Studying the Role of Input-Neighbor Overlap in Retrieval-Augmented Language Models Training Efficiency

Anonymous ACL submission

Abstract

Retrieval-augmented language models have demonstrated performance comparable to much larger models while requiring fewer computational resources. The effectiveness of these models crucially depends on the overlap between query and retrieved context, but the optimal degree of this overlap remains unexplored. In this paper, we systematically investigate how varying levels of query–context overlap affect model performance during both training and inference. Our experiments reveal that increased overlap initially has minimal effect, but substantially improves test-time perplexity and accelerates model learning above a critical threshold. Building on these findings, we demonstrate that deliberately increasing overlap through synthetic context can enhance data efficiency and reduce training time by approximately 40% without compromising performance. We specifically generate synthetic context through paraphrasing queries. We validate our perplexity-based findings on question-answering tasks, confirming that the benefits of retrieval-augmented language modeling extend to practical applications. Our results provide empirical evidence of significant optimization potential for retrieval mechanisms in language model pretraining.

1 Introduction

Language models that are pretrained with retrieval augmentation can match the performance of much larger models trained in the conventional way, while at the same time requiring significantly fewer computational resources (Borgeaud et al., 2022; Izacard et al., 2023). In retrieval-augmented pretraining, the model can query and incorporate information from external sources, which makes it easier to update its knowledge base and allows information to be added, removed, or modified in a transparent and flexible way (Izacard et al., 2023; Wang et al., 2023b; Shi et al., 2024b).

While research on retrieval-augmented language models has shown that accessing external sources reduces reliance on model parameters and leads to lower perplexity, questions remain about the underlying mechanisms driving these improvements. Recent work has explored this issue with a focus on the role of the retrieved context (Borgeaud et al., 2022; Norlund et al., 2023; Doostmohammadi et al., 2023). Findings suggest that the primary reason for reduced perplexity is *surface-level overlap*, i.e., exact token matches, between the queries and the retrieved context. Yet, the optimal degree of overlap is still unclear. Intuitively, while higher overlap appears to provide a stronger signal for language modeling, excessive similarity between queries and retrieved context may lead to over-reliance on retrieval and reduce model generalization in downstream tasks. This raises a fundamental question: what makes retrieved context effective during pretraining? An answer to this question could open the door to a well-founded methodology for designing retrieval corpora to maximize their usefulness for practical applications and training retrieval-augmented systems that rival the performance of much larger conventional language models under significantly tighter resource constraints—making advanced capabilities more accessible, adaptable, and sustainable.

In this paper, we take a significant step towards a deeper understanding of the role of retrieved context in augmented language modeling by systematically exploring how the degree of overlap between queries and context affects model performance both at training and at test time. To this end, we train multiple models under controlled levels of overlap and evaluate them in terms of perplexity and on downstream tasks. Building on our findings, we further investigate to what extent we can deliberately accelerate learning and enhance model performance in a low-resource scenario through data synthesis.

Contributions Our contributions are as follows:

- We investigate how varying degrees of overlap between queries and retrieved context affect test-time perplexity. Additionally, we analyze this variation over training steps, offering insights into how the impact of overlap depends on the amount of training data.
- To validate our findings, we include downstream performance results on a question answering task (Kwiatkowski et al., 2019), ensuring that the observed trends translate to real-world utility.
- We finally show how our findings can be used to train retrieval-augmented language models more data-efficiently than standard models. Specifically, we explore a method where we deliberately increase query–context overlap using synthetic contexts obtained through paraphrasing and find that this leads to faster perplexity reduction with less data.

2 Previous Work

Retrieval augmentation has been widely used in open-domain question answering and has also been applied to the pretraining and finetuning of language models (Karpukhin et al., 2020; Yogatama et al., 2021a; Borgeaud et al., 2022; Izacard et al., 2023; Wang et al., 2023a; Shi et al., 2024a).

Early work, such as that of Guu et al. (2020), explored retrieve-and-edit paradigms, while follow-up studies focused on selecting relevant evidence based on lexical overlap (Asai et al., 2020) or enhancing inference-time generation with retrieval (Khandelwal et al., 2020; Yogatama et al., 2021b). One line of work, exemplified by kNN-LM (Khandelwal et al., 2020), interpolates between model predictions and retrieved contexts at generation time. This approach was later extended in SPALM (Yogatama et al., 2021b), which introduced a learned gating mechanism that dynamically balances between both contributions.

Later efforts have shifted toward integrating retrieval earlier in the training pipeline. Borgeaud et al. (2022) demonstrated that large-scale retrieval-augmented pretraining can substantially reduce perplexity even with a frozen retriever. Izacard et al. (2023) further showed that jointly training the retriever and language model can provide additional performance gains, especially when retrieval is over extremely large datasets (trillions of tokens). Xu et al. (2023) found that approximate nearest neighbors have a positive effect on generalization, acting as a form of regularization.

Recent work has found that retrieval-augmented pretraining leads language models to acquire less world knowledge but improved syntactic proficiency (Samuel et al., 2024). This shows that such training shifts the role of the language model toward interpreting factual information from retrieved contexts, which, in practice, offloads knowledge from the model parameters and allows the use of smaller model sizes. Although the majority of published retrieval-augmented systems rely on relatively small language models, they still demonstrate significant improvements in perplexity, factuality, and downstream accuracy when pretraining is retrieval-enhanced (Borgeaud et al., 2022; Izacard et al., 2023). These findings suggest that retrieval-augmented pretraining is a promising direction for scaling language models more efficiently than approaches based on parameters alone.

3 Background: RETRO Architecture

In this paper, we experiment with retrieval-augmented language models based on the RETRO architecture (Borgeaud et al., 2022). This architecture is similar to GPT but is set up to predict the next token conditioned on an augmented context that, in addition to the previously generated tokens, includes additional tokens obtained via the retrieval mechanism. Technically, this is implemented via an additional cross-attention mechanism between the internal representations of the generated tokens and the encoded context. This design allows the model to incorporate information from the retrieval database without requiring it to be explicitly included in the generated token sequence.

Chunks The retrieval of additional context and its incorporation into next-token prediction is done at the level of *chunks*. A chunk is defined as a contiguous sequence of tokens with a fixed size, which is set as a hyperparameter. In both the original RETRO paper (Borgeaud et al., 2022) and our own work, the chunk size is $m = 64$.

Neighbors When the model has generated a new chunk C_u, that chunk is used as a query to retrieve k similar chunks from the retrieval database. In this context, the chunk C_u is conventionally called the *input chunk*, and the retrieved chunks are called the *neighbors* of C_u. Each neighbor N_u^i is addition-

ally concatenated with the chunk F_u^i that follows N_u^i in the retrieval dataset; that chunk is called the *continuation* of the neighbor. The rationale of the augmentation is that, since the neighbors are retrieved based on their similarity to C_u, their continuations are likely to be similar to C_{u+1}, the next chunk to be generated by the model, and should therefore be able to inform the generation of that chunk. For convenience, we generally use the term *neighbor* to include both the neighbor proper and its continuation.

Retrieval-augmented context In the following, we write $\text{RET}(C_u)$ to denote the retrieval-augmented context that the model uses to generate the tokens in the chunk C_{u+1}. The generation of the first chunk C_1 is not conditioned on any augmented context (only on the usual language modeling context), so $\text{RET}(C_0) = \emptyset$. For $u \geq 1$, the retrieval-augmented context is

$$\text{RET}(C_u) \triangleq ([N_u^1, F_u^1], \ldots, [N_u^k, F_u^k]) .$$

Note that in RETRO, the retrieval happens offline and does not involve any trainable parameters within the model, unlike some other approaches such as ATLAS (Izacard et al., 2023).

4 Experimental Framework

In this section, we describe the components of our experimental framework that are shared across all of our experiments.

4.1 RETRO-fitting

While we could train retrieval-augmented language models from scratch, here we instead opt to train models by continued pretraining of a GPT-style base model with RETRO-style augmented retrieval. We refer to this process as RETRO-*fitting*. With the rise of strong open-source foundation models, RETRO-fitting is more realistic for real-world applications than full pretraining. It also enables us to conduct more experiments, as it significantly reduces training time. Moreover, Borgeaud et al. (2022) show that RETRO-fitted models can achieve a perplexity and downstream performance that is comparable to that achieved with full training.

Technically, RETRO-fitting entails expanding the base model with two new types of layers:

1. an encoder for the retrieved context chunks (neighbors and their continuations); and
2. cross-attention between the retrieved context and the standard language modeling context.

These layers are randomly initialized and trained alongside the rest of the GPT-initialized weights.

4.2 Models

For all our experiments, we RETRO-fit a 345M parameter GPT model pretrained by Nvidia (Shoeybi et al., 2019). This model has 24 transformer layers, each with a hidden size of 1,024 and 16 attention heads, similar to the GPT-2 medium model (Radford et al., 2019). We chose to go with a relatively small base model because we want to specifically explore the potential of offloading information to the retrieval mechanism rather than storing it in the model parameters, which is a core motivation behind retrieval-augmented pretraining. Indeed, previous work on downstream question answering tasks has shown that RETRO sees the more benefit from retrieval the fewer parameters it has to store information in (Wang et al., 2024). Also, small models allow us to do more experiments given a fixed computational budget, and RETRO shows similar perplexity curves regardless of model size (Borgeaud et al., 2022).

4.3 Retrieval

For retrieval, we use the training set of the Pile (Gao et al., 2020), which comprises about 800 GB of text of different genres. We embed this data using mean pooling over representations from `MiniLM-L6-H384-uncased` (Wang et al., 2020). This model is one of the top performers at sentence embedding according to measurements on Sentence Transformers (Reimers and Gurevych, 2019). To perform indexing and approximate search, we use FAISS (Johnson et al., 2019) with the index configuration `OPQ32_64`, `IVF65536_HNSW8`, `PQ32` to enable efficient and scalable approximate nearest neighbor search. This configuration first applies Optimized Product Quantization (OPQ) to rotate and transform embeddings for better quantization, followed by an Inverted File Index (IVF) with 65,536 clusters, where the coarse quantizer is accelerated using a Hierarchical Navigable Small World (HNSW) graph. Finally, Product Quantization (PQ) with 32 subquantizers is used to compress vectors for fast and memory-efficient similarity search. We always feed our models the top $k = 2$ retrieved neighbors, both during training and testing.

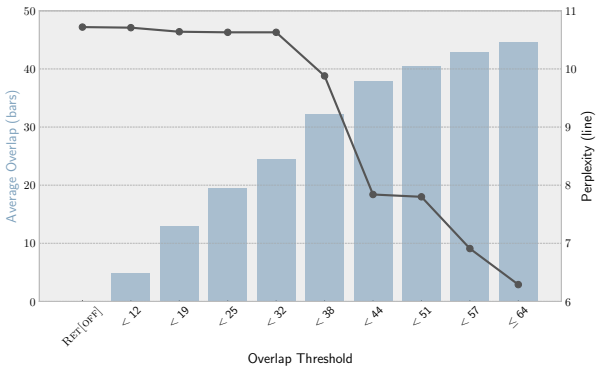

Figure 1: Test perplexity (line) at training step 4,000 and average overlap in terms of number of tokens (bars) for different overlap thresholds during training.

5 Impact of Overlap on Perplexity

As already mentioned, recent work on retrieval-augmented language modeling shows the importance of surface-level overlap between the input chunk and its neighbors in training RETRO (Borgeaud et al., 2022; Norlund et al., 2023; Doostmohammadi et al., 2023). In our first set of experiments, we want to dive deeper and see how different degrees of overlap affect the training of a RETRO model. To this end, we artificially bound overlap at predefined thresholds.

5.1 Overlap Thresholds

By *overlap*, we mean the number of tokens that are shared between the input chunk and one of its neighbors (including continuations). We divide the full range of possible overlap values (0–64) into 10 equally-sized (up to rounding) intervals $[\min_i, \max_i]$ $(1 \leq i \leq 10)$ and train separate models M_i where we only use neighbors with up to $\max_i$ tokens of overlap (e.g., < 25).

During training, for every query chunk, we initially retrieve 20 neighbors and then filter based on the model-specific overlap threshold, prioritizing neighbors with higher overlap. This approach ensures that we retain only naturally occurring neighbors—a subset of the 20 originally retrieved. If the number of neighbors with an overlap below the model-specific threshold is less than k (the number of neighbors provided to the model), we substitute the missing neighbors with zero vectors. We also consider an extreme setting where we provide no neighbors at all, i.e., we replace all neighbors with zeros. We refer to this setting as RET[OFF]. Note that this differs from a GPT model in that it still uses the additional RETRO parameters.

Note that, during training, while the neighbors are filtered by overlap, the input chunks are the same across all experiments. At test time, we always use the naturally retrieved neighbors, without any overlap thresholding; the input chunks will vary based on the previously generated tokens.

5.2 Experimental Setup

We train each overlap-thresholded model by RETRO-fitting the GPT model described in Section 4.2 on the Pile (Gao et al., 2020). We use the entire training set for retrieval, but only train on a maximum of 10,000 steps with a batch size of 128, which corresponds to approximately 54% of the full data. In our training setup, we follow Wang et al. (2024) by using the Adam optimizer (Kingma and Ba, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and a cosine learning rate decay schedule, starting with a maximum learning rate of 2.5e-4, a minimum of 2.5e-5, and a linear warmup phase spanning the first 5,000 samples.

5.3 Results and Analysis

Figure 1 shows the test perplexity and average overlap for each overlap threshold at training step 4,000. We choose this step here because Borgeaud et al. (2022) report that it is where RETRO converges; we show the results for other training steps later. Looking at the plot, we see that, as the threshold increases, the perplexity remains roughly constant up to < 32. However, at the next threshold level, we see a clear drop, and perplexity decreases rapidly as the overlap increases further. Overall, our results show a strong negative correlation between the test perplexity and the maximal overlap during training.

In Figure 2, we look at perplexity over training steps. We see that after the threshold < 32, all models can reach more or less the same low perplexity given enough time, but the number of steps required for this varies significantly. The model for threshold < 38 takes noticeably longer to converge to a similar perplexity than the models with higher maximal overlap, but the difference among thresholds from < 51 to ≤ 64 is relatively small. In summary, we find that, while a minimum amount of overlap is needed to "activate" a RETRO model, as also discussed in Borgeaud et al. (2022), increasing the overlap further leads to faster convergence. The average overlap for the activated models in this experiment is about 30 tokens.

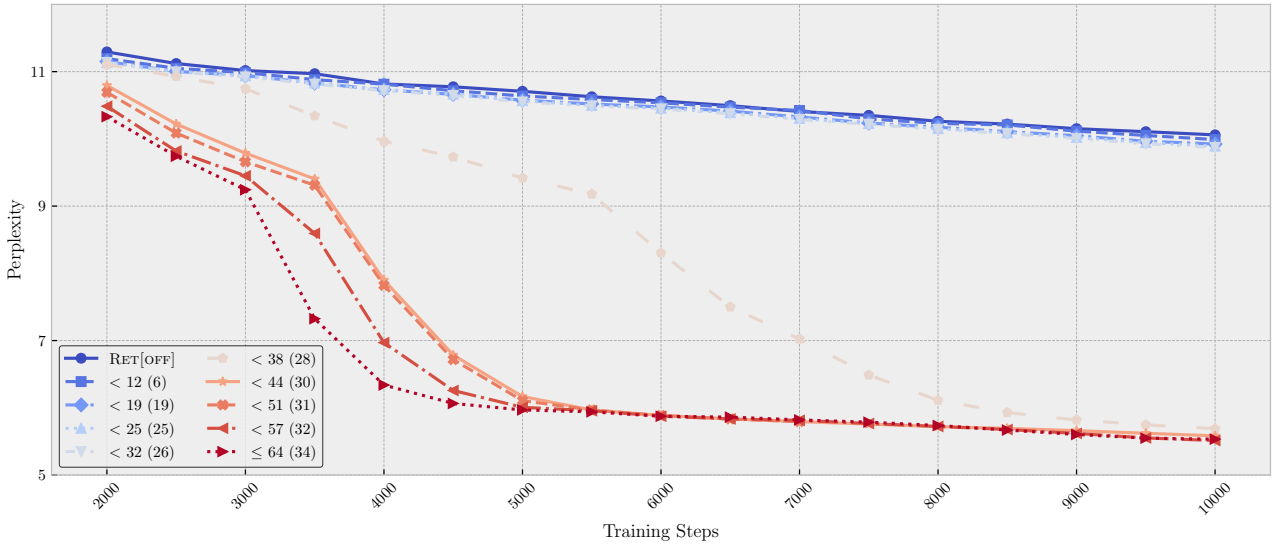

Figure 2: Test perplexity trends for models with different overlap thresholds over training steps. The colors, as well as the line and dot styles, represent the threshold, with colors going from cold to warm as the threshold increases. The legend shows the threshold value, followed by the average overlap for that experiment in parentheses.

6 Overlap and Downstream Tasks

While we have seen that increased overlap reduces the data requirements for RETRO-fitting when performance is measured in terms of perplexity, a separate question is whether this benefit carries over to downstream tasks. To validate this, we apply our overlap-thresholded models to a short-answer generative question answering task.

6.1 Experimental Setup

To improve our RETRO-fitted models' abilities to follow instructions and generate coherent responses, we first instruction-tune them on a blend of open-source datasets provided by Megatron (2023), including Dolly (Conover et al., 2023) and Unnatural Instructions (Honovich et al., 2023). We fine-tune the models for 1,000 steps with a batch size of 128. Following Wang et al. (2024), we compute the loss only on the answer portion of each question–answer pair and update the weights with a learning rate of 5e-6 and a weight decay of $\lambda = 0.01$. For optimization, we again use Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.98$.

While RETRO shares its training objective with GPT models, it requires retrieval of nearest neighbors, which many instruction tuning datasets lack. To avoid using noisy neighbors from the pretraining corpus, we disable the RETRO context encoder using a manually set gate that skips the cross-attention when retrieval is unavailable. This freezes the encoder parameters and updates only the decoder, simplifying tuning and enabling inference both with and without retrieval.

Following the literature, we evaluate our models on the Natural Questions (Kwiatkowski et al., 2019) dataset. We report exact match scores, which means that apart from punctuation marks and whitespace, the model response should exactly match one of the gold answers. Each question in the dataset comes with multiple contexts, which we provide to the model as neighbors with $k = 2$. For generation, we use greedy decoding.

6.2 Results and Analysis

Figure 3 shows the exact match scores of selected models over training steps. To reduce the number of experiments, we exclude certain models that were not activated in our previous experiments, as all such models had a similar (low) performance.

In general, the results on the downstream task follow a similar pattern to our earlier results on perplexity: unactivated models consistently underperform compared to activated models, which we attribute to their failure to make optimal use of the provided neighbors. In contrast, activated models benefit from additional training, showing slightly improved exact match scores up to around step 7,000, after which their performance plateaus. Model < 38 initially lags behind—mirroring its per-

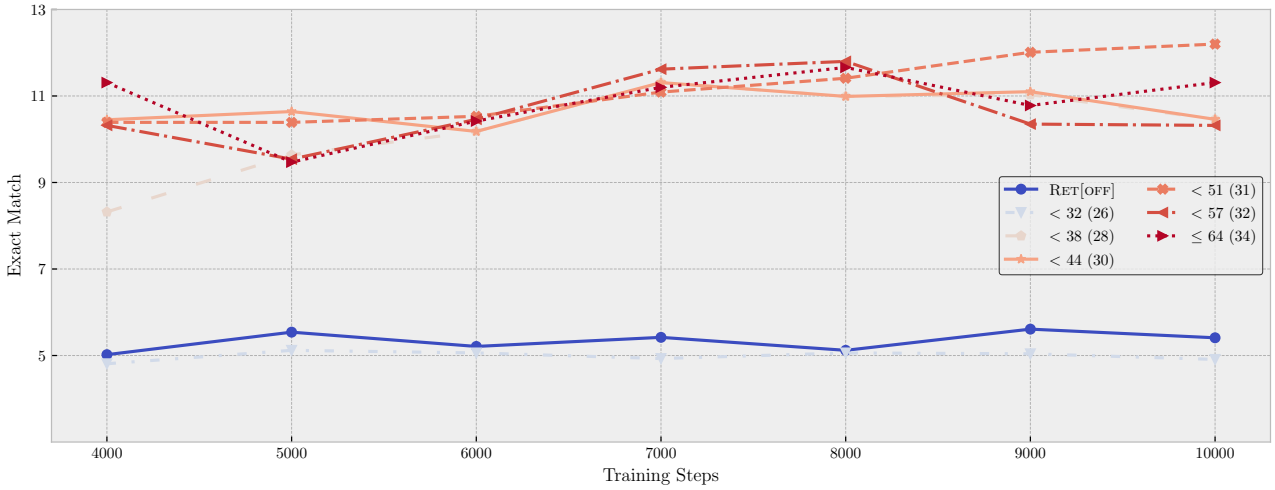

Figure 3: Exact match percentages on the Natural Questions dataset over various training steps and overlap thresholds. The legend shows each threshold value, followed by its corresponding average overlap for that threshold, shown in parentheses.

plexity trend—but eventually catches up with the other activated models. The unactivated models, on the other hand, show a relatively steady (low) performance over different training steps. In summary, these results validate our earlier perplexity-based findings, showing that perplexity can be used as a predictor of downstream task performance.

7 A Low-Resource Scenario

As shown in Figure 2, activating a RETRO model requires about 4,000 training steps at a batch size of 128, which equals about half a million training samples with highly overlapping neighbors. This level of data demand, which comes in addition to the data required to pretrain the base model, is often impractical in real-world scenarios—particularly for low-resource languages or specialized domains where such large datasets are unavailable.

To address this limitation, in this section we explore an approach to activating RETRO models by leveraging synthetic data. Building on our findings regarding the importance of overlap, we propose a methodology that allows us to modulate the strength of the retrieval signal using paraphrased neighbors and thereby control the speed at which a RETRO model gets activated.

7.1 Experimental Setup

Our experiments use the same settings and hyperparameters as described in Section 5.2 for RETRO-fitting and Section 6.1 for instruction tuning and evaluation, with one key difference: we randomly replace one of four chunks in the retrieval-augmented context ($k = 2$ neighbors and their continuations) with a paraphrase of the input chunk.

To get these paraphrases, we use the LLaMA 3 8B instruction-tuned model (AI@Meta, 2024) with the prompt provided in Appendix A. The prompt is designed to maintain enough surface-level similarity between the input chunk and the paraphrase to get a significant overlap, while at the same time ensuring that the paraphrased chunks are not so similar that the model becomes overly reliant on the newly introduced artificial neighbors. We then repeat the experiments from the previous sections using the same thresholds as before, but now with the synthesized neighbors added.

7.2 Results and Analysis

Impact on perplexity Figure 4 shows the test perplexity of the overlap-thresholded models over the training steps with paraphrased neighbors. Paraphrasing, in practice, increases the average overlap per threshold bin. For consistency, we still retain the original threshold labels and report the new average overlaps as a sum of the old average threshold and the increase introduced through paraphrasing.

Compared to the results before the intervention (Figure 2), models that we previously classified as activated exhibit even faster activation with synthetic neighbors added. Their convergence curves are also steeper and approach their mini-

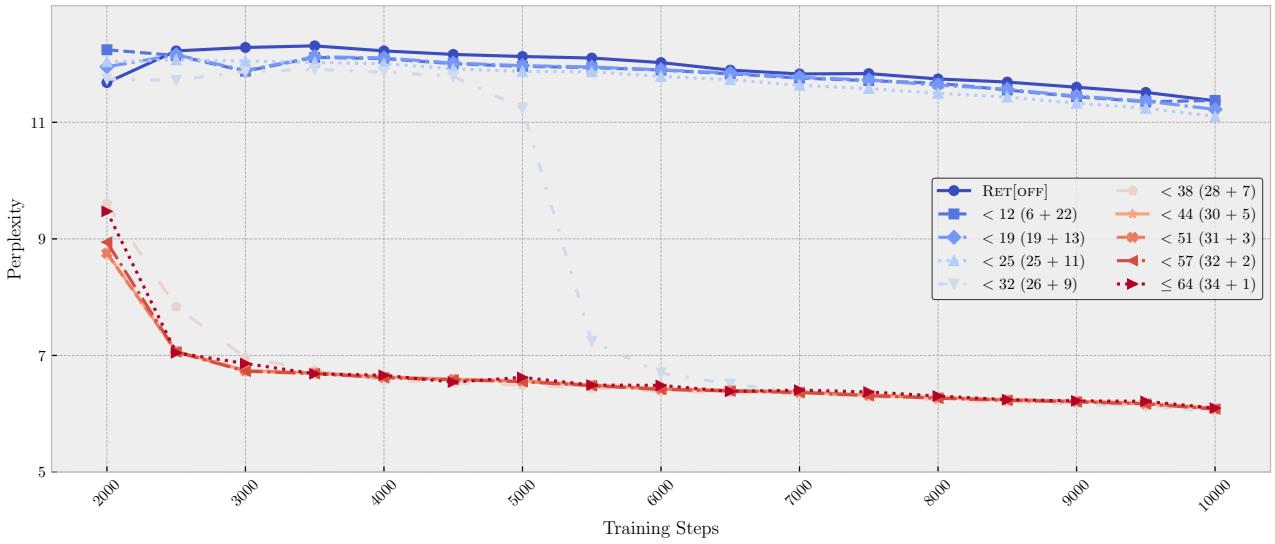

Figure 4: Test perplexity trends of models with varying overlap thresholds over training steps. The legend indicates the threshold value, followed by the average overlap for that threshold and the additional overlap introduced by paraphrases in parentheses.

mum around step 3,000, compared to step 5,000 in the previous setting. This improvement corresponds to roughly 40% less data to reach optimal performance. Model < 32, which was not previously activated, eventually reaches the same levels of perplexity as the models with activated retrieval, although it takes approximately 5,500 steps for this to occur. The price we pay for the faster convergence is higher overall perplexity: the lowest perplexity achieved by the models is now 6.1, compared to 5.6 previously. A plausible explanation for this observation is that paraphrases introduce noise and decrease data variability. The detrimental effect on perplexity is more pronounced in models where retrieval is not activated; these do not reach a perplexity below 11, compared to 10 in the setting without synthetic neighbors.

Looking at the average overlap statistics, we see that values increase substantially at lower thresholds, where the paraphrases add a lot of new overlap, while the effect on the higher-threshold models is significantly smaller. At the same time, convergence happens more quickly for these models regardless, which suggests that factors beyond simple overlap contribute to activating the model's retrieval weights.

Impact on downstream tasks Finally, we turn to the downstream results on the Natural Questions dataset, reported in Table 1. We observe a similar trend as in the setup without paraphrases (Figure 3). To reduce the number of experiments, we focus on the most relevant thresholds—those near the activation point and the extreme case of ≤ 64, to assess whether the addition of paraphrases negatively impacts the model under extreme conditions. The performance approximately doubles once retrieval is activated and remains relatively stable across different training steps. Overall, while there are some minor fluctuations, the models trained with paraphrased neighbors perform on par with models trained without. This result demonstrates that, while perplexity is affected negatively, modulating the overlap signal through paraphrasing can lead to faster activation of the augmented retrieval mechanism without degrading downstream performance.

Thresholds	Training Steps			
	4000	5000	6000	7000
< 25	5.5	5.4	4.9	5.3
< 32	6.1	9.8	10.8	11.1
< 38	11.3	10.1	11.2	11.4
< 44	10.3	11.3	11.5	11.0
≤ 64	9.8	11.4	10.5	10.7

Table 1: Exact match percentages for selected models on the Natural Questions dataset over different training steps. Threshold names and markers match those used in previous plots for ease of comparison.

8 Final Remarks and Discussions

In this paper, we used a pretrained model, which likely contributed to more stable and robust results and enabled us to conduct more extensive experiments. Moreover, RETRO-fitting presents a practical and accessible method for users to integrate retrieval mechanisms into language models, given that these models are already pretrained and training from scratch is costly and resource-intensive. However, it remains unclear whether these results would extend to pretraining a RETRO model from random initialization.

Our experiments were conducted using a single model size. While RETRO tends to show greater benefits on smaller models, since they are more constrained in their capacity and must rely more heavily on retrieved neighbors, studying how model size interacts with retrieval-based methods could provide a more comprehensive understanding of their scalability and efficiency.

In our experiments, we only use paraphrasing of the input. However, in low-resource settings, this approach may be limited, as high-quality paraphrasing models might not be available, or off-the-shelf LLMs may struggle to produce fluent paraphrases in the target language. Therefore, other methods, such as back-translation (Sugiyama and Yoshinaga, 2019) or synonym substitution (Jungiewicz and Smywiński-Pohl, 2019), remain to be explored to determine whether they can similarly reduce perplexity without breaking the model.

Although we significantly increased the overlap between neighbors and the input, we were unable to break the model, suggesting that it has a surprisingly high tolerance for redundancy or dependence on neighboring context. However, it is plausible that beyond a certain point, excessive overlap could lead the model to become overly reliant on its neighbors. This dependence may, in turn, cause performance to degrade when such neighbors are absent or differ at test time.

Our results indicate that unigram overlap between the input and its neighbors serves as a useful heuristic and a simple proxy for understanding what drives the model to attend to the neighbors. However, as shown in the results in Section 7, it is clearly not the full story. Other factors, such as the stronger signal provided by synthetic data compared to natural language (Edunov et al., 2018), or variations in word order (Norlund et al., 2023), may also influence the model's behavior.

9 Conclusions and Future Work

Retrieval-augmented language models have been shown to significantly reduce test-time perplexity, despite their much smaller size compared to standard language models. Prior work has identified the primary driver of this improvement to be the overlap between the input text and its retrieved neighbors during both training and testing (Borgeaud et al., 2022). Initially, this has no effect, but beyond a certain point, the overlap becomes strong enough to tip the model towards using the retrieved neighbors.

We extend this analysis to different time steps, revealing that more overlap accelerates the model's learning and makes it more likely to attend to retrieval. This suggests that a strong overlap signal between the neighbors and the input chunk is crucial for efficient learning. Additionally, we run experiments on a downstream question-answering task to show that these effects extend beyond just perplexity.

We further extend this by replacing one neighbor with a paraphrase of the input chunk to ensure the model always has a relevant, highly overlapping neighbor. This approach significantly enhances the model's data efficiency, reducing training time by approximately 40%. We then evaluate the models on a question-answering task to demonstrate that this type of training does not negatively impact model performance. While it is conceivable that further increasing overlap could eventually harm downstream performance or perplexity, we have not observed this in our experiments.

Future research could investigate pretraining a RETRO model from scratch to better understand the challenges and the role of overlap without the stabilizing effects of prior training. Extending the analysis to larger model sizes would also be valuable, as it could reveal how retrieval-based methods scale and whether their benefits persist across capacities.

Alternative augmentation methods such as back-translation or synonym substitution should be explored, especially in low-resource settings where paraphrasing is less viable. Additionally, determining the threshold at which increased input-neighbor overlap causes model failure remains an open question. Finally, further analysis is needed to uncover other factors beyond overlap that influence a model's attention to its retrieved neighbors.

Limitations

While our findings offer valuable insights into the role of input-neighbor overlap in retrieval-augmented language models, several limitations remain. First, our experiments rely on a pretrained language model, leaving open the question of how overlap affects models trained from scratch, where learning dynamics may differ. Second, we only evaluated a single model size; larger models may exhibit different behaviors with respect to retrieval dependence and overlap sensitivity. Third, although we increased overlap extensively without observing model degradation, we did not determine the point at which excessive overlap may begin to harm performance. Finally, while overlap is a convenient and intuitive metric, it likely does not capture the full complexity of retrieval utility—factors such as word order or semantic similarity in non-overlapping tokens warrant further investigation.

References

AI@Meta. 2024. Llama 3 model card.

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *International Conference on Learning Representations*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

M. Conover, M. Hayes, A. Mathur, J. Xie, J. Wan, S. Shah, A. Ghodsi, P. Wendell, M. Zaharia, and R. Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm. Technical report, Databricks.

Ehsan Doostmohammadi, Tobias Norlund, Marco Kuhlmann, and Richard Johansson. 2023. Surface-based retrieval reduces perplexity of retrieval-augmented language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 521–529, Toronto, Canada. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428, Toronto, Canada. Association for Computational Linguistics.

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*, 24(251):1–43.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Michał Jungiewicz and Aleksander Smywiński-Pohl. 2019. Towards textual data augmentation for neural networks: synonyms and maximum loss. *Computer Science*, 20.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Megatron. 2023. Nvidia Megatron-LM: tools/retro (commit 47e3bd3). https://github.com/NVIDIA/Megatron-LM/tree/47e3bd3047aafbae361e1699d1d8785d678732ca/tools/retro. Accessed: 2025-05-05.

Tobias Norlund, Ehsan Doostmohammadi, Richard Johansson, and Marco Kuhlmann. 2023. On the generalization ability of retrieval-enhanced transformers. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1485–1493, Dubrovnik, Croatia. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners.

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

David Samuel, Lucas Charpentier, and Sondre Wold. 2024. More room for language: Investigating the effect of retrieval on language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 282–305, Mexico City, Mexico. Association for Computational Linguistics.

Weijia Shi, Sewon Min, Maria Lomeli, Chunting Zhou, Margaret Li, Xi Victoria Lin, Noah A. Smith, Luke Zettlemoyer, Wen tau Yih, and Mike Lewis. 2024a. In-context pretraining: Language modeling beyond document boundaries. In *The Twelfth International Conference on Learning Representations*.

Yucheng Shi, Qiaoyu Tan, Xuansheng Wu, Shaochen Zhong, Kaixiong Zhou, and Ninghao Liu. 2024b. Retrieval-enhanced knowledge editing in language models for multi-hop question answering. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 2056–2066, New York, NY, USA. Association for Computing Machinery.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Amane Sugiyama and Naoki Yoshinaga. 2019. Data augmentation using back-translation for context-aware neural machine translation. In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 35–44, Hong Kong, China. Association for Computational Linguistics.

Boxin Wang, Wei Ping, Lawrence McAfee, Peng Xu, Bo Li, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Instructretro: instruction tuning post retrieval-augmented pretraining. In *Proceedings of the 41st International Conference on Machine Learning*, pages 51255–51272.

Boxin Wang, Wei Ping, Peng Xu, Lawrence McAfee, Zihan Liu, Mohammad Shoeybi, Yi Dong, Oleksii Kuchaiev, Bo Li, Chaowei Xiao, Anima Anandkumar, and Bryan Catanzaro. 2023a. Shall we pretrain autoregressive language models with retrieval? a comprehensive study.

Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, et al. 2023b. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *arXiv preprint arXiv:2310.07521*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.

Frank F. Xu, Uri Alon, and Graham Neubig. 2023. Why do nearest neighbor language models work? In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021a. Adaptive semiparametric language models. *Transactions of the Association for Computational Linguistics*, 9:362–373.

Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021b. Adaptive semiparametric language models. *Transactions of the Association for Computational Linguistics*, 9:362–373.

A Paraphrasing Prompt Template

We use the following prompt in our experiments for paraphrasing text chunks:

```
Paraphrase the following text.
- Keep the meaning and the overall structure the same, but you can change the words.
- The paraphrase should have the same length as the input.
- Strictly keep the order of the information intact.
- DO NOT add and DO NOT remove any information.
- Only generate a JSON like {{"paraphrase": THE_PARAPHRASED_TEXT_HERE}} and nothing more:\n\n{chunk}
```