task547_alt_translation_entk_en
task563_discofuse_answer_generation
task566_circa_classification
task569_recipe_nlg_text_generation
task572_recipe_nlg_text_generation
task575_air_dialogue_classification
task578_curiosity_dialogs_answer_generation
task581_socialiqa_question_generation
task584_udeps_eng_fine_pos_tagging
task587_amazonfood_polarity_correction_classification
task590_amazonfood_summary_correction_classification
task593_sciq_explanation_generation
task596_mocha_question_generation
task599_cuad_question_generation
task603_wikitext-103_fill_in_the_blank
task607_sbic_intentional_offense_binary_classification
task610_conllpp_ner
task614_glucose_cause_event_detection
task617_amazonreview_category_text_generation
task620_ohsumed_medical_subject_headings_answer_generation
task623_ohsumed_yes_no_answer_generation
task626_xlwic_sentence_based_on_given_word_sentence_generation
task629_dbpedia_14_classification
task632_dbpedia_14_classification
task635_allegro_reviews_answer_generation
task638_multi_woz_classification
task642_esnli_classification
task647_answer_generation
task664_mmmlu_answer_generation_abstract_algebra
task669_ambigqa_answer_generation
task672_amazon_and_yelp_summarization_dataset_summarization
task674_google_wellformed_query_sentence_generation
task678_ollie_actual_relationship_answer_generation
task682_online_privacy_policy_text_classification
task685_mmmlu_answer_generation_clinical_knowledge
task689_mmmlu_answer_generation_college_mathematics
task692_mmmlu_answer_generation_computer_security
task695_mmmlu_answer_generation_electrical_engineering
task698_mmmlu_answer_generation_global_facts
task701_mmmlu_answer_generation_high_school_computer_science
task704_mmmlu_answer_generation_high_school_government_and_politics
task707_mmmlu_answer_generation_high_school_microeconomics
task710_mmmlu_answer_generation_high_school_statistics
task713_mmmlu_answer_generation_human_aging
task716_mmmlu_answer_generation_jurisprudence
task719_mmmlu_answer_generation_management
task723_mmmlu_answer_generation_moral_disputes
task726_mmmlu_answer_generation_philosophy
task729_mmmlu_answer_generation_professional_law
task732_mmmlu_answer_generation_public_relations
task735_mmmlu_answer_generation_us_foreign_policy
task738_perspectrum_classification
task742_lhoestq_answer_generation_frequency
task745_ai2_arithmetic_questions_arithmetic
task748_glucose_reverse_cause_event_detection
task751_svamp_subtraction_question_answering
task754_svamp_common-division_question_answering
task757_msr_sqa_question_generation
task761_app_review_classification
task768_qed_text_span_selection
task819_pec_sentiment_classification
task823_peixian-rtgender_sentiment_analysis
task831_giga_fren_classification
task834_mathdataset_classification
task844_financial_phrasebank_classification
task847_pubmedqa_question_generation
task850_synthetic_longest_palindrome
task853_hippocorpus_long_text_generation
task856_conv_ai_2_classification
task859_prost_question_generation
task861_prost_mcq_answers_generation
task864_asdiv_singleop_question_answering
task867_mawps_multiop_question_answering
task869_cfq_mcd1_sql_to_explanation
task875_emotion_classification
task881_schema_guided_dstc8_classification
task888_reviews_classification
task891_gap_coreference_resolution
task896_miam_language_classification
task899_freebase_qa_topic_generation
task902_deceptive_opinion_spam_classification
task905_hate_speech_offensive_classification
task908_dialogre_identify_familial_relationships
task917_coqa_question_generation
task921_code_x_glue_information_retreival
task924_event2mind_word_generation
task927_yelp_negative_to_positive_style_transfer
task930_dailydialog_classification
task934_turk_simplification
task937_defeasible_nli_social_classification
task957_e2e_nlg_text_generation_generate
task963_librispeech_asr_next_word_prediction
task967_ruletaker_incorrect_fact_generation_based_on_given_paragraph
task975_prachathai67k_same_genre_classification

Figure 8: Overview of the 1,000 English tasks from the SNI dataset used in the atomic recall setting.