Bidirectional LMs are Better Knowledge Memorizers? A Benchmark for Real-world Knowledge Injection

Yuwei Zhang[1] **Wenhao Yu**[2] **Shangbin Feng**[3] **Yifan Zhu**[1] **Letian Peng**[1]
Jayanth Srinivasa[4] **Gaowen Liu**[4] **Jingbo Shang**[1]*
UC San Diego[1] University of Notre Dame[2] University of Washington[3] Cisco[4]
{yuz163,jshang}@ucsd.edu

Abstract

Despite significant advances in large language models (LLMs), their knowledge memorization capabilities remain underexplored, due to the lack of standardized and high-quality testing grounds. In this paper, we introduce a novel, real-world and large-scale knowledge injection benchmark that continuously evolves without human intervention. Specifically, we propose WIKIDYK, which leverages recently-added and expert-curated facts from Wikipedia's "Did You Know..." entries. Each entry is converted into multiple question–answer pairs spanning diverse task formats from easy cloze prompts to complex multi-hop questions. WIKIDYK currently contains 12,290 facts and 77,180 questions, and its design allows for seamless extension with future updates from Wikipedia editors. Through extensive experiments using continued pre-training, we reveal a surprising insight: despite their prevalence in modern LLMs, Causal Language Models (CLMs) demonstrate significantly weaker knowledge memorization capabilities compared to Bidirectional Language Models (BiLMs), exhibiting a 23% lower accuracy in terms of reliability. To compensate for the smaller scales of current BiLMs, we introduce a modular collaborative framework utilizing ensembles of BiLMs as external knowledge repositories to integrate with LLMs. Experiment shows that this framework further improves the reliability accuracy by up to 29.1%. Code: https://github.com/zhang-yu-wei/WikiDYK.

1 Introduction

Large language models (LLMs) acquire the vast majority of their factual knowledge during pre-training, by learning patterns from massive web-scale corpora (Chang et al., 2024; Chen et al., 2024; Li and Goyal, 2025). This process allows them to recall facts, reason over information, and generate coherent text without explicit supervision. As a result, LLMs are frequently treated as static knowledge bases – capable of answering factual queries based on what they have seen during training. However, this raises an important question: *Can LLMs truly memorize and internalize new knowledge after pretraining?*

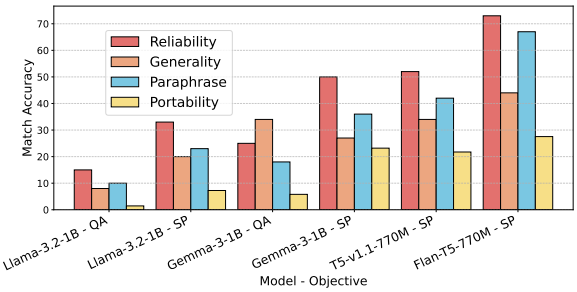

Figure 1: Evaluation results of injecting 1,000 facts from WIKIDYK. We report performance for two CLMs and two BiLMs. A more comprehensive set of results can be found in Appendix C.

Previous work suggests significant challenges in effectively updating the internal knowledge of language models (Jang et al., 2021, 2022; Kim et al., 2023; Ovadia et al., 2023; Fu et al., 2023; Zhang et al., 2023; Mecklenburg et al., 2024; Xu et al., 2025). However, these findings predominantly rely on synthetic datasets derived from noisy Wikipedia snapshots where the injected knowledge often lacks real-world significance and inherent complexities. Moreover, evaluations using synthetic questions often suffer from ill-defined contexts, such as unanswerable questions like "What is the value of y?", undermining their effectiveness in accurately reflecting model performance.

We extend previous efforts on knowledge injection by building a novel large-scale and high-quality benchmark – WIKIDYK – derived from a natural, expert-curated, and constantly updating knowledge source: Wikipedia's "Did You Know..." (DYK) pages[1]. These pages highlight Wikipedia's

* Corresponding authors.

[1] https://en.wikipedia.org/wiki/Wikipedia:Did_you_know

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 16964–16980

Benchmark	Dataset Source	Topical Scope	Update Frequency	Human Curated	Automatic Extension	Available Tasks
EvolvingQA (Kim et al., 2023)	Wikipedia	General	~80K monthly	✗	✓	QA
RealtimeQA (Kasai et al., 2023)	News	News	30 weekly	✓	✗	QA
StreamingQA (Liska et al., 2022)	News	News	9K quarterly	hybrid	✗	QA
TemporalWiki (Jang et al., 2022)	Wikipedia	General	300K monthly	✗	✓	Slot-filling
CKL (Jang et al., 2021)	Pre-train Data	General	–	hybrid	✗	Slot-filling
WIKIDYK (Ours)	Wikipedia	General	~10 daily	✓	✓	QA

Table 1: Comparison of WIKIDYK with existing benchmarks for knowledge injection.

continuous growth and domain diversity by featuring daily updates of facts reviewed by expert Wikipedia editors. Each day, about 10 facts are added to the list, selected from recently expanded articles which likely not exist in pre-training data while adhering to Wikipedia's most important content policies (*e.g.* "that teenage King Baldwin IV of Jerusalem fought one-handed and still went to battle after becoming blind and immobile at 22"). WIKIDYK leverages this structured, human-driven process to ensure both novelty and quality, offering a unique resource for evaluating knowledge injection in language models that go beyond synthetic dataset construction. We compare our WIKIDYK with previous benchmarks in Table 1.

With WIKIDYK we aim to systematically evaluate the performance of LLM knowledge injection. Building on prior work in knowledge editing (Meng et al., 2022a,b; Wang et al., 2023), we design a multi-dimensional evaluation suite using open-domain QA, spanning lower-level knowledge memorization to higher-level knowledge association tasks (Xu et al., 2025). To guarantee the ease of extension in the future, all evaluation questions are generated via a lightweight prompt-based method that employs only the factual knowledge and corresponding Wikipedia articles. An example of our evaluation workflow is visualized in Figure 2.

Based on these designs, we conduct a comprehensive comparative analysis of both model architectures and training objectives for knowledge injection. Given the observations on the "reverse curse" issue of Causal Language Models' (CLMs) (Berglund et al., 2023), we aim to investigate whether Bidirectional Language Models (BiLMs) that leverage the context from both sides can generalize better. Notably, our experiments reveal that the smaller BiLMs significantly outperforms the most recent large CLMs at memorizing knowledge as shown in Figure 1, even after aligning the training objective and the amount of training. We attribute this discrepancy to the reduced context visibility in CLMs during training, which hinders their ability to efficiently encode factual knowledge. In fact, bidirectional attention has also been found to facilitate model editing and finetuning (Ma et al., 2023; Kopiczko et al., 2024). Finally, to address the challenges of scaling knowledge injection while mitigating catastrophic forgetting, we propose a model collaboration framework that leverages BiLMs as dedicated knowledge repositories, that are then dynamically integrated with LLMs through a scope classifier. We showcase on WIKIDYK that it is possible to combine the vast pre-trained knowledge with newly injected of BiLMs effectively. Our framework can also be utilized on applications that require integrating knowledge from multiple domains and enables efficient updates via localized retraining, avoiding full-model retraining overhead.

In summary, our contribution is two-fold: a high-quality and expert-curated dataset WIKIDYK to evaluate LLM knowledge injection, and a call-to-action to revisit bidirectional LMs for neural knowledge modeling.

2 Related Works

Wikipedia's Did You Know (WikiDYK). WikiDYK is a section of Wikipedia that features newly added or expanded facts, updated daily to showcase novel and verified information. The XQA dataset (Liu et al., 2019a) uses the multilingual feature of WikiDYK sections to automatically generate multilingual question-answer pairs by masking named entities in factual statements. (Rybak et al., 2020) takes the Polish split of WikiDYK to benchmark Polish question answering systems. (Prakash et al., 2015) treat WikiDYK as trivia for entities inside Wikipedia to structure the relations between Wikipedia entities. Unlike previous work, we explicitly leverage the temporal structure of WikiDYK to analyze how well language models can incorporate and update newly emerging knowledge over time.

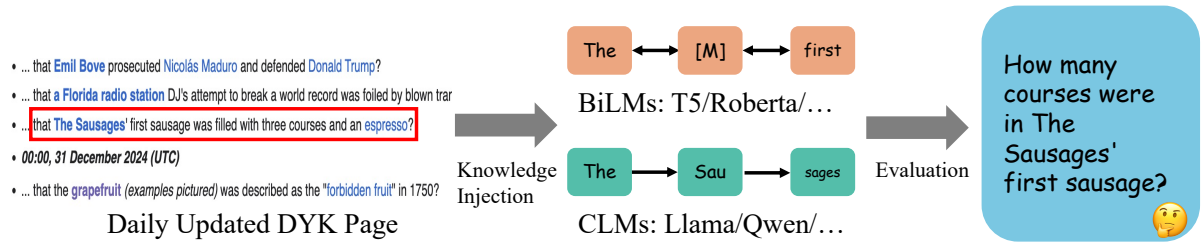

Figure 2: Proposed knowledge injection evaluation workflow. We inject the knowledge from via continued pretraining which can be achieved via various model architectures or training objectives. The injected model is then evaluated with questions from multiple dimensions from easy cloze prompts to complex multi-hop questions. Notice that the images are not used in the dataset.

Temporal Evolution of LLMs. Large language models (LLMs) have a fixed knowledge cutoff (Cheng et al., 2024), necessitating knowledge injection to stay current (Song et al., 2025). Supervised fine-tuning (SFT), especially with fact-based data (Mecklenburg et al., 2024), helps but suffers from retention issues (Ovadia et al., 2023). Surprisingly, even unaligned or random data can perform well (Fu et al., 2023), highlighting the need for better pruning. Plug-and-play methods inject external knowledge into frozen models via map-tuned embeddings (Zhang et al., 2023). Though retrieval-augmented generation (RAG) excels with novel facts (Ovadia et al., 2023), combining retrieval with efficient, aligned fine-tuning may yield more robust reasoning.

Generalization of Injected Knowledge. Once knowledge is injected, LLMs are expected not only to memorize the fine-tuned sequences but also to generalize the new information across diverse contexts. However, numerous studies in knowledge editing have underscored a central challenge: standard fine-tuning methods often struggle to simultaneously meet multiple critical objectives (Meng et al., 2022a; Onoe et al., 2023; Hoelscher-Obermaier et al., 2023; Gupta et al., 2023). To better understand how injected knowledge influences generalization, researchers have explored fine-tuning learning dynamics (Ren and Sutherland, 2025), applied influence functions (Grosse et al., 2023), and analyzed representation correlations (Peng et al., 2025). While a range of benchmarks has been developed to assess generalization in this setting (Kim et al., 2023; Kasai et al., 2023; Liska et al., 2022; Jang et al., 2022, 2021), many fall short in terms of quality and complexity compared to WikiDYK.

3 WIKIDYK Benchmark

In this section, we introduce the collection and analysis of data sets. Our dataset is constructed based on the daily updated Wikipedia's recent additions website [2] which contains numerous facts from newly reviewed articles by experts. In order to facilitate timely updates, we design a simple prompt-based QA generation workflow that can be used to evaluate knowledge injection models from various difficulty levels.

3.1 Data Collection

On the "DYK" webpage, about 10 facts are selected each day from recently created new articles or greatly expanded existing articles. The bold entity (*e.g.* "Gold Digger") within the fact is related to the original article that introduces new knowledge. We scrape both the raw text and the accompanying Wikipedia article with additional cleaning from the webpage. In order to ensure that the knowledge is up-to-date and align with LLM knowledge cutoffs, we only acquire those pages starting from January 2022 until April 2025 with a total number of 12, 290 facts.

3.2 Evaluation

Question Generation. We construct question-answer (QA) pairs using two sources: scraped factual knowledge snippets and Wikipedia articles. To comprehensively evaluate knowledge memorization and association capabilities (Xu et al., 2025), we design a multi-dimensional evaluation that includes five distinct question types as detailed in Table 2. **Reliability**: Directly tests recall of bold entities by formulating questions from the non-bold context of the original fact (*e.g.*, "What song..." for the bold Gold Digger). **Generality**: Extracts answers from implicit non-bold components of the

[2] https://en.wikipedia.org/wiki/Wikipedia:Recent_additions

Type	Example
Original	Kanye West originally wrote the chorus of "**Gold Digger**" from a female point of view.
Reliability	What is the name of the song for which Kanye West originally wrote the chorus from a female point of view? Answer: Gold Digger
Generality	From whose point of view did Kanye West originally write the chorus of "Gold Digger"? Answer: female
Paraphrase	What is the title of the song where Kanye West initially penned the chorus from a woman's perspective? Answer: Gold Digger
Portability	I recently came across a story about an American artist, born in 1977, who reshaped hip-hop with his ever-evolving sound and innovative style. I heard he once wrote a chorus meant to be sung from a female perspective for one of his tracks. Do you know which song that was? Answer: Gold Digger
Locality	Which American artist, born in 1977, revolutionized hip-hop with innovative music and influential fashion ventures, and is known for both his Grammy-winning albums and controversial public persona? Answer: Kanye West

Table 2: Example questions drawn from our multi-dimensional evaluations. Please refer to Section 3.2 for an in-depth explanation of each dimension and task.

same fact (*e.g.*, inferring female from "female point of view") which tests whether the injected model can accurately recall knowledge. **Paraphrase**: Uses syntactically rephrased or lexically substituted versions of the original fact (e.g., from "initially penned" to "originally wrote"). **Locality**: Evaluates retention of pre-trained knowledge (*e.g.*, biographical details) after injected new knowledge, ensuring that no catastrophic forgetting. Specifically, a question is generated based on the description of an entity (also natively included) other than the bold one in the fact (*e.g.* "Kanye West" in Table 2). **Portability**: Requires multi-hop reasoning between injected knowledge (e.g., "chorus written from a female perspective") and pre-trained knowledge (e.g., "groundbreaking artist... experimental beats"). See Appendix B for the question generation prompts.

Metrics and Models. We use substring match accuracy and token F1 as our evaluation metrics following the convention in open-domain QA. A simple question template is applied to facilitate answer generation: "{question}\nAnswer:". For evaluation, we choose 6 open-source LLMs from 3 different model families and scales. See more detailed description in Section 5.1 and Appendix A.

3.3 Static Analysis

We evaluate the zero-shot performance of off-the-shelf language models on WIKIDYK prior to knowledge injection (termed static performance), with results in Table 3. Despite these models reporting knowledge cutoffs extending to 2023, they exhibit near-chance accuracy on WIKIDYK, indicating the novelty of the provided knowledge. In contrast, performance on locality questions—which probe pretraining knowledge—is significantly higher, aligning with expectations for static model behavior. For completeness, we include static results on reliability questions from 2004–2009 in Appendix C.

3.4 Impact of Retrieval Augmented Generation (RAG)

We further analyze the impact of RAG-augmented models in Table 3 where we use all the collected Wikipedia articles as our retrieval data store and use a popular sentence embedding model[3] to retrieve top-k articles. While RAG consistently improves performance, its practical application in knowledge injection faces challenges: (1) the computational overhead of retrieving and processing external contexts remains prohibitive for applications that are latency-sensitive, and (2) reliance on external datastores complicates deployment pipelines and introduces potential privacy risks. These limitations underscore the importance of effective knowledge injection methods.

4 Knowledge Injection Approaches

In this paper, we focus on comparing approaches that can internalize new knowledge into model parameters. Specifically, we continue to pre-train various model architectures with different training objectives as introduced in Section 4.2 and Section 4.3. In Section 4.5, we introduce a collaborative framework that treats BiLMs as knowledge repositories and integrates them with LLMs.

4.1 Knowledge Injection Preliminaries

Knowledge injection (Fu et al., 2023; Zhang et al., 2023; Ovadia et al., 2023; Onoe et al., 2023; Xu et al., 2025) assumes that we have a pre-trained model $\mathcal{M}$ and a set of knowledge $\mathcal{T} = \{t\}$. The model is then trained on the knowledge via a training objective $\mathcal{M}' = \mathcal{L}(\mathcal{T}; \mathcal{M})$ to integrate $\mathcal{T}$ into

[3] BAAI/bge-small-en-v1.5

Model	Reliability		Generality		Paraphrase		Portability		Locality	
	Match	F1	Match	F1	Match	F1	Match	F1	Match	F1
Flan-T5-220M	0.15	3.27	2.50	5.88	0.11	3.33	0.19	2.84	5.46	14.22
Flan-T5-770M	0.23	3.70	3.39	7.68	0.24	3.80	0.25	3.15	10.47	16.45
Llama-2-7b	1.30	0.99	5.84	1.12	1.34	0.97	0.86	0.56	51.36	5.56
Llama-3.1-8B	1.94	1.05	7.68	1.30	2.18	1.06	1.00	0.59	58.52	11.51
+RAG	25.85	12.50	30.32	11.82	25.81	12.62	20.98	5.54	40.86	16.19
Llama-3.2-1B	0.46	0.75	3.59	0.82	0.62	0.81	0.28	0.46	27.99	4.49
+RAG	16.92	6.24	15.71	4.82	15.61	5.98	11.74	2.58	18.29	4.13
Qwen-2.5-1.5B	0.24	2.03	2.81	4.64	0.22	2.18	0.31	1.23	30.89	33.86
+RAG	21.92	15.57	27.12	15.71	22.03	15.24	18.60	11.74	19.49	10.17
Qwen-2.5-7B	0.86	3.04	4.66	6.22	0.78	2.93	0.84	1.30	44.28	40.10
+RAG	25.77	20.09	31.97	24.00	25.68	19.49	25.92	16.03	35.42	23.02
Gemma-3-1B-pt	0.51	0.60	4.33	0.65	0.58	0.64	0.31	0.62	26.58	1.92
+RAG	14.43	5.59	16.59	5.04	13.64	5.66	8.13	2.16	19.96	5.03

Table 3: Performance comparison between static models and +RAG. RAG retrieves top-3 Wikipedia articles per question. `Llama-2-7b` and `Flan-T5` are excluded from RAG due to context length limits.

$\mathcal{M}$. To conduct standardized comparisons, we define a *knowledge upsampling* parameter $s \in \mathbb{N}^+$ as the number of times a single knowledge entry $t \in \mathcal{T}$ is encountered during training, thereby controlling the amount of training.

4.2 Continued Pretraining for CLMs

Next Token Prediction We continue to pretrain the LLM on raw textual knowledge with next-token-prediction objective that maximizes the log-likelihood of $\sum_{i=1}^{l} \log p(t_{i+1}|C_L)$ for a text sequence t with token length l and $C_L = \{t_1, \cdots, t_i\}$ is the left context. We upsample by replicating each t for s times during training.

Synthetic QA Training Inspired by the approach proposed in (Wang et al., 2025b), we prompt `gpt-4.1-mini` to convert the factual knowledge into all possible forms of questions (see prompts in Appendix B). We then fine-tune the LLMs to predict the answer conditioned on the questions. Notice that here we use an external model for QA generation for its simplicity and quality. It is also able to use the corresponding instruct versions of open-source models as described in (Wang et al., 2025b). We upsample from the generated set of training QAs through replication to form the final training set.

Span Prediction In order to align with the training objective of BiLMs, we propose span prediction tasks for CLMs. Specifically, we format each input with a mask prediction prompt: "*Predict the masked words in the following sentence: {input_str}\nMasked words:*\n" where the input string is a corrupted text and the target is the span that recovers it. For a fair comparison with BiLMs, we employ the same masking strategy and upsampling as introduced in Section 4.3. At test time, we use the same prompt template and append a mask token after the question.

4.3 Continued Pretraining for BiLMs

Span Prediction We employ the span prediction objective from T5 (Raffel et al., 2020). Specifically, it maximizes the following log-likelihood during training $\sum_{(i,s)\in\mathcal{S}} \log p(t_i, \cdots, t_{i+s}|C_L, C_R)$ for a random masking strategy $\mathcal{S}$. The upsampling parameter is then $s = |\mathcal{S}|$ where C_L, C_R denote the left and right context. At test time, we simply append an extra token after the question in order to predict the answer to the question.

Exhaustive Masking Strategy In order to improve the sampling efficiency, we design a simple exhaustive sampling strategy. Specifically, we first generate all possible candidates of masked inputs given the minimum and maximum span lengths. During training, we generate a upsampled list of masked inputs based on the candidates. In this way, we guarantee the diversity of training samples while minimizing the upsampling parameter.

4.4 Why BiLMs Memorize Better? A Theoretical Perspective

In this section, we provide theoretical intuition for the empirical advantages of BiLMs over CLMs in knowledge memorization. Assuming both a BiLM $\mathcal{B}$ and a CLM $\mathcal{C}$ are optimally trained under the cross-entropy objective, we demonstrate that the accuracy gap between the two models is lower-

bounded by the conditional mutual information gained from accessing the right context. We formalize this in Theorem 1.

Theorem 1. *Let $X \in \mathcal{V}$ be a target span to be predicted, with contexts C_L and C_R on the left or right, and let $V = |\mathcal{V}|$. Consider two optimally trained language models: a BiLM ($\mathcal{B}$) and a CLM ($\mathcal{C}$). The accuracy gap between the two models satisfies:*

$$Acc_{\mathcal{B}} - Acc_{\mathcal{C}} \geq \frac{I(X; C_R|C_L)}{\log_2 V}$$

This theorem rigorously attributes the accuracy advantage of BiLMs to the additional mutual information released when the model is permitted to condition on future tokens. We defer the formal proof to Appendix D.

4.5 Ensemble Pipeline

Injecting an unbounded amount of new knowledge will inevitably diminish the effectiveness and incur catastrophic forgetting. Building on insights from modular architectures (Mitchell et al., 2022; Li et al., 2022; Feng et al., 2023), we propose a collaborative framework that coordinates multiple BiLMs as external knowledge repositories for LLMs. Our framework organizes external knowledge through two complementary partitioning strategies: (1) semantic clustering, where a Gaussian Mixture Model (GMM) groups facts into clusters based on their dense semantic embeddings, and (2) temporal clustering, which leverages fact timestamps to partition knowledge chronologically. To ensure robust routing, we train a scope classifier to discriminate between in-scope clusters (inter-class separation) and out-of-scope queries. The classifier is optimized using binary cross-entropy loss, where we assign a uniform label of 0 to all out-of-scope training instances. Negative training examples are derived from facts dated between 2004 and 2009. Each cluster is then internalized by a dedicated BiLM, forming a modular knowledge base. During inference, queries are either routed to the most relevant BiLM or deferred to the base LLM if deemed out-of-scope by a confidence threshold. This design ensures that the LLM's original knowledge remains intact, while injected knowledge is adaptively utilized through the BiLM ensemble, effectively mitigating catastrophic forgetting. Our framework thus enables synergistic integration of pretrained and external knowledge without compromising the LLM's foundational capabilities.

5 Results and Analysis

5.1 Experimental Setup

For **CLMs**, we train models from 3 model families including `Llama-2/3` (Touvron et al., 2023; Grattafiori et al., 2024), `Qwen2.5` (Yang et al., 2024) and `Gemma3` (Team et al., 2025) with different sizes using the objectives described in Section 4. We choose base models for knowledge injection since most knowledge is acquired at the pre-training stage. For comparison, we use full-parameter training for models less than 3B and use LoRA (Hu et al., 2022) with rank 32 and $\alpha = 16$ for other models. For learning rate, we use $2e-5$ for full-parameter training and $2e-4$ for LoRA training. For **BiLMs**, we choose `Flan-T5-220M/770M` (Chung et al., 2024), `T5(v1.1)-770M` (Raffel et al., 2020) and Roberta-large (Liu et al., 2019b). We train the full parameters for all BiLM models. For `Flan-T5-220M`, we choose a learning rate of $3e-4$ and for large versions of T5 models we use $1e-4$. Other hyperparameters can be found in Appendix A. Each trained model is then evaluated with all types of questions in the same way as mentioned in Section 3.2. Notice that for `Roberta-large`, we append fixed 10 mask tokens after the question for generation. For ensemble models, we train `DeBERTa-v3-large` (He et al., 2021) as our scope classifier and we integrate the injected BiLMs with `Llama-3.1-8B`.

5.2 Main Results

We demonstrate five insights below extracted from the main results in Table 4.

NTP objective is not suitable for knowledge injection. Notably, results on first four types of questions after NTP training are mostly lower than 1% for match accuracy or even token F1. These results are even lower than the static analysis in Table 3. We also observe catastrophic forgetting according to the locality performance after training. For example the locality match is decreased by 25.62% for `Llama-2-7b`. The poor generalizability can be attributed to both the formatting difference between training and evaluation, and the low context visibility for causal attention mask.

BiLMs are much more effective. The performances of both `Flan-T5-220M/770M` with span prediction are presented as a showcase for BiLMs. Despite smaller scales (220M for base version and 770M for large version), we found these models to

Model	Obj.	Reliability		Generality		Paraphrase		Portability		Locality	
		Match	F1	Match	F1	Match	F1	Match	F1	Match	F1
Llama-2-7b	NTP	0.90	1.04	7.20	1.42	1.09	0.99	0.58	0.66	25.74	2.67
	QA	9.02	13.56	18.38	28.58	4.86	9.33	0.98	4.22	44.60	**50.17**
	SP	10.93	13.90	13.45	14.61	8.40	11.46	1.60	3.55	42.15	41.41
Llama-3.1-8B	NTP	1.49	1.73	9.24	2.52	1.58	1.70	0.78	1.05	45.32	8.72
	QA	6.90	11.21	15.70	25.77	3.72	8.12	1.02	4.06	39.44	49.47
	SP	16.09	18.02	16.67	19.86	12.23	14.29	3.20	4.52	42.04	43.22
Llama-3.2-1B	NTP	0.45	1.65	0.54	7.75	0.43	1.29	0.36	0.64	0.51	6.88
	QA	16.92	18.84	29.80	41.35	13.83	15.95	1.55	2.59	4.81	5.66
	SP	3.03	5.88	4.96	5.95	1.72	4.35	0.55	2.40	16.46	22.34
Qwen-2.5-1.5B	NTP	0.74	3.30	1.00	12.40	0.69	2.92	0.50	0.98	1.28	19.06
	QA	19.08	4.41	**36.16**	7.36	15.83	3.85	4.62	0.97	12.16	4.72
	SP	1.06	1.03	7.21	1.43	1.13	1.03	0.39	1.09	21.80	2.38
Qwen-2.5-7B	NTP	0.46	1.21	5.04	2.19	0.58	1.17	0.64	1.59	28.71	32.51
	QA	2.29	3.91	8.61	9.24	0.74	3.23	0.39	2.48	29.08	29.92
	SP	2.01	1.72	2.73	9.00	1.83	1.46	1.86	0.64	10.20	30.31
Gemma-3-1B	NTP	0.44	1.14	0.60	5.93	0.47	1.03	0.32	0.60	0.39	5.94
	QA	8.17	11.81	15.28	23.96	4.06	7.74	1.04	3.98	27.19	31.73
	SP	7.37	10.41	9.61	11.50	5.50	8.32	1.62	3.98	28.04	32.18
T5(v1.1)-770M	SP	19.15	22.08	12.55	16.05	14.46	17.57	3.92	6.14	10.01	14.03
Flan-T5-220M		10.06	13.38	7.05	10.22	6.10	9.47	0.16	0.65	4.55	6.84
Flan-T5-220M (ens)		39.16	41.96	20.96	23.96	25.72	29.10	2.26	3.66	44.59	9.40
Flan-T5-770M		46.09	48.83	25.58	29.60	33.25	36.70	3.86	6.70	15.47	16.22
Flan-T5-770M (ens)		**52.82**	**53.85**	31.84	**34.91**	**40.02**	**42.04**	**6.56**	**8.14**	**49.58**	11.55

Table 4: Main results of knowledge injection with full dataset. Best results are bolded, and second-best are underscored. NTP stands for next-token-prediction, QA for synthetic QA and SP for span prediction. We set $s = 1,000$ for all experiments. More results of BiLMs can be found in Appendix C.

be much more effective than CLMs. For instance, Flan-T5-770M achieves 46.09% match accuracy on reliability, which reflects that the model can memorize almost half of the knowledge correctly. However, it is still not clear whether the effectiveness is derived from the diverse training examples produced by the random masking strategy or the architectural advantage. Thus it is important to compare the performance under controlled experiment on training objective.

Effectiveness of BiLMs may come from architecture rather than training objective. Results from synthetic QA and span prediction shows notable improvements against NTP baseline, and the former is usually more effective than the latter. For example, the reliability match is improved from 0.45 to 16.92 for Llama-3.2-1B. With span prediction, paraphrase match performance is improved by 7.31% for Llama-2-7b. More importantly, comparing span prediction results on both CLMs and Flan-T5 models, we see the significant superiority on the latter for the first four types of questions, especially considering the aligned training objective and number of upsampling during training. This directly shows that the performance gain of BiLMs may be related with architectural advantage and we encourage further analysis on this matter (see discussion in Section 6).

Ensemble pipeline can further improve the performance. We show the ensemble results with 10 Flan-T5 models and the rejected questions will be answered by Llama-3.1-8B. As shown in Table 4, ensemble models further improve the performance by 29.1% match accuracy on reliability for Flan-T5-220M and 6.73% for the large version. Furthermore, the match accuracy of locality is significantly improved for both versions of Flan-T5. We attribute the performance gain to the dedicated training on the assigned clusters and the cooperation between LLMs and the trained BiLMs. We show further analysis in Appendix 5.6.

Knowledge association shows less improvements. As can be observed in the table, the results for portability is less improved compared with other types of questions for all models and training objectives. Similar phenomenon is also observed in (Xu et al., 2025), who found that continued pre-training reliably recalls edited triples but fails on derivative association queries, and by (Zhong et al., 2023), where accuracy drops from nearly

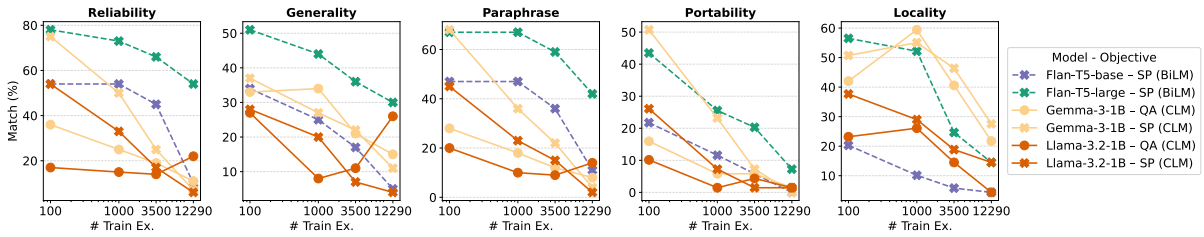

Figure 3: Effect of number of knowledge injected with number of upsampling $s = 1000$.

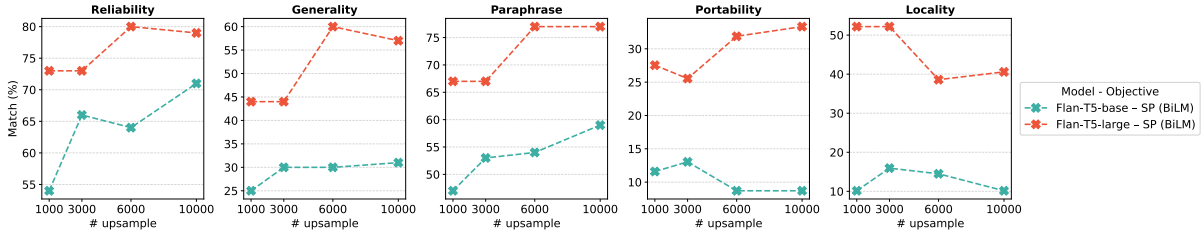

Figure 4: Effect of number of upsampling with 1000 injected knowledge.

90% on single-hop recall to below 15% on two-hop association.

5.3 Effect of Amount of Knowledge Injected

In order to understand the capacity of each model under different training objectives, we analyze the amount of knowledge injected in Figure 3. Specifically, we train the models with the first $100/1,000/3,500$ knowledge entries or the full dataset. For fair comparison, we evaluate all the models with the questions generated from the first 100 knowledge entries. As the number of training data progressively increases from 100 to 3500, the reliability and paraphrase for both `Gemma-3-1B` and `Llama-3.2-1B` decreases sharply while the performances for `Flan-T5` models stay relatively constant. However, the performance decreased significantly after injecting the full dataset especially for `Flan-T5-220M` which demonstrates the capacity limit of it. While single model capacity is always constrained, our ensemble pipeline can alleviate the issue by combining predictions from multiple specialized models. This approach not only compensates for the capacity limitations of smaller models like `Flan-T5-220M` but also leverages complementary strengths across architectures.

5.4 Effect of Number of Upsampling

To understand whether further increasing the number of upsampling could help the performance, we further conduct controlled experiments by training with first 1000 knowledge entries and increased number of upsampling. We found the performance for the first four question types further increases with increased number of upsampling except for the portability performance for `Flan-T5-220M` which stays between $5 - 15\%$. For instance, the match accuracy of paraphrase further increases by up to 15% with increased upsampling. However, the performance saturates after 6000 upsampling for most question types. We can also observe that locality performance is decreased by up to 10% after 3000 upsampling. For efficiency purpose, we control the number of upsampling in the main experiment as 1000.

5.5 Case Study

We show the case study for one of our trained model using `Flan-T5-770M` in Table 5. Despite low lexical similarity between the original fact and question, we still see that the predicted answers can match the ground truths, especially for the third example which shows a portability question. However, we do observe wrong matches, for example the last example in the table shows a hallucinated prediction against the ground truth.

5.6 Ablation Study on Clustering and Scope Classifier

To better understand the effect of clustering and scope classifier quality, we conduct ablation study in Table 6&11. Specifically, we ablate the performance between two kinds of the clustering algorithms: temporal and semantic. We found that temporal cluster consistently performs worse than semantic one. Furthermore, the performance for semantic clustering is improving with more number of clusters while the same is reversed for temporal one. We assume that this might be attributed to the failures of two possible components: scope classifier and knowledge injection. By using ground truth classifier in the ensemble pipeline, we ablate the

Fact	Question	Ground Truth	Prediction
AI expert Tess Posner resigned her role as a CEO in order to concentrate on her music career	Who stepped down from being a CEO to pursue their music career?	Tess Posner	AI expert Tess Posner
Journalist Jack Berry was influential in lifting the ban on female reporters in the locker room at The Masters	I've been reading about a storied men's golf championship held every spring in Augusta, Georgia, ... Who was that journalist?	Jack Berry	Jack Berry
Washington State Route 304 was accidentally removed ... for two years	How long was Washington State Route 304 accidentally removed from the state highway system?	two years	three years

Table 5: Case study on the model predictions. The backbone model is `Flan-T5-770M`, trained on the full dataset with $s = 3{,}000$. Correct predictions are highlighted in green, and incorrect ones in red.

Type	# clusters	Reliability		Generality		Paraphrase		Portability		Locality	
		Match	F1	Match	F1	Match	F1	Match	F1	Match	F1
Temporal	3	17.76	20.52	11.14	13.85	9.98	12.64	0.82	1.51	40.47	7.32
	5	15.18	17.41	11.49	14.00	8.79	11.12	0.84	1.46	35.48	7.21
	10	9.06	10.02	9.73	8.73	5.51	6.40	0.94	1.13	40.47	1.13
Temporal-perfect	3	27.13	30.42	15.23	18.57	18.23	21.74	1.32	3.21	49.58	7.66
	5	38.59	41.95	22.06	25.73	27.00	30.79	3.08	5.86	49.58	7.66
	10	44.24	47.23	27.12	30.19	32.87	36.57	6.46	9.82	49.58	7.66
Semantic	3	28.36	31.55	14.36	17.62	17.28	20.42	1.12	1.83	44.15	2.43
	5	32.16	34.86	16.94	19.79	20.63	23.46	1.55	2.48	45.02	5.00
	10	39.16	41.96	20.96	23.96	25.72	29.10	2.26	3.66	44.59	9.40
Semantic-perfect	3	29.70	33.25	15.54	19.18	19.06	22.80	1.29	3.03	49.58	7.66
	5	35.88	39.03	19.11	22.62	24.08	27.59	2.79	5.64	49.58	7.66
	10	44.23	47.23	27.12	30.19	32.87	36.57	6.46	9.82	49.58	7.66

Table 6: Ablation study on ensemble models using `Flan-T5-220M` as the base model.

effect of scope classifier, which can be found with "-perfect" results. The "Temporal-perfect" performs similarly with "semantic-perfect" which demonstrates that it is mainly the classifier that affects the performance of temporal clustering.

6 Conclusion and Discussion

In this work, we introduced WIKIDYK, a novel real-world, large-scale benchmark for knowledge injection that autonomously evolves over time, eliminating the need for manual updates. To rigorously evaluate knowledge capabilities in language models, we designed a multi-dimensional evaluation suite structured as question-answering tasks, probing both knowledge memorization and associative reasoning. Our extensive experiments reveal a critical limitation: under continued pre-training, Causal Language Models (CLMs) exhibit significantly weaker knowledge memorization compared to Bidirectional Language Models (BiLMs). To address this gap, we proposed a modular collaborative framework that integrates BiLMs as dynamic external knowledge repositories with LLMs. This approach not only compensates for CLMs' limitations but also achieves a 29.1% improvement in reliability, demonstrating the viability of model ensembles for knowledge-intensive tasks.

BiLMs vs. CLMs CLMs become the prevailing choice for LLM architecture. This is largely due to the low-latency generation and simple architecture design. In our experiments, we show that bidirectional attention, when paired with fill-in-the-blank type objective enable models to capture richer dependencies between tokens by leveraging both past and future contexts. Our findings suggest that bidirectional architectures excel in scenarios requiring dense knowledge integration, such as entity disambiguation, factual reasoning, or structured data understanding. However, these results do not negate the advantages of CLMs in generation tasks but instead highlight opportunities for hybrid architectures. uture work might explore dynamic attention mechanisms that adaptively toggle between bidirectional and unidirectional modes, or modular designs where specialized bidirectional components handle knowledge-intensive subtasks as suggested in the paper.

Limitations

We discuss the limitations in this section. Our work claims that BiLMs perform much better than the CLMs. However, we prove this assumption empirically with no theoretical guarantees. Furthermore, limited by the computing resource, we are not able to completely pre-train a BiLM and a CLM under the same set of hyperparameter and data. Instead, we choose the popular pre-trained models for experiments without further controlling the experiments.

Acknowledgments

Our work is sponsored in part by NSF CAREER Award 2239440, NSF Proto-OKN Award 2333790, Sponsored Research Projects from companies like Cisco and eBay, as well as generous gifts from Google, Adobe, and Teradata. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and should not be interpreted as necessarily representing the views, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for government purposes not withstanding any copyright annotation hereon.

Shangbin Feng would like to thank the support of the IBM PhD Fellowship, the Jane Street Graduate Research Fellowship, and the NVIDIA Graduate Fellowship.

References

Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2023. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*.

Hoyeon Chang, Jinho Park, Seonghyeon Ye, Sohee Yang, Youngkyung Seo, Du-Seong Chang, and Minjoon Seo. 2024. How do large language models acquire factual knowledge during pretraining? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Howard Chen, Jiayi Geng, Adithya Bhaskar, Dan Friedman, and Danqi Chen. 2024. Continual memorization of factoids in large language models. *arXiv preprint arXiv:2411.07175*.

Jeffrey Cheng, Marc Marone, Orion Weller, Dawn Lawrie, Daniel Khashabi, and Benjamin Van Durme. 2024. Dated data: Tracing knowledge cutoffs in large language models. *arXiv preprint arXiv:2403.12958*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and 1 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Shangbin Feng, Weijia Shi, Yuyang Bai, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. 2023. Knowledge card: Filling llms' knowledge gaps with plug-in specialized language models. *arXiv preprint arXiv:2305.09955*.

Peng Fu, Yiming Zhang, Haobo Wang, Weikang Qiu, and Junbo Zhao. 2023. Revisiting the knowledge injection frameworks. *arXiv preprint arXiv:2311.01150*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Roger B. Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamile Lukosiute, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. 2023. Studying large language model generalization with influence functions. *CoRR*, abs/2308.03296.

Anshita Gupta, Debanjan Mondal, Akshay Krishna Sheshadri, Wenlong Zhao, Xiang Lorraine Li, Sarah Wiegreffe, and Niket Tandon. 2023. Editing common sense in transformers. *arXiv preprint arXiv:2305.14956*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Jason Hoelscher-Obermaier, Julia Persson, Esben Kran, Ioannis Konstas, and Fazl Barez. 2023. Detecting edit failures in large language models: An improved specificity benchmark. *arXiv preprint arXiv:2305.17553*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Joel Jang, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. 2022. Temporalwiki: A lifelong benchmark for training and evaluating ever-evolving language models. *arXiv preprint arXiv:2204.14211*.

Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, Stanley Jungkyu Choi, and Minjoon Seo. 2021. Towards continual knowledge learning of language models. *arXiv preprint arXiv:2110.03215*.

Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A Smith, Yejin Choi, Kentaro Inui, and 1 others. 2023. Realtime qa: What's the answer right now? *Advances in neural information processing systems*, 36:49025–49043.

Yujin Kim, Jaehong Yoon, Seonghyeon Ye, Sangmin Bae, Namgyu Ho, Sung Ju Hwang, and Se-Young Yun. 2023. Carpe diem: On the evaluation of world knowledge in lifelong language models. *arXiv preprint arXiv:2311.08106*.

Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. 2024. Bitune: Bidirectional instruction-tuning. *arXiv preprint arXiv:2405.14862*.

Aochong Oliver Li and Tanya Goyal. 2025. Memorization vs. reasoning: Updating llms with new knowledge. *arXiv preprint arXiv:2504.12523*.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A Smith, and Luke Zettlemoyer. 2022. Branch-train-merge: Embarrassingly parallel training of expert language models. *arXiv preprint arXiv:2208.03306*.

Adam Liska, Tomas Kocisky, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, D'Autume Cyprien De Masson, Tim Scholtes, Manzil Zaheer, Susannah Young, and 1 others. 2022. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models. In *International Conference on Machine Learning*, pages 13604–13622. PMLR.

Jiahua Liu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2019a. Xqa: A cross-lingual open-domain question answering dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2358–2368.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jun-Yu Ma, Jia-Chen Gu, Zhen-Hua Ling, Quan Liu, and Cong Liu. 2023. Untying the reversal curse via bidirectional language model editing. *arXiv preprint arXiv:2310.10322*.

Nick Mecklenburg, Yiyou Lin, Xiaoxiao Li, Daniel Holstein, Leonardo Nunes, Sara Malvar, Bruno Silva, Ranveer Chandra, Vijay Aski, Pavan Kumar Reddy Yannam, and 1 others. 2024. Injecting new knowledge into large language models via supervised fine-tuning. *arXiv preprint arXiv:2404.00213*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36. ArXiv:2202.05262.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. 2022. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR.

Yasumasa Onoe, Michael JQ Zhang, Shankar Padmanabhan, Greg Durrett, and Eunsol Choi. 2023. Can lms learn new entities from descriptions? challenges in propagating injected knowledge. *arXiv preprint arXiv:2305.01651*.

Oded Ovadia, Menachem Brief, Moshik Mishaeli, and Oren Elisha. 2023. Fine-tuning or retrieval? comparing knowledge injection in llms. *arXiv preprint arXiv:2312.05934*.

Letian Peng, Chenyang An, Shibo Hao, Chengyu Dong, and Jingbo Shang. 2025. Linear correlation in lm's compositional generalization and hallucination. *CoRR*, abs/2502.04520.

Abhay Prakash, Manoj Kumar Chinnakotla, Dhaval Patel, and Puneet Garg. 2015. Did you know?-mining interesting trivia for entities from wikipedia. In *IJCAI*, pages 3164–3170.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Yi Ren and Danica J. Sutherland. 2025. Learning dynamics of LLM finetuning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Piotr Rybak, Robert Mroczkowski, Janusz Tracz, and Ireneusz Gawlik. 2020. Klej: Comprehensive benchmark for polish language understanding. *arXiv preprint arXiv:2005.00630*.

Zirui Song, Bin Yan, Yuhan Liu, Miao Fang, Mingzhe Li, Rui Yan, and Xiuying Chen. 2025. Injecting domain-specific knowledge into large language models: A comprehensive survey. *arXiv preprint arXiv:2502.10708*.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti

Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, and 1 others. 2023. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*.

Yu Wang, Yifan Gao, Xiusi Chen, Haoming Jiang, Shiyang Li, Jingfeng Yang, Qingyu Yin, Zheng Li, Xian Li, Bing Yin, Jingbo Shang, and Julian J. McAuley. 2024. MEMORYLLM: towards self-updatable large language models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Yu Wang, Dmitry Krotov, Yuanzhe Hu, Yifan Gao, Wangchunshu Zhou, Julian McAuley, Dan Gutfreund, Rogerio Feris, and Zexue He. 2025a. M+: Extending memoryllm with scalable long-term memory. *Preprint*, arXiv:2502.00592.

Yu Wang, Xinshuang Liu, Xiusi Chen, Sean O'Brien, Junda Wu, and Julian McAuley. 2025b. Self-updatable large language models by integrating context into model parameters. In *The Thirteenth International Conference on Learning Representations*.

Ruoxi Xu, Yunjie Ji, Boxi Cao, Yaojie Lu, Hongyu Lin, Xianpei Han, Ben He, Yingfei Sun, Xiangang Li, and Le Sun. 2025. Memorizing is not enough: Deep knowledge injection through reasoning. *arXiv preprint arXiv:2504.00472*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhengyan Zhang, Zhiyuan Zeng, Yankai Lin, Huadong Wang, Deming Ye, Chaojun Xiao, Xu Han, Zhiyuan Liu, Peng Li, Maosong Sun, and 1 others. 2023. Plug-and-play knowledge injection for pre-trained language models. *arXiv preprint arXiv:2305.17691*.

Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. 2023. Mquake: Assessing knowledge editing in language models via multi-hop questions. *arXiv preprint arXiv:2305.14795*.

A More Details in Experiments

We illustrate the details of hyperparameters selected in this paper. For all the experiments, we use up to NVIDIA-A100 80G GPU for training and evaluation. We train for 1 epoch if not specified. We use a minimum span length of 1 and maximum of 5.

Next-token-prediction We train with NTP objective on 8 GPUs where the batch size is 256 and upsampling is 1000. The learning rate and model configurations are described as in Section 5.1.

Synthetic QA We train with synthetic QA objective on 8 GPUs where the batch size is 256 and upsampling is 1000. The learning rate and model configurations are described as in Section 5.1.

Span Prediction For CLMs, we train with span prediction objective on 4 GPUs where the batch size is 128 and upsampling is 1000. The learning rate and model configurations are described as in Section 5.1. For BiLMs, we train with span prediction on $2/4$ GPUs where the batch size 128 and upsampling is 1000. The learning rate and model configurations are described as in Section 5.1.

Scope Classifier For scope classifier, we train with `DeBERTa-large-v3` where we set the learning rate $2e-5$ and batch size 128. We train for 10 epochs.

Metrics We evaluate model performance using two metrics: (1) Match, a binary indicator that returns true if the target string is a substring of the model's output (case-sensitive), and false otherwise; and (2) Token-Level F1, the harmonic mean of precision and recall computed by aligning the model's output tokens with the target tokens. The Match metric assesses strict presence of the target sequence, while the Token F1 score quantifies partial lexical overlap, offering complementary insights into generation quality. Both metrics are computed using whitespace-based tokenization to ensure consistency with standard text generation benchmarks.

B Prompts for Question Generation

B.1 Reliability Question Generation Prompt

We use `GPT-4o` for generating reliability QAs with the following prompt:

Listing 1: Prompt for reliability QA Generation

```
Given a DYK fact in JSON format
containing 'text' and 'bold_entity'
fields, generate a question. Your
output should be a JSON object
containing the question with its
corresponding answer. Your response
should follow these criteria:

1. The question should be answerable
using only the information provided in
the fact
2. The answer should be the bold_entity
3. The question should be clear,
natural, and specific so that the
answer can be easily identified (i.e.,
use as many details as possible from
the fact)
```

```
4. The bold entity should not be
mentioned in the question since it is
the answer. But make sure that the
question's answer is the bold entity.

Example:
Input:
{{
   'text': 'that Margrit Waltz has
   ferried planes to points on five
   continents?',
   'bold_entity': 'Margrit Waltz',
}}

Expected output:
{{
    "question": {{
        "text": "Who has ferried planes
        to points on five continents?",
        "answer": "Margrit Waltz"
    }}
}}

Now please generate a question with
answer for this fact:
{test_example}
```

B.2 Paraphrase Question Generation Prompt

We use `GPT-4.1` for generating paraphrase questions based on the reliability question with the following prompt:

Listing 2: Prompt for paraphrase QA generation

```
Given a pair of question and answer,
generate three different paraphrases of
the question. Make sure the answer is
the same as before. Your output should
be a JSON object with a list of
dictionaries under the key
"paraphrases". Each dictionary should
have a "question" key and an "answer"
key. Here is the pair of question and
answer:

Question: {question}
Answer: {answer}
```

We pick the first paraphrased question from the generated list.

B.3 Generality Question Generation Prompt

We use `GPT-4.1` for generating generality questions based on the reliability question and the fact with the following prompt:

Listing 3: Prompt for generality QA generation

```
Given a pair of question and answer,
generate three different alternative
questions. Make sure the question asks
about a different aspect of the same
fact. Remember to follow the rules
below:

1. The answer is one aspect of the fact
(such as an entity / year / number
etc.) apart from the original answer.
2. The answer should be concise and
direct without any redundant words. And
it shoud be a part of the fact.
3. The question should utilize all the
information in the fact and be specific.
4. Do not use any information that is
beyond the fact.
5. Your output should be a JSON object
with a list of dictionaries under the
key "alternatives". Each sub-dictionary
should have a "question" key and an
"answer" key.

Here is the pair of question and answer:

Fact: {fact}
Question: {question}
Answer: {answer}
```

B.4 Portability Question Generation Prompt

To generate the portability questions, we need to first identify an entity from the original knowledge entry that is non-bolded and have an associated Wikipedia article. After that, we prompt the `o3-mini` to first generate an entity description given its Wikipedia link (Notice that we discard those knowledge entries where the entity can not be found or the link is broken). The following is the entity description prompt:

Listing 4: Prompt for entity description generation

```
Replace the entity name with a
description of it without mentioning
the entity name. The description should
be unique and specific. Make sure that
you can infer the entity name using the
description. You might also be provided
with the wikipedia page of the entity.
The output should be a JSON object with
the following format:

{{
    "description": "The description of
    the entity",
}}

Wikipedia page: {page}
Entity name: {entity}
```

Then, we further prompt `o3-mini` to generate multi-hope QA pairs by replacing the entity with the description based on the reliability question.

Listing 5: Prompt for portability QA generation

```
Below are a few examples of natural,
scenario-based questions where a user
describes a scenario and then asks a
question:

Example 1:
```

```
Alternative description: "a historic
European city known for its iconic
architecture and cobblestone streets."
User's natural question: "I recently
visited a charming European city famous
for its unique architecture and quaint
streets. Can you tell me about a famous
monument there?"
Entity name: Paris

Example 2:
Alternative description: "a
groundbreaking technology company that
revolutionized communication with its
innovative products."
User's natural question: "I've been
reading about a tech company that
changed how we communicate through its
innovative gadgets. What product are
they best known for?"
Entity name: Apple

Now, given the alternative description
and the original question below,
generate a new, natural, scenario-based
question. The new question should
describe a scenario without mentioning
the original entity name and then ask
the question in a natural,
conversational manner.

Alternative description: {description}
Entity name: {entity}
Original question: {question}

The output should be a JSON object with
the following format:

{{
    "question": "The modified question"
}}
```

B.5 Locality Question Generation Prompt

We use `gpt-4.1` for locality question generation based on the previously generated entity description.

Listing 6: Prompt for portability QA generation

```
You'll generate a question-answer pair
based on the description of an entity.

For each statement, you'll return a
JSON object containing:
1. "question": The question that
corresponds to the statement
2. "answer": The answer to the question

Example outputs:

1. Input: Jupiter is the largest planet
in our solar system.
Output:
{{
  "question": "What is the largest
  planet in our solar system?",
  "answer": "Jupiter"
}}

2. Input: The capital of France is
Paris.
Output:
{{
  "question": "What is the capital of
  France?",
  "answer": "Paris"
}}

Entity: {entity}
Description: {description}
```

B.6 Training QA Generation Prompt

We use `gpt-4.1-mini` for training QA generation for the approach described in Section 4.

Listing 7: Prompt for training QA generation

```
Given a context, please generate
related questions as comprehensively as
possible with corresponding answers.
The question has to be based on the
context and the answer should be a
short phrase.
This is an example:
Context: A small coastal town has a
beach known for its colorful sea glass.
The town hosts an annual festival
celebrating this unique feature with
art and conservation efforts.
Question: What attracts tourists to the
small coastal town
annually? Answer: The unique sea glass
beach.
Question: What is celebrated at the
town's annual festival?
Answer: The natural phenomenon of sea
glass.
Question: What type of activities are
featured at the festival?

Format your output in a JSON object
like the one below:
{{
    "questions": [
        {{
            "question": "What attracts
            tourists to the small
            coastal town annually?",
            "answer": "The unique sea
            glass beach."
        }},
        {{
            "question": "What is
            celebrated at the town's
            annual festival?",
            "answer": "The natural
            phenomenon of sea glass."
        }},
        {{
            "question": "What type of
            activities are featured at
            the festival?",
            "answer": "Art and
            conservation efforts."
        }}
    ]
}}
Context: {fact}
```

C More Results

We present more results in this section. First, we show the numerical results that correspond to Figure 3 in Table 7-9. Notice that we also experimented with MemoryLLM (Wang et al., 2024) and M+ (Wang et al., 2025a). Results show that despite their claimed long-context, they are not able to utilize the questions for answer our evaluation questions, which results in lower performance compared with knowledge injection. We can also observe the performance of other BiLMs in Table 8 where we see that `T5-v1.1-large` also performs better than CLMs while lower then `Flan-T5` models. Thus, we only include `Flan-T5` models in our main results. Second, we also show the performance of reliability questions from 2004 to 2009 in Table 10. As can be observed, the performance decreases progressively with newer questions.

D Theoretical Insights

Proof of Theorem 1 introduced in Section 4.4.

Proof. Recall Fano's inequality that provides a lower bound on the probability of error for predicting X given context C.

$$P_e \geq \frac{H(X|C) - 1}{\log_2 V} \tag{1}$$

where P_e denotes the probability of predicting wrong span ($\widehat{X} \neq X$). We apply this inequality to the two models.

$$P_e^{\mathcal{C}} - P_e^{\mathcal{B}} \geq \frac{H(X|C_L) - H(X|C_L, C_R)}{\log_2 V} \tag{2}$$

By the definition of accuracy ($Acc = 1 - P_e$), it can be transformed to the following.

$$Acc_{\mathcal{B}} - Acc_{\mathcal{C}} \geq \frac{H(X|C_L) - H(X|C_L, C_R)}{\log_2 V} \tag{3}$$

Because of the definition of conditional entropy $I(X; C_R|C_L) = H(X|C_L) - H(X|C_L, C_R)$, we have:

$$Acc_{\mathcal{B}} - Acc_{\mathcal{C}} \geq \frac{I(X; C_R|C_L)}{\log_2 V} \tag{4}$$

□

In practice, the bias introduced during optimization can provide errors on the bound.

Model	Obj.	Reliability		Generality		Paraphrase		Portability		Locality	
		Match	F1	Match	F1	Match	F1	Match	F1	Match	F1
Llama-3.1-1B	QA	17.00	19.66	27.00	35.71	20.00	24.96	10.14	15.96	23.19	33.00
	SP	54.00	49.47	28.00	26.27	45.00	41.86	26.09	22.53	37.68	38.89
Gemma-3-1B	QA	36.00	37.38	33.00	39.88	28.00	31.59	15.94	18.55	42.03	49.37
	SP	75.00	63.36	37.00	34.15	68.00	57.98	50.72	48.87	50.72	62.42
Flan-T5-220M	SP	56.00	58.69	34.00	33.73	47.00	49.83	21.74	28.49	20.29	20.87
Flan-T5-770M		78.00	79.27	51.00	49.67	67.00	66.87	43.48	45.50	56.52	73.72
MemoryLLM	—	13.00	0.75	13.00	0.58	10.00	0.75	4.35	0.53	5.80	0.47
Mplus		4.00	0.39	8.00	0.29	5.00	0.41	4.35	0.40	14.49	0.40

Table 7: Results of knowledge injection with first 100 knowledge entries.

Model	Obj.	Reliability		Generality		Paraphrase		Portability		Locality	
		Match	F1	Match	F1	Match	F1	Match	F1	Match	F1
Llama-3.2-1B	QA	15.00	18.39	8.00	15.47	10.00	13.03	1.45	7.33	26.09	32.00
	SP	33.00	29.87	20.00	20.01	23.00	21.28	7.25	11.02	28.99	37.13
Gemma-3-1B-pt	QA	25.00	30.08	34.00	41.27	18.00	22.97	5.80	9.72	59.42	62.95
	SP	50.00	39.66	27.00	24.44	36.00	30.73	23.19	21.24	55.07	49.46
roberta-large	SP	3.00	15.55	14.00	12.84	3.00	11.80	0.00	0.00	0.00	3.29
t5-large		32.00	34.88	19.00	22.28	28.00	31.91	8.70	10.22	24.64	29.13
t5-v1.1-large		52.00	53.78	34.00	32.66	42.00	41.23	21.74	24.54	46.38	42.75
Flan-t5-220M		54.00	57.56	25.00	28.97	47.00	50.57	11.59	15.50	10.14	15.30
Flan-t5-770M		73.00	73.97	44.00	43.22	67.00	68.21	27.54	35.76	52.17	58.21

Table 8: Results of knowledge injection with first 1000 knowledge entries.

Model	Obj.	Reliability		Generality		Paraphrase		Portability		Locality	
		Match	F1	Match	F1	Match	F1	Match	F1	Match	F1
Llama-3.2-1B	QA	14.00	16.21	11.00	16.79	9.00	13.56	4.35	8.88	14.49	20.00
	SP	17.00	20.74	7.00	9.66	15.00	17.46	1.45	6.41	18.84	26.75
Gemma-3-1B-pt	QA	19.00	22.34	21.00	29.55	12.00	14.25	5.80	9.28	40.58	47.54
	SP	25.00	24.04	22.00	23.64	22.00	21.89	7.25	12.00	46.38	45.65
Flan-t5-220M	SP	45.00	49.14	17.00	21.30	36.00	41.09	5.80	9.72	5.80	10.87
Flan-t5-770M		66.00	66.33	36.00	37.68	59.00	59.87	20.29	28.58	24.64	37.25

Table 9: Results of knowledge injection with first 3500 knowledge entries.

Model	2004		2005		2006		2007		2008		2009	
	Match	F1	Match	F1	Match	F1	Match	F1	Match	F1	Match	F1
Llama-3.1-8B	12.03	2.03	10.42	2.39	6.53	2.04	5.12	1.63	4.34	1.75	3.79	1.64
Llama-3.2-1B	2.22	1.24	0.69	1.12	0.43	1.02	0.84	1.03	0.58	1.04	0.62	0.99
Qwen-2.5-1.5B	2.65	6.08	1.39	5.49	0.60	4.44	0.43	3.64	0.43	3.73	0.33	3.50
Qwen-2.5-7B	6.66	10.36	6.05	10.36	2.66	7.08	2.39	6.53	1.95	6.13	1.48	5.32
Gemma-3-1B-pt	2.39	0.72	1.46	0.72	1.03	0.72	0.67	0.75	0.58	0.78	0.59	0.73

Table 10: Results for older reliability questions from 2004 to 2009.

Type	# clusters	Reliability		Generality		Paraphrase		Portability		Locality	
		Match	F1	Match	F1	Match	F1	Match	F1	Match	F1
Temporal	3	33.50	35.78	24.50	27.41	22.82	25.35	3.18	4.41	45.02	10.71
	5	19.92	22.38	17.81	21.09	12.95	15.48	2.02	2.83	35.48	5.69
	10	13.43	14.17	13.66	13.06	8.85	9.71	1.54	1.91	40.47	7.17
Temporal-perfect	3	51.57	53.88	34.50	38.12	41.51	44.24	10.11	13.90	49.58	7.66
	5	51.60	54.14	34.58	38.79	41.22	44.52	12.15	15.90	49.58	7.66
	10	65.70	66.72	42.69	46.15	55.13	57.18	20.24	23.76	49.58	7.66
Semantic	3	44.24	46.55	27.85	31.37	32.00	34.68	4.44	6.19	44.60	11.24
	5	44.03	45.93	29.48	32.66	32.46	34.75	4.27	5.73	45.02	5.00
	10	52.82	53.85	31.84	34.91	40.02	42.04	6.56	8.14	49.58	11.55
Semantic-perfect	3	46.62	49.17	30.25	34.21	35.79	38.84	6.60	9.97	49.58	7.66
	5	49.28	51.50	33.28	37.36	37.88	40.83	9.42	13.11	49.58	7.66
	10	56.57	57.62	36.29	39.77	45.71	47.80	11.88	15.02	49.58	7.66

Table 11: Ablation study on ensemble models using `Flan-T5-770M` as the base model.