When Should Knowledge Be Parameterized? A Study of Knowledge Allocation in Hybrid Language Models

Anonymous ACL submission

Abstract

Hybrid language models combine parametric learning with retrieval, yet it remains unclear which knowledge should be stored in model parameters and which should remain external.

Most existing systems implicitly duplicate knowledge across both components by fine-tuning on the full corpus while also retrieving from it at inference time. Although this strategy often improves accuracy, it obscures the question of how knowledge should be allocated between parametric and non-parametric memory.

In this work we study the knowledge allocation problem: deciding which knowledge should be parameterized and which should remain accessible through retrieval. We investigate whether intrinsic properties of knowledge—such as temporal stability, cross-context reuse, and compressibility—help explain when parameterization is beneficial.

Across question answering benchmarks we find that the benefit of parameterization varies across knowledge types. In particular, compressibility and reuse signals correlate more strongly with parameterization gains than other candidate properties in our experiments. While naïve hybrid systems achieve the highest raw accuracy by duplicating knowledge across components, property-based allocation strategies achieve competitive performance while parameterizing substantially less knowledge.

Overall, our results suggest that much of the benefit of hybrid training can be retained while parameterizing only about half of the knowledge corpus, highlighting the importance of principled knowledge allocation in hybrid language models.

1 Introduction

Large language models encode substantial knowledge in their parameters. However, parametric knowledge representations suffer from several limitations, including difficulty updating outdated information, memorization errors, and the high cost of retraining.

Retrieval-augmented generation (RAG) addresses some of these challenges by conditioning generation on external knowledge sources, allowing models to access up-to-date or instance-specific information.

As a result, hybrid language models systems that combine parametric training with retrieval—have become increasingly common. In practice, these systems are typically constructed by fine-tuning a language model on a knowledge corpus while also retrieving documents from that corpus at inference time. This approach often yields strong performance because knowledge can be accessed both from parameters and through retrieval.

However, this design implicitly duplicates knowledge across both components. The model learns information during training while also maintaining the same information in the retrieval corpus. While effective, this strategy raises a fundamental question:
Which knowledge should actually be stored in model parameters, and which should remain external?

We frame this question as the knowledge allocation problem and provide the first empirical study of how intrinsic knowledge properties relate to the effectiveness of parameterization in hybrid language models.

Most existing work focuses on architectural or inference-time decisions, such as how to retrieve documents, how to condition generation on retrieved passages, or when retrieval should be triggered. In contrast, we investigate a complementary question: what kinds of knowledge benefit from parameterization in the first place.

Our central hypothesis is that the suitability of knowledge for parameterization depends on intrin-

sic properties of the knowledge itself. For example, stable and widely reused knowledge may benefit from being encoded in model parameters, while volatile or instance-specific knowledge may be better handled through retrieval.

Across question answering benchmarks we observe three main findings:

1. Naïve hybrid systems achieve the highest raw accuracy, confirming the effectiveness of combining parametric knowledge with retrieval.
2. The benefit of parameterization varies across knowledge types, with compressibility and reuse signals showing clearer associations with parameterization gains.
3. Property-based allocation strategies achieve competitive performance while relying on substantially less parameterized knowledge.

Together, these results suggest that effective hybrid language models require not only improved retrieval mechanisms or training procedures, but also principled decisions about where knowledge should be stored.

Contributions. Our contributions are:

- We introduce the knowledge allocation problem in hybrid language models.
- We analyze intrinsic knowledge properties that may influence the effectiveness of parameterization.
- We provide a systematic empirical study of multiple allocation strategies across QA benchmarks.
- We show that property-based allocation strategies retain most of the benefit of naïve hybrid training while parameterizing substantially less knowledge.

2 Problem Formulation: Knowledge Allocation in Hybrid Models

We consider hybrid language models composed of two complementary components:

1. a parametric component, trained via fine-tuning on a subset of knowledge data;
2. a retrieval component, which accesses information from an external corpus at inference time.

Let K denote a set of knowledge units, where each unit corresponds to a coherent piece of information such as a paragraph or entity description.

The knowledge allocation problem asks how to partition K into two subsets:

- K_{param}: knowledge encoded in model parameters
- K_{retr}: knowledge accessed through retrieval

Formally, we define an allocation function

$$A : K \rightarrow \{\text{param}, \text{retr}\}$$

that determines where each knowledge unit should be stored.

Existing hybrid systems typically adopt implicit allocation strategies such as:

- **All-Parametric** — encode all knowledge through training
- **All-Retrieval** — rely exclusively on retrieval
- **Naïve Hybrid** — fine-tune on the full corpus while also retrieving from it

While naïve hybrid systems often achieve strong performance, they effectively duplicate knowledge across both components. In contrast, our work investigates whether intrinsic properties of knowledge units can guide allocation decisions.

3 Knowledge Properties Relevant to Allocation

We hypothesize that the usefulness of parameterization depends on intrinsic properties of the knowledge itself.

We focus on three properties that may influence the effectiveness of parameterization.

Temporal Stability. Knowledge that remains valid over long periods of time may benefit from parameterization, whereas rapidly changing information may be better handled through retrieval.

Cross-Context Reuse. Knowledge that appears across many contexts and tasks may benefit from parametric encoding, as the model can internalize reusable patterns.

Compressibility. Some knowledge can be represented compactly as generalizable patterns, while other information consists primarily of idiosyncratic instances. Highly compressible knowledge may therefore be more suitable for parametric storage.

4 Operationalizing Knowledge Properties

We operationalize knowledge properties using measurable signals derived from the training corpus.

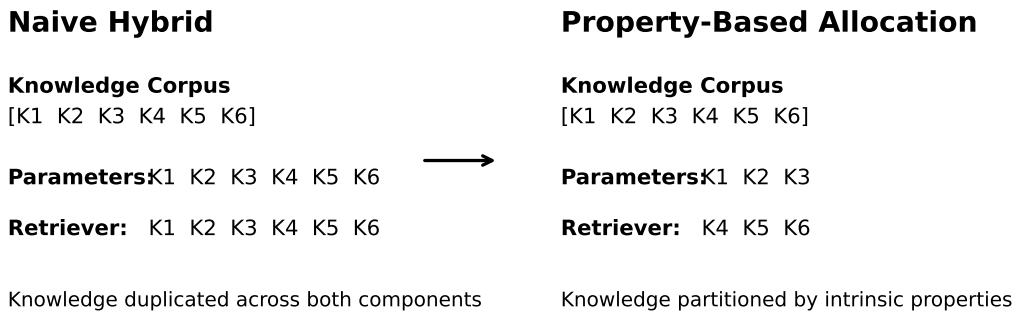

Figure 1: The knowledge allocation problem in hybrid language models. Naïve hybrid systems duplicate knowledge across parameters and retrieval, while property-based allocation partitions knowledge between the two components.

Reuse (frequency). We approximate cross-context reuse using the frequency with which a document title appears among the supporting documents of training QA examples. To reduce skew from highly frequent titles, we apply logarithmic normalization to the resulting counts.

Temporal stability. We estimate stability using the density of temporal markers in the document text, including date expressions, month names, and recency terms such as *currently* or *latest*. Documents with denser temporal references are assigned lower stability scores.

Compressibility. Compressibility is approximated using the gzip compression ratio of the document text. Text that is more regular or repetitive receives higher compressibility scores.

All property scores are computed at the document-title level using the full concatenated text of all chunks associated with that title. The resulting raw scores are then rank-normalized across the training corpus to produce values in $[0, 1]$, and the normalized title-level scores are assigned to all chunks belonging to that title.

Combined allocation heuristics. In addition to property-specific allocation strategies, we evaluate heuristics that combine multiple signals. In particular, we test a **stability–compressibility heuristic** that integrates both properties into a single allocation score. For each knowledge unit k, we compute stability $s_{\text{stab}}(k)$ and compressibility $s_{\text{comp}}(k)$, and define:

$$s_{\text{combined}}(k) = 0.5\, s_{\text{stab}}(k) + 0.5\, s_{\text{comp}}(k)$$

To ensure fair comparison across allocation strategies, we keep the parameterized fraction fixed across experiments. For each strategy, knowledge units are ranked according to the corresponding property score, and the top 50% are assigned to the parametric training set while the remaining 50% are indexed for retrieval. This design isolates the effect of the allocation heuristic itself rather than the amount of parameterized knowledge.

5 Experimental Setup

5.1 Datasets

HotpotQA. HotpotQA (Yang et al., 2018) is a multi-hop question answering dataset constructed from Wikipedia. Each example requires reasoning over multiple supporting paragraphs.

After preprocessing, the resulting training corpus contains 90,447 training questions and 482,021 documents used for training and allocation analysis (533,355 knowledge document chunks used for retrieval). We evaluate on 1,000 validation examples.

FreshQA. FreshQA (Li et al., 2024) is a temporally sensitive QA benchmark designed to evaluate models on knowledge with varying update frequencies. Since FreshQA does not provide a training corpus, it is used only for evaluation. We evaluate on 600 FreshQA examples.

5.2 Preprocessing

Documents are grouped by title and split into overlapping chunks using the base model tokenizer.

- Chunk size: 768 tokens
- Overlap: 16 tokens

Chunked documents are indexed for retrieval, while allocation properties are computed over the training corpus.

5.3 Model and Retrieval Setup

- Base model: Mistral-7B-Instruct-v0.2 (Jiang et al., 2023a)

Setting	Value
Base model	Mistral-7B-Instruct-v0.2
Embedding model	all-mpnet-base-v2
Retrieval top-k	3
Max new tokens	32
HotpotQA train QA pairs	90,447
HotpotQA document chunks	533,355
HotpotQA evaluation examples	1,000
FreshQA evaluation examples	600

Table 1: Core experimental settings used in all experiments.

- Retriever encoder: sentence-transformers/all-mpnet-base-v2 (Reimers and Gurevych, 2019; Song et al., 2020)
- Retrieval: top-$k = 3$ chunks
- Generation: max_new_tokens = 32

5.4 Evaluation Protocol

We report Exact Match (EM) and F1 for HotpotQA.

We also analyze three knowledge conditions:

- **Seen**: supporting passages were parameterized
- **Partial**: some supporting passages were parameterized
- **Unseen**: supporting passages must be retrieved

We additionally measure parameter efficiency, defined as the fraction of knowledge units assigned to parametric training. Because our focus is on knowledge allocation rather than generation faithfulness, we treat answer correctness as the primary signal for evaluating allocation strategies.

6 Results

We evaluate several knowledge allocation strategies against three canonical baselines:

- **All-Parametric**, which encodes the entire knowledge corpus in model parameters;
- **All-Retrieval**, which relies exclusively on retrieval at inference time; and
- **Naïve Hybrid**, which both fine-tunes on the full corpus and retrieves from it.

In addition to these baselines, we test several property-based allocation heuristics derived from the knowledge characteristics introduced in Section 4. These include stability-based allocation, compressibility-based allocation, a combination of the two, and a frequency-based heuristic that approximates cross-context reuse. A random allocation baseline performs comparably to some heuristics but does not outperform prop erty-based strategies, suggesting that simple property signals provide useful guidance beyond arbitrary partitioning.

For clarity, the main figures present only the strongest and most interpretable strategies. The full comparison across all evaluated allocation heuristics is reported in Appendix A.

Across experiments we observe three main patterns:

1. Naïve hybrid systems achieve the highest raw accuracy.
2. Property-based allocation strategies achieve competitive performance while parameterizing substantially less knowledge.
3. The benefit of parameterization varies across knowledge properties.

6.1 Overall Performance

Table 2 compares the performance of different knowledge allocation strategies on HotpotQA.

Naïve hybrid training achieves the highest overall performance, reaching 0.431 F1, confirming the effectiveness of combining parametric training with retrieval. This result is consistent with prior work showing that hybrid systems benefit from both memorized knowledge and retrieval grounding.

However, several property-based allocation strategies achieve competitive performance despite parameterizing only part of the knowledge corpus. In particular, stability-based allocation reaches 0.406 F1, and the combined stability–compressibility heuristic reaches 0.394 F1, both approaching the performance of naïve hybrid training.

Compressibility-based allocation and the frequency-based heuristic also perform reasonably well, though slightly below stability-based allocation. Together, these results suggest that selectively parameterizing knowledge based on intrinsic properties can recover much of the benefit of naïve hybrid training (roughly 94% of naïve hybrid performance) while parameterizing only about half of the knowledge corpus.

6.2 Seen vs. Unseen Knowledge

To better understand how allocation strategies affect knowledge usage, we analyze model performance depending on whether the supporting passages for a question were parameterized during training.

We analyze performance across the three knowl-

Strategy	EM	F1
All-Parametric	0.265	0.371
Naive Hybrid	**0.319**	**0.431**
Stability	0.311	0.406
Stability + Compressibility	0.289	0.394
Compressibility	0.282	0.385
Frequency	0.271	0.367

Table 2: Performance of knowledge allocation strategies on HotpotQA. Naive hybrid achieves the highest accuracy by parameterizing all knowledge. Property-based allocation strategies achieve competitive performance while enabling selective parameterization.

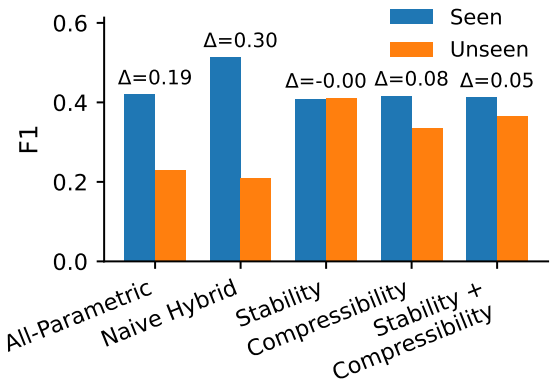

Figure 2: Performance on questions whose supporting passages were parameterized during training (seen) versus those requiring retrieval (unseen). Naive hybrid achieves the highest accuracy when knowledge is memorized but shows a larger drop when retrieval is required. Property-based allocation strategies exhibit more balanced behavior.

edge conditions defined in Section 5.4: *seen*, *partial*, and *unseen* knowledge.

Figure 2 shows performance differences between seen and unseen knowledge conditions.

Naïve hybrid systems achieve the highest accuracy when supporting knowledge has been parameterized. However, they also show a larger performance drop when supporting passages must be retrieved. This behavior reflects the strong reliance of naïve hybrid models on memorized knowledge acquired during training.

In contrast, property-based allocation strategies exhibit more balanced behavior across these conditions. Because only part of the knowledge corpus is parameterized, these models rely more consistently on retrieval during inference, reducing the performance gap between seen and unseen knowledge.

We also evaluated a frequency-based heuristic as a proxy for cross-context reuse. While it showed some evidence of parameterization benefit, its effect was weaker and less consistent than compressibility-based and stability-based allocation in our current setup. We therefore focus the main analysis on the strongest and most interpretable strategies, and report the full strategy comparison in the appendix.

This result suggests that selective parameterization may encourage hybrid models to use parametric and retrieval-based knowledge in a more complementary manner.

6.3 Knowledge Properties and Parameterization Benefit

Figure 3 summarizes how the parameterization advantage varies across the three knowledge properties we study.

To estimate whether a knowledge property predicts when parameterization is beneficial, we compare the performance of a naive hybrid model—which combines fine-tuning with retrieval—to a retrieval-only model. For each property, knowledge units are divided into high- and low-scoring groups. We then measure the difference in F1 between the naive hybrid model and the retrieval-only model on questions supported by those knowledge units. The resulting value, which we refer to as the *parameterization advantage*, reflects the performance gain obtained by adding parametric knowledge to a retrieval-based system. Property scores are computed independently of model predictions.

We observe that compressibility and reuse signals show clearer associations with parameterization gains, suggesting that knowledge exhibiting these properties may be particularly well suited for parametric encoding. In contrast, temporal stability is less consistently predictive in our experiments. This may reflect limitations in how stability is operationalized in our dataset, or more complex interactions between stability and retrieval behavior.

Overall, the results indicate that not all knowledge properties are equally informative for allocation decisions, and identifying reliable signals for parameterization remains an open challenge. We observe that compressibility and reuse signals show clearer associations with parameterization gains, suggesting that knowledge exhibiting these properties may be particularly well suited for parametric encoding. In contrast, temporal stability is less consistently predictive in our experiments. This may reflect limitations in how stability is operationalized in our dataset, or more complex interactions between stability and retrieval behavior.

Overall, the results indicate that not all knowl-

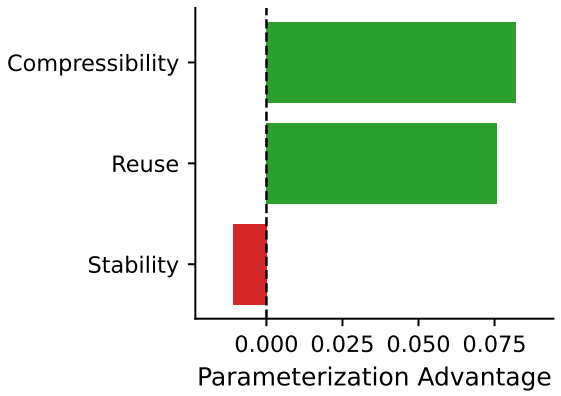

Figure 3: Parameterization advantage across knowledge properties. Values show the difference in F1 between a naïve hybrid model and a retrieval-only model on questions associated with high-scoring knowledge units for each property.

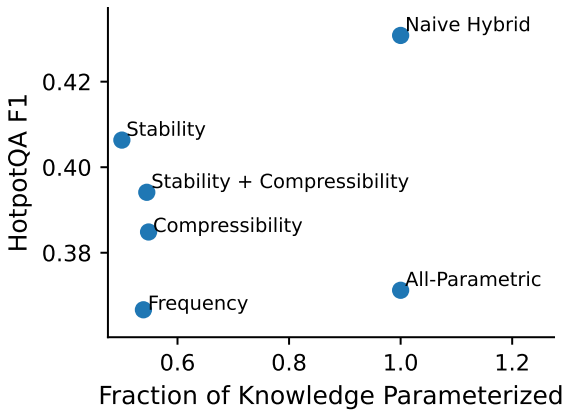

Figure 4: Parameter efficiency of different allocation strategies. Several property-based strategies achieve competitive performance while parameterizing roughly half of the knowledge corpus.

edge properties are equally informative for allocation decisions, and identifying reliable signals for parameterization remains an open challenge.

6.4 Parameter Efficiency

Finally, we examine how much knowledge must be parameterized to achieve competitive performance.

Figure 4 summarizes the relationship between the fraction of parameterized knowledge and model accuracy. Naïve hybrid systems parameterize the entire knowledge corpus while achieving the highest performance. However, several property-based strategies parameterize only about half of the knowledge units while retaining most of the hybrid model's accuracy.

For example, stability-based allocation parameterizes roughly 50% of the knowledge corpus while achieving 0.406 F1, compared to 0.431 F1 for naïve hybrid training. Similarly, compressibility-based strategies parameterize slightly more than half of the knowledge while maintaining competitive performance.

These results suggest that much of the benefit of hybrid training can be obtained without parameterizing the entire knowledge corpus, indicating that principled knowledge allocation may substantially improve parameter efficiency in hybrid language models.

6.5 FreshQA Evaluation

To examine whether our observations extend to temporally evolving knowledge, we evaluate the models on FreshQA, a benchmark designed to measure robustness to knowledge updates. FreshQA categorizes questions based on how frequently their answers change (*never*, *slow*, *fast*).

Model	Never F1	Fast F1
Retrieval	0.191	0.021
Naive Hybrid	0.300	0.059
Stability Allocation	0.271	0.018

Table 3: FreshQA evaluation across knowledge update rates. Parameterization provides larger gains for stable knowledge (*never*) than for rapidly changing knowledge (*fast*).

Table 3 reports results for stable (*never*) and rapidly changing (*fast*) questions. Consistent with our hypothesis, parameterization provides larger gains for stable knowledge. The naïve hybrid model substantially outperforms retrieval-only systems on *never* questions, while the advantage is smaller for rapidly changing knowledge.

These results suggest that the effectiveness of parameterization depends in part on the temporal stability of the underlying knowledge.

7 Analysis and Discussion

Our results provide new insight into how knowledge is represented and utilized in hybrid language models.

Across experiments, naïve hybrid systems achieve the highest raw accuracy. This is not surprising: naïve hybrid training effectively duplicates knowledge across both parametric memory and retrieval. As a result, the model can rely on memorized knowledge during generation while still benefiting from retrieval grounding when needed. The FreshQA results further support the intuition that parameterization is more beneficial for stable knowledge, while rapidly changing knowledge may be better handled through retrieval.

However, this strategy also has important limitations. Parameterizing the entire knowledge corpus increases training cost and reduces flexibility when knowledge must be updated. Moreover, duplicating knowledge across both memory mechanisms obscures which knowledge actually benefits from parameterization.

Our experiments suggest that selective parameterization guided by knowledge properties can provide a useful alternative. Property-based allocation strategies parameterize only part of the knowledge corpus while maintaining competitive performance. In particular, stability-based allocation achieves performance close to naïve hybrid training while parameterizing roughly half of the knowledge units.

These findings indicate that much of the benefit of hybrid training may come from parameterizing specific types of knowledge, rather than the entire corpus.

Our analysis also reveals that not all candidate knowledge properties are equally informative for allocation decisions.

In our experiments, compressibility and reuse signals show clearer associations with parameterization benefit than temporal stability. One possible explanation is that compressibility and reuse are easier to approximate with simple dataset statistics, whereas stability may depend on broader contextual or temporal factors that are not fully captured by our signals.

These results suggest that designing reliable indicators of parameterization suitability remains an open research challenge. Future work may benefit from more sophisticated property estimation methods or from combining multiple signals.

Taken together, the results suggest that hybrid language models should not treat parametric and retrieval memory as interchangeable sources of information.

Instead, the effectiveness of each memory mechanism appears to depend on the characteristics of the knowledge being represented. Knowledge that is frequently reused or highly compressible may benefit from parametric encoding, while other information may be better handled through retrieval.

Future hybrid architectures may therefore benefit from explicit mechanisms for allocating knowledge between parametric and non-parametric memory, rather than duplicating the entire corpus across both components.

8 Related Work

Research on knowledge representation in language models spans three closely related directions: retrieval-augmented generation, parametric knowledge in language models, and studies comparing parametric and non-parametric memory.

Retrieval-Augmented and Hybrid Language Models. A large body of work augments language models with external retrieval to improve factual accuracy and provide access to external knowledge. Early architectures such as REALM (Guu et al., 2020) and Retrieval-Augmented Generation (Lewis et al., 2020) demonstrated that combining parametric language models with document retrieval substantially improves performance on knowledge-intensive tasks. Dense retrieval methods such as DPR (Karpukhin et al., 2020) established retrieval as a standard component of open-domain question answering systems. More recent work explores stronger hybrid systems and retrieval-augmented few-shot learning (Izacard et al., 2023), as well as adaptive retrieval strategies that decide when to retrieve during generation (Jiang et al., 2023b; Asai et al., 2024). In contrast to this line of work, which primarily focuses on how and when to retrieve information, our work studies the complementary question of which knowledge should be stored in model parameters in the first place.

Parametric Knowledge in Language Models. Another line of research investigates the extent to which factual knowledge is encoded in model parameters. Petroni et al. (2019) showed that pretrained language models can answer factual cloze queries without external retrieval, suggesting that language models act as implicit knowledge bases. Subsequent work on closed-book question answering demonstrated that large language models can store substantial factual knowledge directly in their parameters (Roberts et al., 2020). Other studies have examined the limits of parametric knowledge, showing that memorization is uneven across fact types and domains (Heinzerling and Inui, 2021). Our work builds on this perspective but shifts the focus from whether language models store knowledge to how different types of knowledge should be distributed across parametric and retrieval-based memory.

Comparing Parametric and Non-Parametric Memory. Several recent studies directly com-

pare the strengths of parametric and retrieval-based memory. Most closely related to our setting, Mallen et al. (2023) show that retrieval is particularly beneficial for long-tail or less popular facts, while parametric memory performs well for more common knowledge. Follow-up work also investigates when retrieval helps or hurts language models and how retrieval can be triggered selectively based on model uncertainty or task requirements (Jiang et al., 2023b; Asai et al., 2024). These works reason primarily at the level of queries or inference-time decisions. In contrast, our work studies the structural question of knowledge allocation: whether intrinsic properties of knowledge itself predict whether it should be parameterized or retrieved.

Prior work has also explored the interaction between parametric fine-tuning and retrieval mechanisms in hybrid language models. Shaked (2026) study knowledge placement in small language models and show that combining fine-tuning with retrieval can improve question answering performance. While that work focuses on practical strategies for integrating the two components, the present study investigates a complementary question: whether intrinsic properties of knowledge itself can help determine when information should be parameterized or retrieved.

Interpreting and Editing Parametric Knowledge. A complementary line of research investigates how factual knowledge is represented inside model parameters. Prior work has identified knowledge-bearing neurons in transformer models (Dai et al., 2022) and demonstrated that factual associations can be edited directly in model weights (Meng et al., 2022). While these studies analyze how knowledge is stored after training, our work focuses on the earlier question of which knowledge should be parameterized in the first place.

While prior work has explored the interaction between parametric and retrieval-based knowledge, most studies focus on architectural design, retrieval strategies, or inference-time decisions. In contrast, our work examines the structural question of knowledge allocation: how knowledge itself should be distributed between parametric and non-parametric memory. Rather than treating hybrid models as systems that simply combine both mechanisms, we investigate whether intrinsic properties of knowledge can help predict when parameterization is beneficial. Our experiments provide the first empirical analysis of this question across multiple allocation strategies.

9 Conclusion

We introduced the knowledge allocation problem in hybrid language models: determining which knowledge should be stored in model parameters and which should remain accessible through retrieval.

Through a systematic empirical study, we showed that the effectiveness of parameterization varies across different types of knowledge. In particular, compressibility and cross-context reuse signals are more predictive of parameterization benefit than temporal stability in our experiments.

While naïve hybrid systems achieve the highest raw accuracy by duplicating knowledge across parametric and retrieval memory, property-based allocation strategies recover most of this performance while parameterizing substantially less knowledge—approximately half of the knowledge corpus in our experiments.

These results suggest that hybrid language models should treat knowledge allocation as a first-class design decision, rather than implicitly duplicating knowledge across components. We hope this work motivates further research on learning allocation policies and designing architectures that explicitly reason about where knowledge should reside.

10 Limitations

Our property signals are intentionally simple and may not fully capture the underlying structure of knowledge. In particular, temporal stability is approximated using textual markers rather than real-world update frequency, which may partly explain its weaker predictive power in our experiments. In addition, our experiments focus on QA benchmarks and a fixed 50/50 allocation regime. Future work should evaluate learned allocation policies, richer signals such as entity popularity or update frequency, and broader task settings beyond question answering.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *Proceedings of ICLR*.

Damai Dai, Li Dong, Yichong Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge

neurons in pretrained transformers. In *Proceedings of ACL*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. In *Proceedings of ICML*, pages 3929–3938.

Benjamin Heinzerling and Kentaro Inui. 2021. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In *Proceedings of EACL*, pages 2398–2410.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Pierre Savary, Chris Bamford, Devendra Chaplot, Diego de las Casas, Florent Bressand, Guillaume Lengyel, and 1 others. 2023a. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023b. Active retrieval augmented generation. In *Proceedings of EMNLP*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of EMNLP*, pages 6769–6781.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*.

Xinyu Li and 1 others. 2024. Freshqa: A benchmark for evaluating temporal generalization in question answering. In *Proceedings of ACL*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of ACL*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. In *Advances in Neural Information Processing Systems*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? In *Proceedings of EMNLP-IJCNLP*, pages 2463–2473.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of EMNLP*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of EMNLP*, pages 5418–5426.

Sigal Shaked. 2026. Optimizing knowledge placement in small language models: Combining fine-tuning and retrieval-augmented generation. In *Proceedings of the 15th International Conference on Pattern Recognition Applications and Methods*, pages 125–132. SCITEPRESS.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. In *Advances in Neural Information Processing Systems*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of EMNLP*.

A Additional Results

A.1 Full Results Across Allocation Strategies

For clarity, the main paper presents only the strongest and most interpretable allocation strategies in the primary tables and figures. Table 4 reports the full HotpotQA comparison across all evaluated strategies.

Strategy	EM	F1
Naive Hybrid	0.319	0.431
Stability	0.311	0.406
Random Split	0.291	0.397
Stability + Compressibility	0.289	0.394
Compressibility	0.282	0.385
Frequency + Stability	0.283	0.381
Frequency + Compressibility	0.281	0.375
All-Parametric	0.265	0.371
Frequency	0.271	0.367
All-Retrieval	0.069	0.193

Table 4: Full HotpotQA results across all evaluated allocation strategies.