ALEX:A Light Editing-knowledge Extractor

Minghu Wang[1,2,3], **Shuliang Zhao**[1,2,3*], **Yuanyuan Zhao**[2,3,4,5], **Hongxia Xu**[1,2,3]

[1]College of Computer and Cyber Security, Hebei Normal University, Hebei 050024, China
[2]Hebei Provincial Engineering Research Center for Supply Chain Big Data Analytics & Data Security, Hebei 050024, China
[3]Hebei Provincial Key Laboratory of Network and Information Security, Hebei 050024, China
[4]School of Mathematical Sciences, Hebei Normal University, Hebei 050024, China
[5]Dept of Information Engineering, Shijiazhuang College of Applied Technology, Hebei 050043, China

Abstract

The static nature of knowledge within Large Language Models (LLMs) makes it difficult for them to adapt to evolving information, rendering knowledge editing a critical task. However, existing methods struggle with challenges of scalability and retrieval efficiency, particularly when handling complex, multi-hop questions that require multi-step reasoning. To address these challenges, this paper introduces ALEX (A Light Editing-knowledge Extractor), a lightweight knowledge editing framework. The core innovation of ALEX is its hierarchical memory architecture, which organizes knowledge updates (edits) into semantic clusters. This design fundamentally reduces retrieval complexity from a linear $O(N)$ to a highly scalable $O(K + N/C)$. Furthermore, the framework integrates an Inferential Query Synthesis (IQS) module to bridge the semantic gap between queries and facts , and a Dynamic Evidence Adjudication (DEA) engine that executes an efficient two-stage retrieval process. Experiments on the MQUAKE benchmark demonstrate that ALEX significantly improves both the accuracy of multi-hop answers (MultiHop-ACC) and the reliability of reasoning paths (HopWise-ACC). It also reduces the required search space by over 80% , presenting a promising path toward building scalable, efficient, and accurate knowledge editing systems.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding and generation, serving as extensive repositories of world knowledge. However, a fundamental limitation of these models is the static nature of their internal knowledge, which is acquired during a resource-intensive pre-training phase. As the world evolves and new information emerges, LLMs can quickly become outdated, leading to the generation of factually incorrect or "hallucinated" responses. The critical task of updating, correcting, or extending the knowledge within these models without resorting to complete retraining is known as knowledge editing (Gu et al. 2024; Meng et al. 2022, 2023; Zhong et al. 2023).

Current approaches to knowledge editing predominantly fall into two categories: parametric and memory-based methods. Parametric methods, such as ROME (Meng et al. 2023) and MEMIT (Meng et al. 2022), directly modify the model's internal weights to instill new factual associations. While precise, these techniques often suffer from unintended consequences; editing one fact can interfere with related information, causing cascading errors that disrupt the model's reasoning capabilities, especially in complex multi-hop scenarios. In contrast, memory-based approaches like MeLLo (Zhong et al. 2023) and PokeMQA (Gu et al. 2024) store factual updates in an external memory, decoupling the editing process from the model's core parameters. This offers greater flexibility but introduces significant challenges in scalability and retrieval efficiency (Zhang et al. 2024; Zhuang et al. 2024).

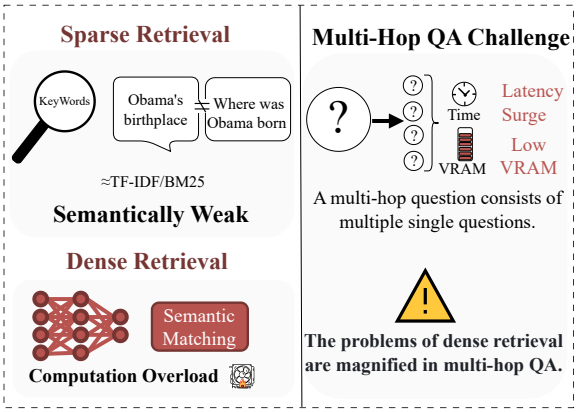

Figure 1: Sparse vs. Dense Retrieval: Speed vs. Accuracy.

These challenges are magnified in the context of multi-hop question answering, which requires models to decompose complex queries and synthesize information across multiple reasoning steps. The effectiveness of memory-augmented systems in this domain hinges on their ability to accurately and efficiently retrieve the correct edit at each step. However, a persistent "semantic gap" exists between the phrasing of user queries and the declarative format of stored edits, making retrieval fragile. This reflects a well-known trade-off: efficient sparse retrieval methods (Jiang et al. 2023; Trivedi et al. 2023) often fail to capture semantic nuances (Nguyen et al. 2024; Lee et al. 2023), while more accurate dense retrievers incur substan-

*Corresponding author

tial computational overhead (Reichman and Heck 2024; Gao et al. 2023). While strategies like Chain-of-Thought (CoT) have improved multi-step reasoning over static knowledge, they lack native mechanisms to incorporate external updates, often leading to error propagation when underlying facts change. This highlights a critical need for a knowledge editing framework that is simultaneously scalable, efficient, and robust to the semantic diversity of natural language queries.

To address these limitations, we propose ALEX (A Light Editing-knowledge Extractor), a novel framework that combines hierarchical memory compression with dynamic retrieval DEA to achieve accurate and efficient knowledge editing at scale. Crucially, as its name suggests, ALEX is designed not as a standalone editor but as a lightweight, modular extractor that can be seamlessly integrated into existing memory-based methods to replace their inefficient retrieval components. ALEX introduces a hierarchical architecture that transforms the edit memory from a flat, linear structure into a network of semantically organized clusters, using the K-means++ algorithm (Hassan et al. 2025) for its strong empirical performance and minimal overhead. This design fundamentally changes the retrieval complexity from a linear $O(N)$ to a highly scalable $O(K+N/C)$. Our framework is composed of three synergistic modules:

1. The **Semantic Manifold Partitioning (SMP) Engine** proactively organizes edits into semantically cohesive clusters, trained with a dual-objective that optimizes intra-cluster cohesion and inter-cluster contrastiveness (Ding et al. 2025).

2. The **Inferential Query Synthesis (IQS) Module** bridges the semantic gap by reformulating each stored fact into a diverse set of hypothetical questions, anticipating how a user might inquire about that information.

3. The **Dynamic Evidence Adjudication (DEA) Engine** implements an efficient two-stage retrieval process that first identifies relevant clusters and then pinpoints the most salient edit using both direct and inferential matching signals.

In summary, the contributions of our paper are as follows:

- A dual-objective training strategy unifies contrastive clustering with QA ranking, enabling rapid convergence (5 epochs) while preserving semantic fidelity across clusters.
- The **Inferential Query Synthesis (IQS)** and **Dynamic Evidence Adjudication (DEA)** modules collaborate to optimize retrieval. This synergy enhances matching precision through pseudo-query scoring and reduces the edit search space by over 80% through hierarchical filtering.
- ALEX achieves state-of-the-art performance on MQuAKE, demonstrating effectiveness in both accuracy and scalability.

2 Related Work

Knowledge Editing Techniques Contemporary approaches to knowledge editing in large language models bifurcate into parametric and memory-based methodologies. An "edit" itself is the specific factual update, often expressed as a declarative statement (e.g., "The Eiffel Tower is located in Paris") intended to correct or add to the model's knowledge.

Parametric methods, such as ROME and MEMIT, focus on modifying internal model weights to update factual associations. While ROME locates specific layers for targeted edits, MEMIT extends this capability to handle mass edits through multi-layer adjustments. However, these methods inherently suffer from cascading interference—editing one fact often disrupts related knowledge structures, leading to inconsistent outputs in multi-hop reasoning scenarios. Moreover, their reliance on static parameter updates renders them ill-suited for dynamic environments requiring frequent knowledge updates.

In contrast, memory-based approaches like MeLLo and PokeMQA decouple knowledge storage from model parameters by maintaining edits in memory. MeLLo iteratively prompts language models to align generated answers with stored edits, while PokeMQA introduces programmable scope detectors to disentangle conflict resolution from question decomposition. Despite their flexibility, these methods face challenges in scalability: memory consumption for dense retrievers (e.g., Contriever) grows linearly with the number of edits, and full-index similarity computations incur significant computational costs, particularly for multi-hop queries. Recent studies further highlight an accuracy-efficiency trade-off in lightweight retrievers, where aggressive compression strategies often degrade semantic matching fidelity. Furthermore, the retrieval effectiveness of these systems often hinges on the direct semantic overlap between a query and the stored declarative edits, creating a vulnerability when faced with the diverse phrasings and inferential nature of multi-hop questions.

Efficient Retrieval Paradigms Dense retrieval systems, exemplified by DPR(Karpukhin et al. 2020) and Contriever(Izacard et al. 2021), leverage transformer-based encoders to map queries and documents into continuous vector spaces. While these models excel at semantic matching, their computational overhead scales significantly with dataset size due to exhaustive similarity calculations. The storage demands of high-dimensional embeddings further exacerbate deployment challenges on resource-constrained platforms. Clustering-based optimizations, such as hierarchical indexing and product quantization, mitigate these issues by approximating nearest neighbor searches. However, such methods struggle with concept drift—static clusters fail to adapt to incremental edits, while retraining indices introduces operational latency.

Sparse retrieval systems, including BM25(Robertson and Zaragoza 2009) and SPLADE(Formal, Piwowarski, and Clinchant 2021), prioritize efficiency through inverted indices and term-frequency heuristics. Their interpretable keyword-matching mechanisms enable rapid document retrieval, particularly for lexically overlapping queries. Yet, their inability to handle semantic variations (e.g., paraphrases or polysemy) significantly limits performance in complex multi-hop scenarios. Hybrid architectures(Ren et al. 2023; Xia et al. 2024) attempt to reconcile these trade-offs but often introduce architectural complexity, as seen in

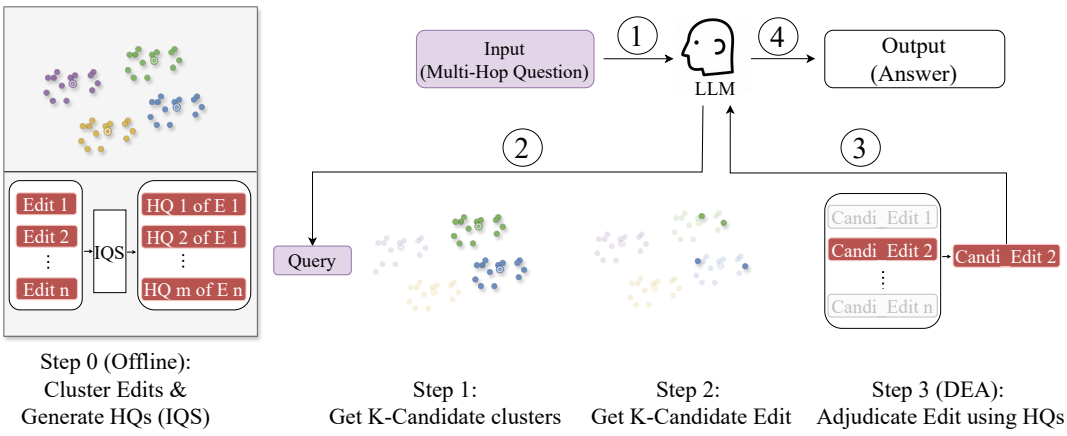

Figure 2: Workflow of our knowledge editing system.

late-fusion models that separately process lexical and semantic signals.

Multi-Hop Reasoning Architectures Chain-of-Thought (CoT) prompting has become a mainstream strategy for multi-step reasoning, enabling language models to decompose complex questions into intermediate subqueries. While effective for reasoning over static knowledge, CoT lacks mechanisms to incorporate external updates, often resulting in error propagation when underlying facts change. To address this, recent memory-augmented systems introduce retrieval at each reasoning step. However, as the number of edits grows and queries become more semantically complex, ensuring both retrieval accuracy and efficiency becomes increasingly challenging. The absence of adaptive reuse mechanisms further limits responsiveness, especially in real-time settings. Collectively, these limitations highlight a critical need for a knowledge editing framework that can scale efficiently, bridge the semantic gap between user queries and stored facts, and adapt dynamically to evolving knowledge within complex reasoning workflows.

3 ALEX

3.1 Framework Overview

Existing memory-based methods for multi-hop question answering, such as MeLLo and PokeMQA, struggle with scalability due to linear memory growth and high computational costs during retrieval. To overcome these challenges, we introduce **ALEX**, a memory-efficient framework built on a novel hierarchical architecture. At its core, ALEX partitions the entire knowledge edit memory into distinct semantic clusters. This design transforms the memory complexity from a linear $O(N)$ to a much more scalable $O(K + N/C)$, where N is the total number of edits, K is the number of clusters, and C is the average cluster size.

The framework's workflow is implemented through three synergistic modules, which are detailed in the subsequent sections: 1. The **Semantic Manifold Partitioning (SMP) Engine** proactively organizes the edits into a structured, hierarchical memory. 2. The **Inferential Query Synthesis (IQS) Module** bridges the lexical gap between user queries and stored facts by reformulating each edit into a set of diverse, hypothetical questions. 3. The **Dynamic Evidence Adjudication (DEA) Engine** executes an efficient, two-stage retrieval process.

When a multi-hop question is presented, it is first decomposed into simpler sub-queries. For each sub-query, the DEA engine rapidly identifies relevant clusters and then pinpoints the most salient edit within them, leveraging the synthesized questions from the IQS module to ensure robust matching. As illustrated in Figure 2, this integrated architecture ensures both high retrieval efficiency and accurate multi-step reasoning.

3.2 Semantic Manifold Partitioning (SMP) Engine

To address the scalability and semantic limitations of standard dense retrieval in multi-hop settings, we propose a **Semantic Manifold Partitioning (SMP) Engine** that hierarchically organizes the edit space. Unlike standard retrieval methods that suffer from linear computational growth and lack structural modularity, our SMP engine partitions the edit memory into semantically cohesive and structurally diverse groups. This not only reduces search redundancy but also facilitates efficient information updating and localized retrieval.

Dual-Objective Optimization. We design a training framework that jointly optimizes two objectives: promoting intra-cluster coherence and enforcing inter-cluster separability. Given a training batch, the model minimizes the following composite loss:

$$\mathcal{L}_{\text{train}} = \lambda \mathcal{L}_{\text{cohesion}} + (1 - \lambda)\mathcal{L}_{\text{contrast}}, \tag{1}$$

where the cohesion loss encourages the alignment of each edit with its assigned cluster centroid:

$$\mathcal{L}_{\text{cohesion}} = -\frac{1}{K}\sum_{c=1}^{K}\frac{1}{|\mathcal{G}_c|}\sum_{e\in\mathcal{G}_c}\cos\left(\phi(e), \mu_c\right), \tag{2}$$

and the contrastive loss promotes inter-cluster distinction. For each edit e_i in a batch B, we use a corresponding syn-

thetically generated query q_i as an anchor. The loss penalizes similarity between this anchor and other edits (negatives) in the batch from different clusters:

$$\mathcal{L}_{\text{contrast}} = -\frac{1}{B}\sum_{i=1}^{B}\log\frac{\exp\left(\cos(\phi(q_i),\phi(e_i))/\tau\right)}{\sum_{e_j\in\mathcal{N}_i}\exp\left(\cos(\phi(q_i),\phi(e_j))/\tau\right)}. \tag{3}$$

Here, λ is a hyperparameter balancing the two objectives, set to 0.4 based on performance on a held-out validation set. $\phi(\cdot)$ denotes the embedding function, $\cos(\cdot,\cdot)$ is the cosine similarity, and $\mathcal{N}_i$ represents the set of in-batch negatives for q_i. The clustering model is initialized using a K-means++ scheme where initial centroids are sampled with probabilities weighted by their similarity to a diverse set of pre-selected anchor edits, improving initial cluster quality. Further training and optimization details are provided in Appendix E.

Multi-Modal Representation. To support robust partitioning across semantically diverse edits, each edit e_i is encoded with a hybrid feature vector that incorporates both neural and lexical signals:

$$\mathbf{f}_i = \text{concat}\left(\phi(e_i), \frac{\ell(e_i)}{L_{\max}}, \frac{w(e_i)}{W_{\max}}\right), \tag{4}$$

where $\phi(e_i) \in R^d$ is the MPNet embedding, and $\ell(\cdot), w(\cdot)$ are the character length and word count of the edit, which are then normalized by their maximum values $L_{\max}$ and $W_{\max}$, respectively.

Dynamic Manifold Adaptation. Post-training, the SMP engine supports dynamic updates. This mechanism monitors cluster cohesion via metrics such as the silhouette score and automatically triggers partial reclustering when the average silhouette score of any cluster falls below a threshold $\theta_s = 0.5$, or the global average score drops by more than 20% relative to its post-training peak. This ensures the manifold structure adapts to evolving data distributions. The detailed procedure is elaborated in Appendix B.

3.3 Inferential Query Synthesis (IQS) Module

In memory-based knowledge editing, factual updates are typically stored as declarative statements. However, user queries—especially in multi-hop question answering—often differ from these statements in surface form. This discrepancy makes it difficult to retrieve relevant edits using direct query-edit similarity alone. To bridge this gap, the **Inferential Query Synthesis (IQS)** module reformulates each edited fact e_j into a set of N_h hypothetical questions $\mathcal{H}(e_j) = \{h_1^{(j)}, h_2^{(j)}, \ldots, h_{N_h}^{(j)}\}$ by prompting a large language model. These questions aim to capture diverse and plausible ways a user might inquire about the underlying fact. To ensure quality, the generated question set $\mathcal{H}(e_j)$ is scored using a composite function that balances semantic alignment with the edit against internal diversity. The score is calculated as a trade-off between a **relevance score** and a **redundancy penalty**. First, we define the average relevance of the hypothetical questions to the original edit e_j:

$$\mathcal{R}(e_j, \mathcal{H}(e_j)) = \frac{1}{N_h}\sum_{i=1}^{N_h}\cos(\phi(h_i^{(j)}), \phi(e_j)). \tag{5}$$

Next, we define the internal redundancy of the question set, measured as the average pairwise similarity between all questions:

$$\mathcal{D}(\mathcal{H}(e_j)) = \frac{2}{N_h(N_h-1)}\sum_{1\le i<k\le N_h}\cos(\phi(h_i^{(j)}), \phi(h_k^{(j)})). \tag{6}$$

The final quality score S_j for the question set is then given by:

$$S_j = \mathcal{R}(e_j, \mathcal{H}(e_j)) - \gamma\mathcal{D}(\mathcal{H}(e_j)), \tag{7}$$

where $\phi(\cdot)$ is the sentence embedding function shared across the system. The hyperparameter γ, which controls the trade-off, was set to 0.3 empirically on a validation set to penalize excessive redundancy without sacrificing relevance. Details on prompt templates, caching, and filtering heuristics are provided in Appendix C.

3.4 Dynamic Evidence Adjudication (DEA) Engine

To robustly identify the most relevant factual edit for a given query while ensuring efficiency, we propose the **Dynamic Evidence Adjudication (DEA) Engine**. This module addresses two central challenges: (i) the computational burden of exhaustive search, and (ii) the retrieval fragility against paraphrased or compositional queries. The DEA engine alleviates these issues via a two-stage pipeline that combines statistical filtering with semantically enriched scoring.

Stage I: Cluster-Level Statistical Filtering. Given a query embedding $\phi(q)$ and a set of K cluster centroids $\{\mu_i\}_{i=1}^K$, we first compute the normalized relevance of each cluster using a z-score transformation over cosine similarity. Let:

$$z_i = \frac{\cos(\phi(q), \mu_i) - \bar{s}}{\sigma_s}, \tag{8}$$

where $\bar{s}$ and σ_s are the mean and standard deviation of the cosine similarities between the query $\phi(q)$ and all cluster centroids. Clusters with $z_i \ge \zeta$ (we use $\zeta = 1.0$) are retained as candidate clusters. To ensure computational tractability, we cap the number of selected clusters at $M = 3$. This value was determined on a validation set to offer a robust balance between recall and efficiency. This two-stage approach—a dynamic z-score filter followed by a fixed cap—provides a flexible yet bounded selection mechanism. The z-score acts as a coarse-grained heuristic, while the cap M guarantees a predictable computational load, preventing the selection of an overly large set of clusters.

Stage II: Edit-Level Semantic Adjudication. Within the filtered clusters, each candidate edit e_j is scored using a composite function that integrates two forms of evidence: direct alignment (literal evidence) and reasoning consistency (inferential evidence). The literal evidence is measured by the direct cosine similarity between the query and the edit, while the inferential evidence is evaluated by matching the query against a set of hypothetical questions $\mathcal{H}(e_j)$. The final adjudication score is defined as:

$$\begin{aligned}\Psi(e_j) = \alpha\cdot\cos(\phi(q),\phi(e_j)) \\ + \beta\cdot\text{Agg}_{h\in\mathcal{H}(e_j)}\cos(\phi(q),\phi(h)).\end{aligned} \tag{9}$$

The trade-off parameters α and β are set to 0.5 to give equal weight to literal and inferential evidence. We employ max-pooling for the Agg operator, as it proved more effective in our preliminary experiments at isolating the signal from the single most relevant hypothetical question.

Selection. The edit with the highest adjudication score is deterministically selected:

$$e^* = \arg\max_{e_j} \Psi(e_j). \tag{10}$$

This two-stage process effectively prunes the search space. The z-score filtering discards statistically irrelevant portions of the memory, while the semantic adjudication stage uses the synthesized queries from the IQS module to ensure that even complex or paraphrased user queries can be robustly matched to the correct underlying fact. This synergy between statistical pruning and deep semantic evaluation allows the engine to achieve high efficiency without compromising retrieval fidelity.

4 Experiments

4.1 Experimental Setup and Baselines

We evaluate on five MQuAKE-related datasets (Zhong et al. 2023; Wang et al. 2024) (Appendix D) with three LLM backbones (Touvron et al. 2023; Dubey et al. 2024; Dai et al. 2024). Hardware and training details are in Appendix E. We compare our retriever, ALEX, against several non-parametric baselines. To ensure a fair comparison, the hyper-parameters for all baseline models were configured by following the recommendations in their respective original papers and validated on our held-out set. **MeLLo** (Zhong et al. 2023) is a memory-based method that decomposes multi-hop questions for external memory retrieval; its performance relies on the quality of decomposition. **DeepEdit** (Wang et al. 2024) is a non-parametric, decoding-based method that employs a depth-first search strategy to build reasoning chains. At each step, it retrieves candidate facts to guide the generation process. **PokeMQA** (Gu et al. 2024) uses a decoupled approach with a scope detector and knowledge prompts, whose accuracy is critical for performance.

4.2 Evaluation Metrics

We evaluate model performance using **MultiHop-ACC** (MA) and **HopWise-ACC** (HA) (Zhong et al. 2023; Saxena, Tripathi, and Talukdar 2020; Chen et al. 2020; Fei et al. 2022). To assess our retriever's effectiveness, we measure **Cluster ACC** (identifying the correct edit's cluster) and **Retrieval ACC** (retrieving the correct edit from the cluster) (Gaido et al. 2024; Li et al. 2024; Chen et al. 2023a; Cui and Sachan 2023; Chen et al. 2023b). Detailed definitions are in Appendix F.

4.3 Overall Performance Evaluation

We conduct extensive experiments to assess the effectiveness of the proposed ALEX framework across five datasets–MQuAKE-CF-3K, MQuAKE-CF-3K-v2, MQuAKE-T, MQuAKE-2002, and MQuAKE-Hard and three representative backbone families: DeepSeek, LLaMA 3.1 8B, and LLaMA 2 7B. As summarized in Table 1, ALEX consistently improves both MultiHop-ACC and HopWise-ACC across all model-dataset combinations.

DeepSeek setting. Under the DeepSeek backbone, ALEX yields substantial gains. On MQuAKE-CF-3K, MeLLo(ALEX) outperforms the vanilla MeLLo by +4.7% MultiHop-ACC and +7.3% HopWise-ACC, while PokeMQA(ALEX) achieves even larger improvements of +9.43% and +13.77%, respectively. Notably, on the more challenging MQuAKE-Hard dataset, PokeMQA(ALEX) achieves a +35.17% increase in HopWise-ACC, highlighting ALEX's advantage in long and difficult multi-hop scenarios.

LLaMA 3.1 8B setting. ALEX remains effective when applied to larger models. On MQuAKE-T, MeLLo(ALEX) achieves +7.02% and +19.72% improvements in MultiHop-ACC and HopWise-ACC, respectively. On MQuAKE-CF-3K-v2, PokeMQA(ALEX) surpasses its baseline by +5.47% in MultiHop-ACC and +5.78% in HopWise-ACC, showing the framework's transferability across scales and benchmarks.

LLaMA 2 7B setting. Even in more resource-constrained environments, ALEX consistently brings improvements. For instance, on MQuAKE-T, MeLLo(ALEX) improves MultiHop-ACC from 59.26% to 68.2% (+15.1% relative gain) and HopWise-ACC from 32.95% to 42.63%. These results confirm ALEX's scalability and robustness across varying model capacities.

Nuanced Performance on Smaller Models. Interestingly, a closer analysis of smaller models like LLaMA 2 7B reveals a nuanced trade-off. While ALEX substantially improves the reliability of the reasoning chain (HopWise-ACC), in some cases, the final answer accuracy (MultiHop-ACC) sees a marginal change. For example, on the MQuAKE-CF-3K dataset, applying ALEX to PokeMQA with LLaMA 2 7B improved HopWise-ACC from 22.91% to 36.80%, while MultiHop-ACC experienced a slight decrease (34.20% $\rightarrow$ 33.43%).

This pattern suggests that ALEX's hierarchical retrieval is highly effective at identifying the correct evidence for each reasoning step. However, smaller models may occasionally struggle to synthesize this correctly retrieved information into a perfect final answer. This indicates that while ALEX successfully fortifies the reasoning process, the final synthesis step remains a bottleneck for less powerful backbone models, a challenge that warrants further investigation.

Performance Across Reasoning Depths To analyze the impact of ALEX under varying reasoning complexities, we compare PokeMQA and PokeMQA(ALEX) on MQuAKE-CF-3K under 2-hop, 3-hop, and 4-hop conditions, as illustrated in Figure 3. ALEX yields progressively greater improvements as reasoning depth increases. In particular, the gains under 4-hop settings are the most pronounced, indicating that ALEX significantly enhances the model's ability to maintain long-range logical coherence–an area where conventional QA models often struggle.

LLM	Method	M-CF-3K		M-CF-3K-v2		M-CF-T		M-CF-2002		M-CF-Hard	
		MA	HA	MA	HA	MA	HA	MA	HA	MA	HA
DeepSeek	DeepEdit	37.88	26.02	39.81	31.74	80.00	62.95	48.53	35.57	49.31	41.13
	DeepEdit (ALEX)	**41.95**	**30.12**	**44.88**	**34.96**	**84.75**	**70.23**	**54.87**	**41.98**	**54.92**	**47.96**
	MeLLo	31.70	21.55	36.50	25.70	80.60	61.92	42.70	30.65	2.30	2.85
	MeLLo (ALEX)	**36.40**	**28.85**	**42.00**	**33.40**	**92.70**	**81.64**	**49.10**	**40.93**	**2.60**	**3.95**
	PokeMQA	41.90	29.83	45.00	32.90	78.80	56.47	55.50	38.21	39.00	25.78
	PokeMQA (ALEX)	**51.33**	**43.60**	**53.50**	**47.43**	**87.33**	**76.49**	**65.58**	**58.42**	**79.20**	**74.35**
Llama3.1 8B	MeLLo	8.36	3.78	9.16	4.03	46.78	61.92	10.08	4.59	0.69	1.04
	MeLLo (ALEX)	**9.61**	**4.81**	**10.53**	**5.24**	**53.80**	**81.64**	**11.59**	**5.03**	**0.79**	**1.61**
	PokeMQA	34.96	21.39	**41.00**	26.16	82.76	71.47	44.00	28.34	24.94	17.21
	PokeMQA (ALEX)	**40.43**	**34.23**	39.32	**26.76**	**85.70**	**76.01**	**46.30**	**40.60**	**37.29**	**30.76**
Llama 2 7B	DeepEdit	11.30	6.45	13.72	7.61	38.13	30.71	12.88	10.06	7.03	6.17
	DeepEdit (ALEX)	**14.05**	**8.03**	**16.04**	**8.52**	**44.96**	**37.98**	**14.97**	**11.98**	**8.48**	**7.47**
	MeLLo	16.10	8.58	14.50	6.85	59.26	32.95	15.68	8.44	3.26	0.55
	MeLLo (ALEX)	**18.50**	**11.25**	**16.70**	**8.69**	**68.20**	**42.63**	**16.18**	4.80	**3.75**	**0.91**
	PokeMQA	**34.20**	22.91	**37.20**	**24.81**	67.55	53.57	43.80	30.66	35.89	23.19
	PokeMQA (ALEX)	33.43	**36.80**	36.80	24.90	**78.53**	**64.34**	**45.70**	**31.46**	**40.56**	**28.90**

Table 1: Performance comparison of different methods on multi-hop question answering with knowledge editing across various datasets using different language models. Here, the prefix "M-" denotes datasets that are part of the MQuAKE-related benchmark. Evaluation metrics include Multi-hop Accuracy (MA) and HopWise Answer Accuracy (HA).

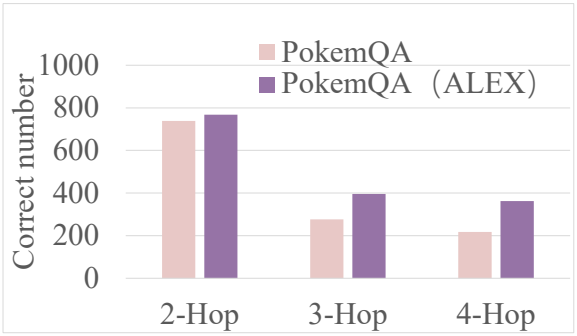

Figure 3: The comparison of the correct number of PokeMQA and PokeMQA (ALEX) under 2-Hop, 3-Hop, and 4-Hop conditions on the MQuAKE-CF-3K

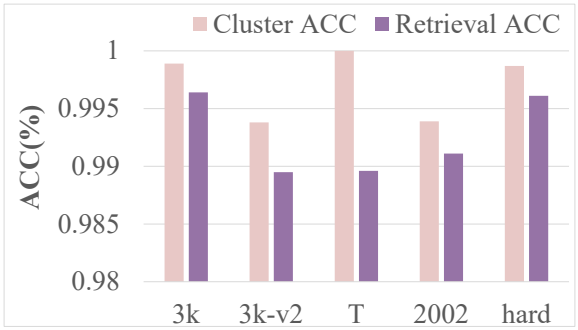

Figure 4: Performance of the clustering and retrieval components across five datasets. The number of clusters (K=12) was optimized for these datasets as detailed in Appendix B.

Component-wise Validation: Cluster and Retrieval Accuracy To validate our two-stage retrieval pipeline, we evaluated its coarse-grained clustering and fine-grained retrieval components. As shown in Figure 4, the high accuracy for both stages across all datasets confirms the effectiveness of our design for multi-hop inference.

Summary and Insights Overall, ALEX demonstrates substantial improvements across the vast majority of our experimental settings, with particularly strong gains on harder questions and longer reasoning chains. The improvements in HopWise-ACC are especially noteworthy, highlighting that ALEX not only helps models reach the correct final answer but also enhances the reliability of intermediate reasoning steps. While performance gains are universal with the powerful DeepSeek backbone, we observed a nuanced trade-off in smaller models like Llama 2 7B, where significant improvements in reasoning-path accuracy (HA) sometimes coincided with a marginal decrease in final-answer accuracy (MA), pointing to a complex interplay between retrieval architecture and model capacity.

4.4 Ablation Study

To understand the contribution of individual components in ALEX, we conduct two sets of ablation studies. The first focuses on the effects of our key modules, IQS and DEA, while the second analyzes the robustness of ALEX's clustering-based retriever.

Impact of IQS and DEA Modules We evaluate the role of the IQS and DEA modules by selectively disabling them on three representative datasets. As summarized in Table 2, removing either component causes a substantial performance drop. For instance, on MQuAKE-T, disabling both reduces MultiHop-ACC from 87.33% to 70.53%, confirming they make distinct and complementary contributions. Interestingly, the DEA mechanism shows stronger influence on

DEA	IQS	M-CF-3K-v2 MA	M-CF-3K-v2 HA	M-T MA	M-T HA	M-Hard MA	M-Hard HA
×	×	36.87	30.94	70.53	59.79	62.90	57.24
√	×	48.17	42.68	82.07	71.74	67.55	62.17
×	√	41.75	35.15	75.92	64.04	74.84	69.77
√	√	**53.50**	**47.43**	**87.33**	**76.49**	**79.20**	**74.35**

Table 2: Ablation study of the DEA and IQS modules on three MQuAKE datasets (M-CF-3K-v2, M-T, and M-Hard), showing their individual and combined contributions to performance.

MQuAKE-CF-3K-v2, while IQS exerts more impact on MQuAKE-Hard. This suggests that DEA excels at resolving ambiguities in shallow queries, while IQS better supports deeper entity disambiguation.

Retrieval Effectiveness Across Clustering Granularity We further investigate how the retriever's clustering granularity affects its performance. To analyze robustness, we manually varied the number of clusters (K), deviating from the automatically selected value, and measured performance. This analysis demonstrates ALEX's stability across a range of non-optimal configurations. In Figure 5, we report Cluster ACC and Retrieval ACC under different K values. Results show that both metrics remain stable across different granularities, with optimal performance typically achieved when K is around 12. This validates the reliability of ALEX's coarse-to-fine retrieval strategy.

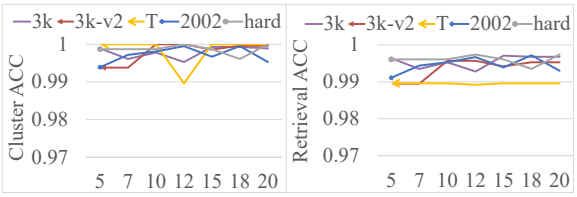

Figure 5: The figure illustrates Cluster ACC and Retrieval ACC under varying numbers of cluster classifications.

4.5 Efficiency Analysis

Building on the previous analysis of clustering granularity, we now assess retrieval efficiency under the same varying cluster configurations (K = 7, 10, 12, 15, 18, and 20). This analysis quantifies the reduction in the edit search space, with results presented in Figure 6.

As shown, pre-clustering significantly reduces the number of retrieval candidates compared to a baseline without clustering, which must search the entire edit set. For instance, on MQuAKE-CF-3K-v2, the average retrieval count drops from 2764 to 368 when using 12 clusters—an 86.7% reduction in the search space. We observe a diminishing return as K increases: moving from 7 to 12 clusters yields large efficiency gains, while further increasing K leads to smaller improvements. This is expected, as finer clustering risks over-fragmentation.

This analysis, combined with the robustness results from our ablation study (Section 4.4), confirms that ALEX's coarse-to-fine retrieval is highly effective. It simultaneously achieves significant computational savings—reducing the search space by up to 86.7%—while maintaining stable and high retrieval accuracy across various cluster granularities. This synergy between efficiency and robustness highlights the practical value of our approach.

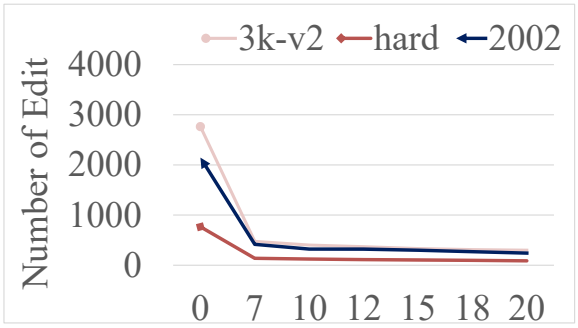

Figure 6: The figure depicts the number of edits that need to be retrieved per question on average for each dataset under different numbers of cluster classifications.

5 Conclusion

We introduce ALEX, a lightweight framework that addresses the scalability and accuracy limitations of knowledge editing in multi-hop reasoning. ALEX's core innovation is a hierarchical memory structure that reorganizes edits into semantic clusters, reducing retrieval complexity from O(N) to a highly scalable O(K+N/C). Complemented by inferential query synthesis and dynamic evidence adjudication modules, ALEX achieves state-of-the-art performance on MQuAKE datasets, significantly improving reasoning accuracy and reducing the search space by over 80%. While acknowledging a performance trade-off on smaller models, future work will focus on enhancing its query synthesis module. ALEX presents a promising path toward scalable, efficient, and accurate knowledge editing for large language models.

6 Limitations and Future Work

We acknowledge certain limitations. A nuanced performance trade-off was observed on smaller models, where improvements in reasoning-path fidelity (HopWise-ACC) did not always translate to equivalent gains in final-answer accuracy (MultiHop-ACC) . This suggests a potential bottleneck in the synthesis capacity of these models when processing the retrieved facts. Furthermore, the Inferential Query Synthesis (IQS) module, while effective, can be susceptible to ambiguous inputs.

Future work will focus on enhancing the IQS module through techniques like edit-conditioned prefix tuning. Regarding efficiency, this study prioritized validating the algorithmic approach. For the sake of experimental rigor, our subsequent work will involve developing a C++ implementation of ALEX. This will facilitate a direct and equitable performance benchmark against highly-optimized, low-level libraries like Faiss.

Acknowledgements

This research is partially supported by The National Social Science Fund of China No. 18ZDA200, Introducing Talents of Studying Overseas Fund of Hebei No. C20230339, Special Science and Technology Fund of Hebei Normal University No. L2023T03.

References

Chen, W.; Zha, H.; Chen, Z.; Xiong, W.; Wang, H.; and Wang, W. Y. 2020. HybridQA: A Dataset of Multi-Hop Question Answering over Tabular and Textual Data. In Cohn, T.; He, Y.; and Liu, Y., eds., *Findings of the Association for Computational Linguistics: EMNLP 2020*, 1026–1036. Online: Association for Computational Linguistics.

Chen, X.; Chen, T.; Chen, W.; Awadallah, A. H.; Wang, Z.; and Cheng, Y. 2023a. DSEE: Dually Sparsity-embedded Efficient Tuning of Pre-trained Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 8208–8222. Toronto, Canada: Association for Computational Linguistics.

Chen, Y.; Jin, D.; Huang, C.; Liu, J.; and Lei, W. 2023b. TRAVEL: Tag-Aware Conversational FAQ Retrieval via Reinforcement Learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 3861–3872. Singapore: Association for Computational Linguistics.

Cui, P.; and Sachan, M. 2023. Adaptive and Personalized Exercise Generation for Online Language Learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 10184–10198. Toronto, Canada: Association for Computational Linguistics.

Dai, D.; Deng, C.; Zhao, C.; Xu, R.; Gao, H.; Chen, D.; Li, J.; Zeng, W.; Yu, X.; Wu, Y.; Xie, Z.; Li, Y.; Huang, P.; Luo, F.; Ruan, C.; Sui, Z.; and Liang, W. 2024. DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1280–1297. Bangkok, Thailand: Association for Computational Linguistics.

Ding, J.; Yin, J.; Jia, L.; Fu, X.; and Wang, H. 2025. Clustering algorithm by Boundary Detection base on Entropy of KNN. *IEEE Access*.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Fei, Z.; Zhang, Q.; Gui, T.; Liang, D.; Wang, S.; Wu, W.; and Huang, X. 2022. CQG: A Simple and Effective Controlled Generation Framework for Multi-hop Question Generation. In Muresan, S.; Nakov, P.; and Villavicencio, A., eds., *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 6896–6906. Dublin, Ireland: Association for Computational Linguistics.

Formal, T.; Piwowarski, B.; and Clinchant, S. 2021. SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, 2288–2292. New York, NY, USA: Association for Computing Machinery. ISBN 9781450380379.

Gaido, M.; Papi, S.; Negri, M.; and Bentivogli, L. 2024. Speech Translation with Speech Foundation Models and Large Language Models: What is There and What is Missing? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 14760–14778. Bangkok, Thailand: Association for Computational Linguistics.

Gao, L.; Ma, X.; Lin, J.; and Callan, J. 2023. Precise Zero-Shot Dense Retrieval without Relevance Labels. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1762–1777. Toronto, Canada: Association for Computational Linguistics.

Gu, H.; Zhou, K.; Han, X.; Liu, N.; Wang, R.; and Wang, X. 2024. PokeMQA: Programmable knowledge editing for Multi-hop Question Answering. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 8069–8083. Bangkok, Thailand: Association for Computational Linguistics.

Hassan, E.; Malik, F.; Khan, Q. W.; Ahmad, N.; Sardaraz, M.; Karim, F. K.; and Elmannai, H. 2025. A Hybrid K-Means++ and Particle Swarm Optimization Approach for Enhanced Document Clustering. *IEEE Access*.

Izacard, G.; Caron, M.; Hosseini, L.; Riedel, S.; Bojanowski, P.; Joulin, A.; and Grave, E. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Jiang, Z.; Xu, F.; Gao, L.; Sun, Z.; Liu, Q.; Dwivedi-Yu, J.; Yang, Y.; Callan, J.; and Neubig, G. 2023. Active Retrieval Augmented Generation. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 7969–7992. Singapore: Association for Computational Linguistics.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In Webber, B.; Cohn, T.; He, Y.; and Liu, Y., eds., *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6769–6781. Online: Association for Computational Linguistics.

Lee, D.; Hwang, S.-w.; Lee, K.; Choi, S.; and Park, S. 2023. On Complementarity Objectives for Hybrid Retrieval. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 13357–13368. Toronto, Canada: Association for Computational Linguistics.

Li, R.; Liu, Q.; He, L.; Zhang, Z.; Zhang, H.; Ye, S.; Lu, J.; and Huang, Z. 2024. Optimizing Code Retrieval: High-Quality and Scalable Dataset Annotation through Large Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2053–2065. Miami, Florida, USA: Association for Computational Linguistics.

Meng, K.; Bau, D.; Andonian, A.; and Belinkov, Y. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 35: 17359–17372.

Meng, K.; Sharma, A. S.; Andonian, A. J.; Belinkov, Y.; and Bau, D. 2023. Mass-Editing Memory in a Transformer. In *The Eleventh International Conference on Learning Representations*.

Nguyen, T.; Chatterjee, S.; MacAvaney, S.; Mackie, I.; Dalton, J.; and Yates, A. 2024. DyVo: Dynamic Vocabularies for Learned Sparse Retrieval with Entities. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 767–783. Miami, Florida, USA: Association for Computational Linguistics.

Reichman, B.; and Heck, L. 2024. Dense Passage Retrieval: Is it Retrieving? In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Findings of the Association for Computational Linguistics: EMNLP 2024*, 13540–13553. Miami, Florida, USA: Association for Computational Linguistics.

Ren, Y.; Cao, Y.; Guo, P.; Fang, F.; Ma, W.; and Lin, Z. 2023. Retrieve-and-Sample: Document-level Event Argument Extraction via Hybrid Retrieval Augmentation. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 293–306. Toronto, Canada: Association for Computational Linguistics.

Robertson, S.; and Zaragoza, H. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.*, 3(4): 333–389.

Saxena, A.; Tripathi, A.; and Talukdar, P. 2020. Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J., eds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 4498–4507. Online: Association for Computational Linguistics.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2023. Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 10014–10037. Toronto, Canada: Association for Computational Linguistics.

Wang, Y.; Chen, M.; Peng, N.; and Chang, K.-W. 2024. Deepedit: Knowledge editing as decoding with constraints. *arXiv preprint arXiv:2401.10471*.

Xia, M.; Zhang, X.; Couturier, C.; Zheng, G.; Rajmohan, S.; and Rühle, V. 2024. Hybrid-RACA: Hybrid Retrieval-Augmented Composition Assistance for Real-time Text Prediction. In Dernoncourt, F.; Preoţiuc-Pietro, D.; and Shimorina, A., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 120–131. Miami, Florida, US: Association for Computational Linguistics.

Zhang, J.; Zhang, H.; Zhang, D.; Yong, L.; and Huang, S. 2024. End-to-End Beam Retrieval for Multi-Hop Question Answering. In Duh, K.; Gomez, H.; and Bethard, S., eds., *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 1718–1731. Mexico City, Mexico: Association for Computational Linguistics.

Zhong, Z.; Wu, Z.; Manning, C.; Potts, C.; and Chen, D. 2023. MQuAKE: Assessing Knowledge Editing in Language Models via Multi-Hop Questions. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 15686–15702. Singapore: Association for Computational Linguistics.

Zhuang, Z.; Zhang, Z.; Cheng, S.; Yang, F.; Liu, J.; Huang, S.; Lin, Q.; Rajmohan, S.; Zhang, D.; and Zhang, Q. 2024. EfficientRAG: Efficient Retriever for Multi-Hop Question Answering. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 3392–3411. Miami, Florida, USA: Association for Computational Linguistics.