Quest: Query-centric Data Synthesis Approach for Long-context Scaling of Large Language Model

Anonymous ACL submission

Abstract

Large language models, initially pre-trained with a limited context length, can better handle longer texts by continuing training on a corpus with extended contexts. However, obtaining effective long-context data is challenging due to the scarcity and uneven distribution of long documents across different domains. To address this issue, we propose a Query-centric data synthesis method, abbreviated as Quest. Quest is an interpretable method based on the observation that documents retrieved by similar queries are relevant but low-redundant, thus well-suited for synthesizing long-context data. The method is also scalable and capable of constructing large amounts of long-context data. Using Quest, we synthesize a long-context dataset up to 128k context length, significantly outperforming other data synthesis methods on multiple long-context benchmark datasets. In addition, we further verify that the Quest method is predictable through scaling law experiments, making it a reliable solution for advancing long-context models.

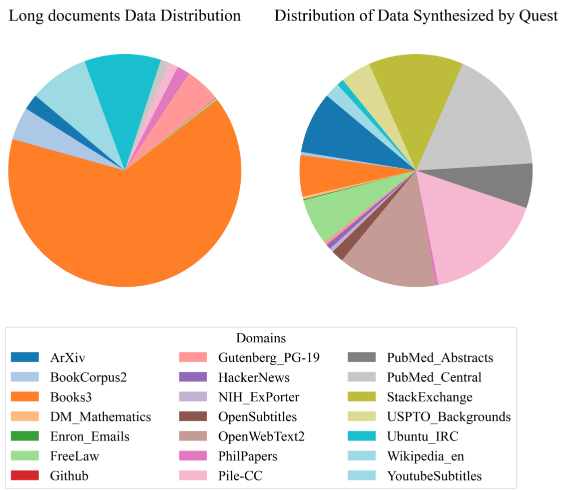

Figure 1: (Left) Distribution of long documents up to 128k in Pile. (Right) Distribution of 128k long-context data synthesized by Quest.

1 Introduction

Large Language Models (LLMs) are pre-trained with pre-defined context lengths, and recent advancements have highlighted the importance of extending the context lengths. The LLaMA models, for instance, have increased their context lengths from 2k (LLaMA) to 4k (LLaMA2) and 8k (LLaMA3) (Touvron et al., 2023a,b; Meta, 2024). LLMs with longer context lengths excel in handling complex tasks (Caciularu et al., 2023; Bairi et al., 2023; Mazumder and Liu, 2022). When facing demands for very long contexts, such as 128k, a widely adopted method is to continue training LLMs with long-context data(Roziere et al., 2023; Xiong et al., 2023; Fu et al., 2024).

To obtain long-context data, (Xiong et al., 2023; Fu et al., 2024) have filtered out long documents meeting the target context length, though these documents often derive from a few specific domains. As shown in Figure 1 (left), our analysis of the widely used Pile corpus (Gao et al., 2020) reveals that long documents are mainly concentrated in the Books3 dataset, leading to a skewed distribution that worsens with longer target contexts. Previous studies (Guu et al., 2020; Levine et al., 2021; Shi et al., 2023) have suggested synthesizing long-context data by aggregating semantically similar documents, such as concatenating a document with its top k retrieved documents. However, these methods often result in redundancy due to similar sentences, especially in large-scale corpora, reducing token prediction difficulty and context diversity, thereby weakening long-context modeling effectiveness (see Section 6.2 for analysis). Thus, there's a pressing need for a method to effectively aggregate relevant but low-redundant documents for long-context data synthesis. Additionally, this method must be highly scalable to construct large datasets for continued training.

This paper proposes Quest, a Query-centric

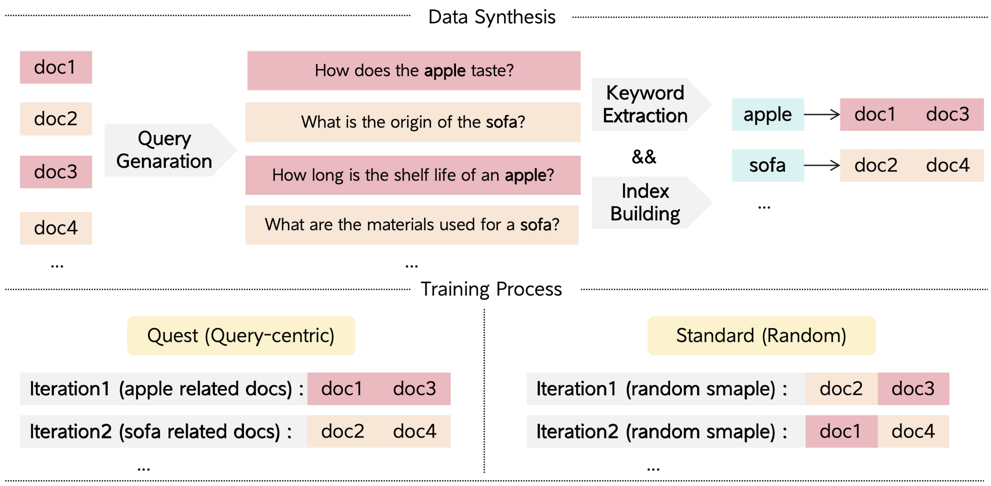

Figure 2: (1) Overview of Query-centric Data Synthesis (Quest) method. (2) Unlike the standard pre-training strategy that randomly shuffled documents in the input context, Quest places relevant documents in the same context.

data synthesis approach, to construct scalable long-context data. We draw our inspiration from the fact that similar queries can aggregate relevant but non-redundant documents via search engines(Mallia et al., 2021; Babenko and Lempitsky, 2014; Kaushik et al., 2004). However, although large amounts of queries can be crawled on the Internet, ensuring the diversity and quality of these queries remains challenging. Thus, we predict potential queries for each document through a generative model. By controlling the generative sampling process, we balance diversity and quality. Specifically, Quest starts by using a lightweight query-prediction model (Raffel et al., 2020; Nogueira et al., 2019; Wu et al., 2022), to predict varied potential queries for each document. Documents sharing the same query are grouped as relevant, mimicking an inverse search process. Quest then clusters similar queries into coarse-grained keywords, akin to topics. Thus documents associated with similar queries are further indexed by the same keywords. Finally, Quest randomly samples from documents indexed by the same keywords and concatenates the selected documents to build long-context data.

The scaling law has been extensively studied in pre-training (Kaplan et al., 2020; Henighan et al., 2020; Hoffmann et al., 2022; Alabdulmohsin et al., 2022; OpenAI, 2023; Bi et al., 2024; Su et al., 2024). However, the scaling law for synthesized long-context data remains unexplored, despite its importance for long-context modeling. Therefore, we further investigate the scaling laws of synthesized long-context data across various model scales using our Quest method. Through accurately modeling and curve-fitting training processes in smaller settings, we expect to predict the training processes on larger datasets.

Through extensive experiments, we show that Quest significantly outperforms other data synthesis methods on multiple long-context benchmarks with context lengths ranging from 32k to 128k. Applying the Quest method to state-of-the-art pretrained model LLaMA3 achieves impressive performance on the widely used Needle-in-a-Haystack task. Furthermore, scaling law experiments confirm the predictability of the Quest method, making it a reliable solution for advancing long-context models.

Our contributions are summarized as follows:

1. We propose a query-centric data synthesis method to alleviate long-context data scarcity and uneven domain distribution.

2. Extensive experiments on 32k and 128k context lengths show that our method outperforms existing approaches.

3. We investigate the scaling law of synthesized long-context data and confirm the predictability of our method.

2 Related Work

Long-Context Language Models The success of LLMs has sparked interest in enabling these models to process longer texts. Some works adapt models for longer texts without additional training by modifying position encoding. For example, Han (Han et al., 2023) and Xiao (Xiao et al., 2023) adjust the attention matrix to generate long contexts, while Jin (Jin et al., 2024) compresses position encoding into the pre-trained range. Other works involve continued training for better performance. Xiong (Xiong et al., 2023) demonstrates that long-context capabilities can be acquired by continually pre-training from short-context models. Chen (Chen et al., 2023b) uses position interpolation to change the distribution of position encoding, and Yen (Yen et al., 2024) proposes context expansion with parallel encoding. Advances have also been made using RoPE (Su et al., 2021), enabling LLMs to handle longer positions. PoSE (Zhu et al., 2023) employs skip-wise position indices, allowing position encoding to adapt to different lengths. However, these approaches often overlook the scarcity and uneven distribution of long text data during continued training, relying on filtering long documents from existing corpora (Xiong et al., 2023; Fu et al., 2024) or randomly splicing short documents to achieve a fixed length (Roziere et al., 2023; Chen et al., 2023c; Tworkowski et al., 2024; Chen et al., 2023a; Li et al., 2023).

Data Synthesis and Augmentation for Long-Context Acquiring effective long-context data for training is challenging. Some previous retrieval-augmented pre-training works (Guu et al., 2020; Levine et al., 2021) can synthesize long-context data. Guu (Guu et al., 2020) clusters semantically similar texts within the same context window, while Levine (Levine et al., 2021) shows that incorporating semantically related but non-adjacent sentences within the same pre-training example enhances sentence representations. Shi (Shi et al., 2023) uses a traveling salesman algorithm to address document redundancy in the kNN method. However, previous data synthesis efforts were limited to context lengths of 8k or less, and the benefits of synthesizing longer data were unclear. This work uses various methods to synthesize texts up to 128k, demonstrating the effectiveness of Quest in synthesizing long-context data.

Scaling Laws For a broad spectrum of factors x, scaling laws (Kaplan et al., 2020; Henighan et al., 2020; Hoffmann et al., 2022) indicate that their impact on the loss L of a pre-trained model follows a power law relationship. Here, x may represent model sizes, quantities of training data, or training steps, with parameters to be determined. Previous research (Alabdulmohsin et al., 2022; OpenAI, 2023; Bi et al., 2024; Su et al., 2024; Xiong et al., 2024) highlights the impressive predictive power of scaling laws. Notably, fitting this relationship to a set of smaller models, training datasets, or computational resources enables precise extrapolation to predict the test loss for much larger cases across several orders of magnitude. This capability allows practitioners to estimate the performance of a pre-trained large language model without incurring the substantial cost of completing extensive training runs.

3 Method

Algorithm 1 Query-centric Data Synthesis (Quest) Method

Require: Dataset $D = \{d_i\}$, Context length L, Split ratio r
Ensure: Training texts with context length L

```
1:  Initialize lists Q and K
2:  for each d_i ∈ D do
3:      Q ← Q ∪ doc2query(d_i)
4:  end for
5:  for each q_i ∈ Q do
6:      K_i ← {k ∈ Rake(q_i) ∧ score(k) ≥ 3.0}
7:      K ← K ∪ {random(K_i)}
8:  end for
9:  I ← {(k_i, d_i) | d_i ∈ D}
10: Sort I by size and split: I_s = {i ∈ I | rank(i) ≤ r × |I|}, I_l = I \ I_s
11: for each training step do
12:     Sample I_k ∈ I_s ∪ I_l (oversample I_s)
13:     T ← concat(sample(I_k)), |T| ≥ L
14:     Train with T
15: end for
```

This section details our proposed Query-centric Data Synthesis (Quest) method. Given a dataset with diverse documents $D = \{d_i\}$, our goal is to effectively aggregate relevant but low-redundant documents for synthesizing training texts with a context length of L. An overview of our approach can refer to Figure 2. Quest mainly includes five steps. First, a query $\{q_i\}$ is predicted for each document $\{d_i\}$ in the corpus. Next, a topic keyword $\{k_i\}$ is extracted from each query. Thirdly, documents with the same keyword are grouped or indexed together. Then, we split the keyword-based inverted indexes according to the number of documents. During training, at each step, we perform sampling without replacement for the documents $\{k_i\}$ within a sampled index. The details of each stage are provided below.

1. **Query Prediction:** We utilize the open-source doc2query model(Nogueira et al., 2019) to predict queries $\{q_i\}$ for each document $\{d_i\}$. For texts that exceed the context length limit of the doc2query model, we segment them into parts and generate a query for each segment. Consequently, for a document $\{d_i\}$, a list of queries $Q_i = \{q_i^1, \ldots, q_i^n\}$ is predicted.

2. **Keyword Extraction:** We extract keywords from each query $\{q_i\}$ with an efficient tool, *Rake*[1]. For texts with multiple queries, *Rake* generates several lists of keywords $K_i = \{k_i^1, \ldots, k_i^n\}$. To ensure the quality of extracted keywords, we adopt two filtering strategies. First, we filter out keywords with a *Rake* score below 3.0. Second, we remove frequent but non-informative keywords such as "following sentence" or "best way" (see Appendix B.2 for details). Then, we randomly select one of the remaining keywords to serve as the representative keyword for the document.

3. **Building a Keyword-based Inverted Index:** We can build a keyword-based inverted index I after we map each document to its representative keyword. Documents with an identical representative keyword are indexed together.

4. **Indexes Split:** We rank the keyword-based inverted indexes in ascending order based on the number of documents within each index and divide the sorted indexes into two sets. The top-ranked *split_ratio%* of the keyword-based inverted indexes are assigned to the short-index set I_s, while the remainder is assigned to the long-index set I_l.

5. **Training Process:** We perform sampling without replacement from the documents within a sampled index and concatenate the selected documents up to the model's context length L for training. We oversample the short indexes to ensure that the number of tokens participating in training is evenly distributed between the short and long indexes.

4 Experiments

In this section, we first introduce the experimental settings (Section 4.1). Then we provide a detailed description of our baseline methods (Section 4.2) and the experimental results (Section 4.3 and Section 4.4).

4.1 Experimental Setup

We conduct continued training on Pythia(Biderman et al., 2023) models of different scales, specifically 1.4B, 6.9B, and 12B. Pythia is a series of models trained on the Pile(Gao et al., 2020) dataset, explicitly designed for research. Experiments conducted with Pythia offer good reproducibility.

We apply the Quest method on Pythia's pre-training data, i.e., the Pile dataset, which does not lead to domain transfer issues. Specifically, we extract 30B tokens of keyword-indexed documents from the 300B tokens of the original Pile dataset.

During training, we use the open-source framework GPT-NeoX[2] with a batch size of 4M tokens for all settings. The AdamW optimizer(Loshchilov and Hutter, 2017) with $\beta_1 = 0.9$ and $\beta_2 = 0.95$ parameters and a cosine learning rate schedule is employed. We also use Flash Attention2(Dao, 2023) and ZeRO(Rajbhandari et al., 2020) to optimize memory and performance. The learning rates are $5e^{-5}$ for the 1.4B model, $4e^{-5}$ for the 6.9B model, and $2e^{-5}$ for the 12B model. For more details, see Appendix B.1.

4.2 Baselines Methods

We compare the proposed Quest method with the previous data synthesis methods:

1. **Standard Method** shuffles and concatenates documents randomly in the input context and has been the standard practice in pre-training (Ouyang et al., 2022; Le Scao et al., 2023; Touvron et al., 2023a).

2. **kNN (Retrieval-augmented Language Model Pre-training)** (Guu et al., 2020; Levine et al., 2021) places each document along with the top k retrieved documents in the same input context.

3. **ICML(Shi et al., 2023) Method** is a recently proposed method that utilizes a traveling salesman algorithm to alleviate the document redundancy problem in the kNN method by ranking similarities and determining the optimal training path.

[1]https://pypi.org/project/rake-nltk

[2]`https://github.com/EleutherAI/gpt-neox`

Train&Test	Model size	Method	Avg.	Sgl.	Multi.	Sum.	Few.	Syn.	Code.
32k	1.4B	Standard	20.94	19.24	17.46	20.65	26.75	2.04	36.41
		KNN	19.97	17.26	13.01	**22.97**	24.16	2.33	39.22
		ICLM	19.82	**20.01**	14.71	21.95	23.09	1.94	35.31
		Quest	**22.06**	17.97	**17.98**	21.91	**28.06**	**2.33**	**42.25**
32k	6.9B	Standard	22.48	18.07	**16.83**	22.33	**30.23**	**3.86**	40.91
		KNN	21.65	18.5	13.64	22.56	28.18	3.76	41.88
		ICLM	20.86	17.82	15.34	22.35	25.86	1.21	41.15
		Quest	**23.23**	**19.21**	14.13	22.45	30.14	2.96	**50.55**
32k	12B	Standard	24.85	22.18	21.94	22.30	**32.05**	**3.78**	43.73
		KNN	22.95	20.55	20.48	23.51	29.19	2.47	37.44
		ICLM	24.07	22.67	**23.29**	23.41	30.99	1.5	37.09
		Quest	**25.24**	**22.34**	21.08	**23.74**	31.91	3.22	**46.8**

Table 1: Experimental results of models with 32k context length. For detailed results, please refer to Appendix C.

For implementing kNN, we utilize a product quantized inverted file (IVFPQ) FAISS index with a code size of 32 and 32,768 corresponding inverted lists. For ICLM, we follow the GitHub repository[3] to synthesize long-context data.

4.3 Evaluation

We evaluate four methods, including Quest and three baseline methods, with evaluation lengths ranging from 32k to 128k. To comprehensively compare Quest with baseline methods, the datasets from different evaluation tasks are divided into two categories: long-text benchmark and short-text benchmark.

1. **Long-text Benchmark**: For 32k context length, we adopt the widely-used Longbench(Bai et al., 2023) Benchmark, testing six task types: Single-document QA, Multi-document QA, Summarization, Few-shot learning, Synthetic, and Code completion, totaling 17 datasets. For 128k context length, we focus on the widely-used Longbook QA task(Zhang et al., 2024), on which the pre-trained models perform reasonably well without instruction tuning(Fu et al., 2024).

2. **Short-text Benchmark**: To assess long-text models on short-text tasks, we select seven widely-used short-text datasets: WinoGrande(Sakaguchi et al., 2021), PIQA(Bisk et al., 2020), Logiqa(Liu et al., 2020), Lambada (OpenAI)(Paperno et al., 2016), HellaSwag(Zellers et al., 2019), ARC-Easy, and ARC-Challenge(Clark et al., 2018).

[3] https://github.com/swj0419/in-context-pre-training

Train&Test	Model size	Method	Longbook QA
128k	1.4B	Standard	9.94
		KNN	10.36
		ICLM	10.70
		Quest	**11.30**
128k	6.9B	Standard	14.47
		KNN	13.38
		ICLM	14.92
		Quest	**17.95**
128k	12B	Standard	17.81
		KNN	16.42
		ICLM	18.44
		Quest	**18.92**

Table 2: Experimental results of models with 128k context length.

Quest achieves the best performance under the 32k context length. Table 1 presents the comparison results on the Longbench, showing that Quest outperforms other methods across various model sizes. The results indicate that KNN and ICLM underperform the Standard method, likely due to grouping textually similar documents, which leads to redundancy in the context. To further analyze the behavior of these methods with a 32k context length, we use TSNE for visualization. Figure 3 (left) demonstrates that the Standard method results in the most dispersed document clusters due to random aggregation. In contrast, KNN and ICLM methods show tightly clustered documents, while Quest exhibits moderate aggregation. This suggests that Quest minimizes document redundancy within the extended context, aligning with its superior performance in the Longbench results.

Quest achieves the best performance under the 128k context length. To further assess the efficacy of the Quest method in extended long-context settings, we extend the context length to 128k and evaluate the trained models on the Longbook QA task. Table 2 shows that Quest consistently outperforms other methods across various model sizes.

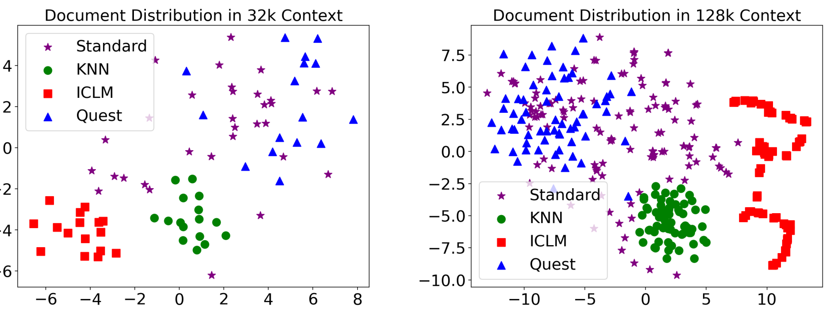

Figure 3: TSNE visualization of aggregated documents from different methods. See Appendix D for more examples.

Figure 3 (right) illustrates the distribution characteristics of documents aggregated by different methods under the 128k context length. Quest continues to aggregate relevant but low-redundant documents within the 128k context. Additionally, ICLM identifies a document-similarity-based pathway that, while not as effective as Quest, also reduces document redundancy and improves model performance compared to KNN. The standard method, which uses randomly sampled documents with no semantic relevance, performs poorly, with this weakness becoming more pronounced at 128k. Overall, the impressive performance of the Quest method under both 32k and 128k context lengths, along with the visualization results, demonstrates Quest's superiority and scalability in synthesizing better long-context data.

Quest retains good performance on short text. To verify how well Quest maintains model performance on short text tasks, we evaluate it on seven commonly reported tasks, as shown in Table 3. Compared with the base model, the performance on short text evaluation remains after continued training with long context data derived from the Quest method. In contrast, other long-text synthesis methods result in varying degrees of degradation in short-text evaluation.

4.4 Performance on the SOTA Model.

To further verify the effectiveness of Quest, we continue to experiment with the current state-of-the-art (SOTA) open-source model LLaMA3(Meta, 2024). We evaluate the LLaMA3-8B post-trained with the Quest method on the widely used Needle-in-a-Haystack task [4]. As shown in Figure 4, our Quest-LLaMA3-8B achieves an accuracy of 97%, significantly exceeding the highest accuracy (88% (Fu et al., 2024)) among the previous non-instruction-tuned models on this task. In addition, we present the results of our model on the Longbook QA task. Table 4 shows that Quest-LLaMA achieves the highest score among open-source models, further narrowing the gap with GPT-4 Turbo under 128K length setting.

[4]https://github.com/gkamradt/LLMTest_NeedleInAHaystack

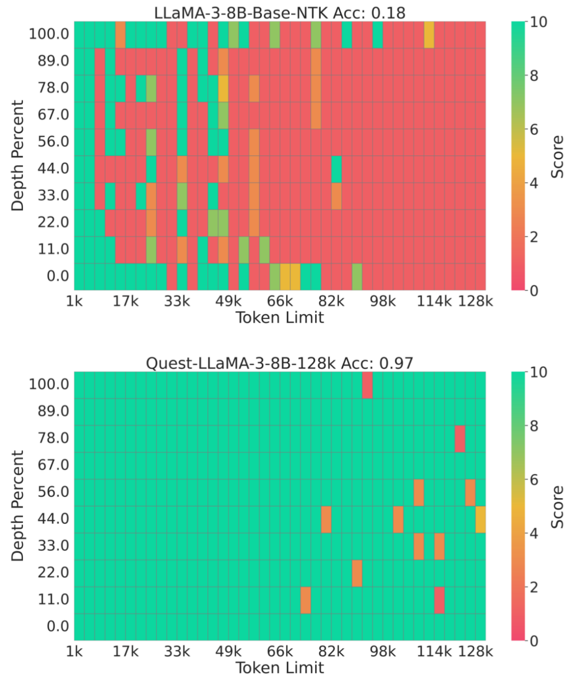

Figure 4: Performance on Needle-in-a-Haystack task. The x-axis depicts the document's length, termed the "haystack," whereas the y-axis illustrates the position of the "needle" (a brief sentence) within the document, spanning from 1K to 128K tokens.

5 Scaling Law of Synthesized Long-context Data

To explore the scaling law of synthesized long-context data, we vary the amount of training data for different model sizes (1.4B, 6.9B, and 12B)

Model	Avg	Win	PIQA	LogiQA	LAMBADA	Hella	ARC-E	ARC-C
Pythia	0.4830	0.5746	**0.7095**	0.2120	0.6163	**0.4042**	**0.6048**	**0.2594**
+ Standard	0.4802	0.5675	0.6975	0.2227	0.6507	0.3943	0.5821	0.2466
+ KNN	0.4769	0.5651	0.7089	0.2028	0.6480	0.3946	0.5737	0.2449
+ ICLM	0.4816	**0.5785**	0.7024	0.2120	**0.6546**	0.3941	0.5753	0.2543
+ Quest	**0.4831**	0.5691	0.7024	**0.2304**	0.6472	0.3961	0.5770	**0.2594**

Table 3: Short text performance comparison of different models on various tasks.

Method / Train Len	Model size	Longbook QA
LLaMA-3-8B 8k(Meta, 2024)♣	8B	13.87
LongLoRA 100k(Chen et al., 2023c)♢	7B	24.30
LongLoRA 64k(Chen et al., 2023c)♢	13B	24.60
YaRN Mistral 128k(Peng et al., 2023)♢	7B	26.30
Yi-9B-200K(AI et al., 2024)♣	9B	30.35
LLaMA-2-7B-80K(Fu et al., 2024)♢	7B	27.40
LLaMA-2-13B-64K(Fu et al., 2024)♢	13B	31.10
GPT-4-Turbo-128k♢	-	**37.40**
Quest-LLaMA-3-8B-128k(ours)	8B	**32.39**

Table 4: Comparisons with the state-of-the-art long-context pre-trained models on the Longbook QA task. ♢: results from (Fu et al., 2024); ♣ results are evaluated by ourselves.

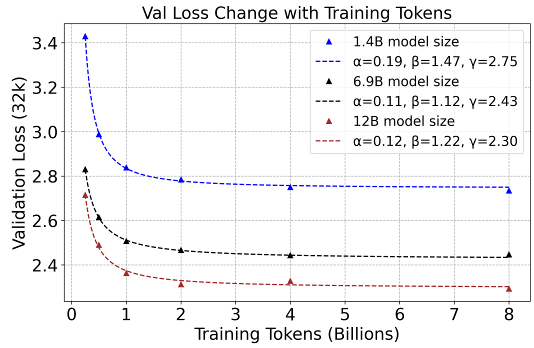

Figure 5: Scaling law of synthesized long-context data under different model sizes.

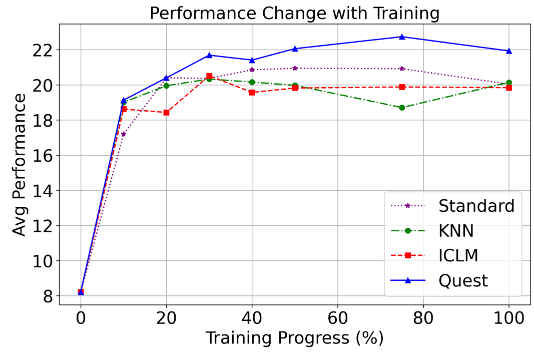

Figure 6: Performance trends during the training progress using data synthesis methods.

under the 32k context length setting. Formally, we formulate the scaling law of the validation loss by studying different model sizes N and dataset sizes D:

$$L(D) = \alpha \exp(-\beta D) + \gamma$$

This formula applies to each model size, where $\{\alpha, \beta, \gamma\}$ are variables to be learned. In our experiments, each model is trained separately on datasets of different sizes: 250 million, 500 million, 1 billion, 2 billion, and 4 billion tokens. Then, we fit a curve for each model size, showing the relationship between the data scaling and the validation loss at the end of each training, as shown in Figure 5.

Furthermore, we verify the correctness of the learned scaling law formula under the data size of 8 billion. For each model size, we compare the relative error between the validation loss at the end of training with 8 billion data and its predicted counterpart via the learned scaling law. The relative error of the 1.4B model is 0.5%, the relative error of the 6.9B model is -0.5%, and the relative error of the 12B model is 0.4%. These negligible errors are a strong demonstration of the scalability and predictability of Quest's data synthesis approach.

6 Analysis

This section provides an in-depth analysis of the Quest method. Considering the high computational cost of LLM experiments, our ablation experiments are performed under a 32k context length, 1.4B model size setting unless otherwise stated.

6.1 Quest's Advantage Gradually Expands with Training Progress

This section studies the performance trends of the training progress using data synthesized by the Quest method. As shown in Figure 6, on the Longbench benchmark, the Quest method consistently outperforms other data synthesis methods from start to finish and exhibits superior evaluation performance. Additionally, the training progress using the Quest method saturates significantly later. In contrast, other methods generally reach performance saturation within the first 40% of the training progress. These two distinct advantages further demonstrate that the Quest method is a superior long-context data synthesis approach compared to the previous methods.

Method	Avg.	Sgl.	Multi.	Sum.	Few.	Syn.	Code.
Long document	21.11	19.77	15.15	22.11	24.81	2.46	41.84
Quest	**22.06**	17.97	17.98	21.91	28.06	2.33	42.25

Table 5: Performance comparison of using long document and Quest synthesized long-context data.

6.2 Quest Balances Document Similarity for Superior Performance

To investigate the impact of document similarity within the same context on performance, we randomly sample contexts derived from different methods and calculate the similarity between documents aggregated into the same context. As shown in Figure 7, we find that under different model size and context length settings, the performance of models shows a trend of first improving and then declining as similarity increases. This indicates that both irrelevant and highly similar document aggregations can lead to performance degradation in long-context modeling.

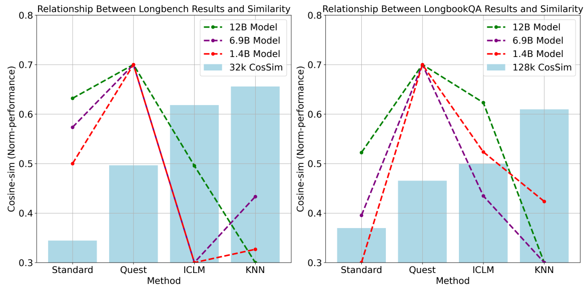

Figure 7: The trend of long-context performance as the similarity of aggregated documents increases. All results are normalized within the similarity range.

6.3 Quest-Synthesized Long-context Outperforms Long Document

Some long documents have already reached the target context length in the pre-training corpus. We compare the performance of using Quest-synthesized long-context data with using only long documents for training. Table 5 shows that using Quest-synthesized long-context data achieves better results on Longbench than using only long documents. Long documents perform worse because they only exist in a few domains, resulting in a skewed data distribution. The Quest method, on the other hand, can cover every domain, resulting in more diverse synthesized long-context data and better performance in evaluation tasks. We also attempt further comparisons with a context length of 128k. However, long documents exceeding 128k in the Pile are rare and inadequate to support a fair comparison experiment. As the target context length increases, the scarcity problem becomes more pronounced, highlighting the importance of effective long-context synthesis methods.

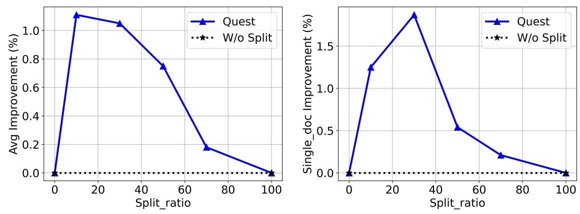

Figure 8: The performance changing trend with *split_ratio* increase. For detailed results, please refer to Appendix C.

6.4 Impact of Split Ratio

This section studies the impact of *split_ratio*, which controls the proportion of oversampled keyword-based inverted indexes. Figure 8 shows the performance changing trend with *split_ratio* increase. Experimental results indicate that overall performance follows a trend of initially growing and then declining with the increase in *split_ratio*. The proportion of oversampled indexes between 10% and 30% yields the best results in average. This shows that appropriate oversampling of the indexes with fewer documents is beneficial to long-context modeling, again illustrating the importance of balanced data distribution for long-context capabilities.

7 Conclusion

In this paper, we introduce Quest, a novel method for synthesizing scalable long-context data by grouping and concatenating relevant but low-redundant documents associated with similar queries. This approach alleviates data scarcity and uneven distribution in long-context data for improving the long-context modeling ability of pre-trained models. Extensive experiments demonstrate that Quest outperforms existing approaches across various long-context benchmarks, proving it to be an effective and reliable solution for advancing long-context models.

8 Limitations

While employing GPT-4(Achiam et al., 2023) for keyword generation could potentially enhance performance due to its proficiency in handling complex tasks, the computational requirements to process large datasets make this option impractical for our purposes. Consequently, we have opted for a more resource-efficient doc2query(Nogueira et al., 2019) model for query prediction and the Rake algorithm for keyword extraction. However, these methods introduce biases that may affect the diversity and quality of the outputs.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

01. AI, :, Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Tao Yu, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. 2024. Yi: Open foundation models by 01.ai. *Preprint*, arXiv:2403.04652.

Ibrahim M Alabdulmohsin, Behnam Neyshabur, and Xiaohua Zhai. 2022. Revisiting neural scaling laws in language and vision. *Advances in Neural Information Processing Systems*, 35:22300–22312.

Artem Babenko and Victor Lempitsky. 2014. The inverted multi-index. *IEEE transactions on pattern analysis and machine intelligence*, 37(6):1247–1260.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Ramakrishna Bairi, Atharv Sonwane, Aditya Kanade, VageeshD C, Arun Iyer, Suresh Parthasarathy, Sriram Rajamani, B. Ashok, and Shashank Shet. 2023. Codeplan: Repository-level coding using llms and planning.

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. 2024. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Avi Caciularu, MatthewE. Peters, Jacob Goldberger, Ido Dagan, and Arman Cohan. 2023. Peek across: Improving multi-document modeling via cross-document question-answering.

Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. 2023a. Clex: Continuous length extrapolation for large language models. *arXiv preprint arXiv:2310.16450*.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023b. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023c. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. 2024. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*.

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 Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Chi Han, Qifan Wang, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. 2023. Lm-infinite: Simple on-the-fly length generalization for large language models. *arXiv preprint arXiv:2308.16137*.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al.

2020. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. 2024. Llm maybe longlm: Self-extend llm context window without tuning. *Preprint*, arXiv:2401.01325.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Raghav Kaushik, Rajasekar Krishnamurthy, Jeffrey F Naughton, and Raghu Ramakrishnan. 2004. On the integration of structure indexes and inverted lists. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 779–790.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2023. Bloom: A 176b-parameter open-access multilingual language model.

Yoav Levine, Noam Wies, Daniel Jannai, Dan Navon, Yedid Hoshen, and Amnon Shashua. 2021. The inductive bias of in-context learning: Rethinking pretraining example design. *arXiv preprint arXiv:2110.04541*.

Shanda Li, Chong You, Guru Guruganesh, Joshua Ainslie, Santiago Ontanon, Manzil Zaheer, Sumit Sanghai, Yiming Yang, Sanjiv Kumar, and Srinadh Bhojanapalli. 2023. Functional interpolation for relative positions improves long context transformers. *arXiv preprint arXiv:2310.04418*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Antonio Mallia, Omar Khattab, Torsten Suel, and Nicola Tonellotto. 2021. Learning passage impacts for inverted indexes. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1723–1727.

Sahisnu Mazumder and Bing Liu. 2022. Lifelong and continual learning dialogue systems.

Meta. 2024. Introducing meta llama 3: The most capable openly available llm to date.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*.

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

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Weijia Shi, Sewon Min, Maria Lomeli, Chunting Zhou, Margaret Li, Victoria Lin, Noah A Smith, Luke Zettlemoyer, Scott Yih, and Mike Lewis. 2023. In-context pretraining: Language modeling beyond document boundaries. *arXiv preprint arXiv:2310.10638*.

Hui Su, Zhi Tian, Xiaoyu Shen, and Xunliang Cai. 2024. Unraveling the mystery of scaling laws: Part i. *arXiv preprint arXiv:2403.06563*.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *Preprint*, arXiv:2104.09864.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. 2024. Focused transformer: Contrastive training for context scaling. *Advances in Neural Information Processing Systems*, 36.

Xing Wu, Guangyuan Ma, Wanhui Qian, Zijia Lin, and Songlin Hu. 2022. Query-as-context pretraining for dense passage retrieval. *arXiv preprint arXiv:2212.09598*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. 2023. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*.

Yizhe Xiong, Xiansheng Chen, Xin Ye, Hui Chen, Zijia Lin, Haoran Lian, Jianwei Niu, and Guiguang Ding. 2024. Temporal scaling law for large language models. *arXiv preprint arXiv:2404.17785*.

Howard Yen, Tianyu Gao, and Danqi Chen. 2024. Long-context language modeling with parallel context encoding. *arXiv preprint arXiv:2402.16617*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, et al. 2024. ∞bench: Extending long context evaluation beyond 100k tokens. *arXiv preprint arXiv:2402.13718*.

Dawei Zhu, Nan Yang, Liang Wang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2023. Pose: Efficient context window extension of llms via positional skip-wise training. *arXiv preprint arXiv:2309.10400*.

A Long-Context Data Distribution Sources

The distribution of native long-context data is uneven. As detailed in Section 1, the distribution of long text sources is highly uneven. To address this imbalance, we employed Quest to synthesize long text data. Figure 9 illustrates the distribution of text sources before and after the application of Quest. The implementation of Quest has markedly increased the volume of data in domains such as ArXiv, FreeLaw, OpenWebText2, Pile-CC, and PhilPapers, where native data was previously minimal or nonexistent.

B Implementation Details of Quest

B.1 Model Configuration

We present the model configuration in Table 6. For the other baselines, we only altered the training dataset, keeping the model configuration unchanged.

B.2 Filtering of Keywords

We used *Rake* for keyword extraction and found many high-frequency but meaningless keywords. Therefore, we maintained a list of stop words and performed a keyword extraction on the generated query to avoid selecting these stop words. Some of the stop words are listed in Table 7. Moreover, to enhance the quality of keyword extraction, we applied a post-processing step to clean the keywords generated by the *Rake* algorithm. This involves removing punctuation and filtering out keywords that are either less than four characters in length or have a score below three. This cleaning process ensures that the extracted keywords are both meaningful and relevant.

C 32K Longbench Results

We report the performance of Longbench on 17 English subtasks. Table 8 and Table 9 are the detailed results of Table 1. Table 10 and Table 11 are the detailed results of Figure 8.

D Examples of similarity visualizations.

We present more visualization results. Figure 10 shows the t-SNE visualization of documents within a 32k context, while Figure 11 illustrates documents within a 128k context. The Standard method's random concatenation of documents results in an overly dispersed distribution, disrupting

	32K			128K		
model size	1.4B	6.9B	12B	1.4B	6.9B	12B
rotary-pct	0.25					
rotary-emb-base	100000			5000000		
β_1	0.9					
β_2	0.95					
eps	$1e^{-8}$					
lr	$5e^{-5}$	$4e^{-5}$	$2e^{-5}$	$5e^{-5}$	$4e^{-5}$	$2e^{-5}$
precision	bfloat16					
Zero_stage	1					
gradient-clipping	1.0					
weight-decay	0.1					
lr-decay-style	cosine					
train-iters	1000					
warmup-iters	200					
seq-length	32768			131072		
GPU-type	H800					
GPU-numbers	16	32	32	32	32	32
training-time	6.3h	14h	20.6h	9.5h	30h	39h

Table 6: Model Training Configuration

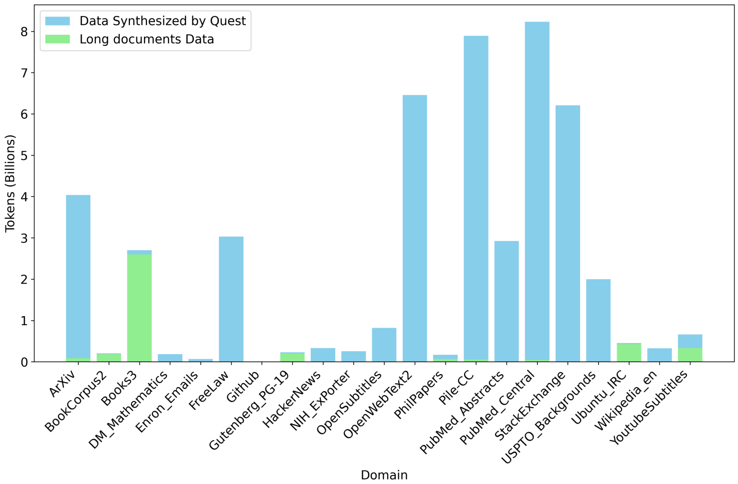

Figure 9: Comparison of the distribution of 128k documents before and after using Quest.

Column 1	Column 2	Column 3
best way	get rid	bad idea
good way	main differences	valid way
following sentence	two sentences	better way
mean	passage mean	following data
good idea	best ways	correct way
sentence mean	next word	following passage
part 1	current state	following equation

Table 7: Stop Keywords

document relationships and leading to poorer performance. In a 32k context, ICLM causes excessive clustering due to shorter similarity-path lengths, mirroring a KNN-like distribution and impairing performance on the Longbench benchmark. However, the 128k context allows ICLM to form longer similarity paths, dispersing document distribution and enhancing performance on the LongbookQA benchmark.

Notably, Quest maintains an evenly dispersed document distribution in both contexts, underscoring its superior performance on both benchmarks. These findings indicate that overly dispersed or concentrated document semantics can harm model performance, while Quest improves performance by clustering query-related documents, ensuring relevance and avoiding redundancy.

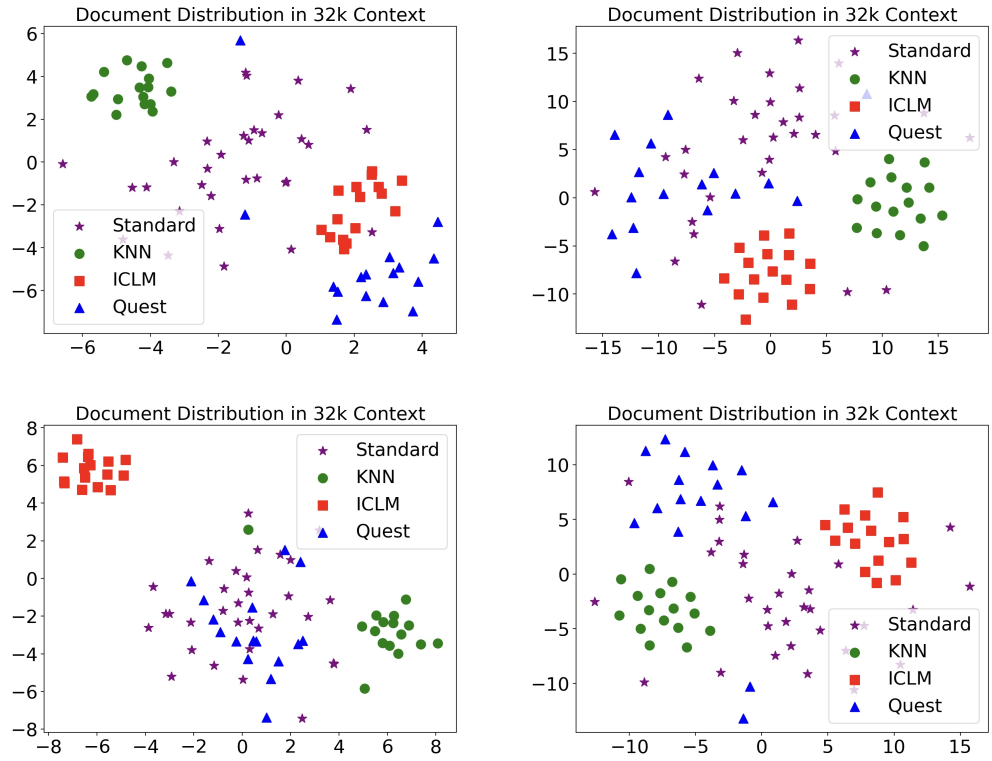

Figure 10: Visualizing Documents Comprising a 32k Context.

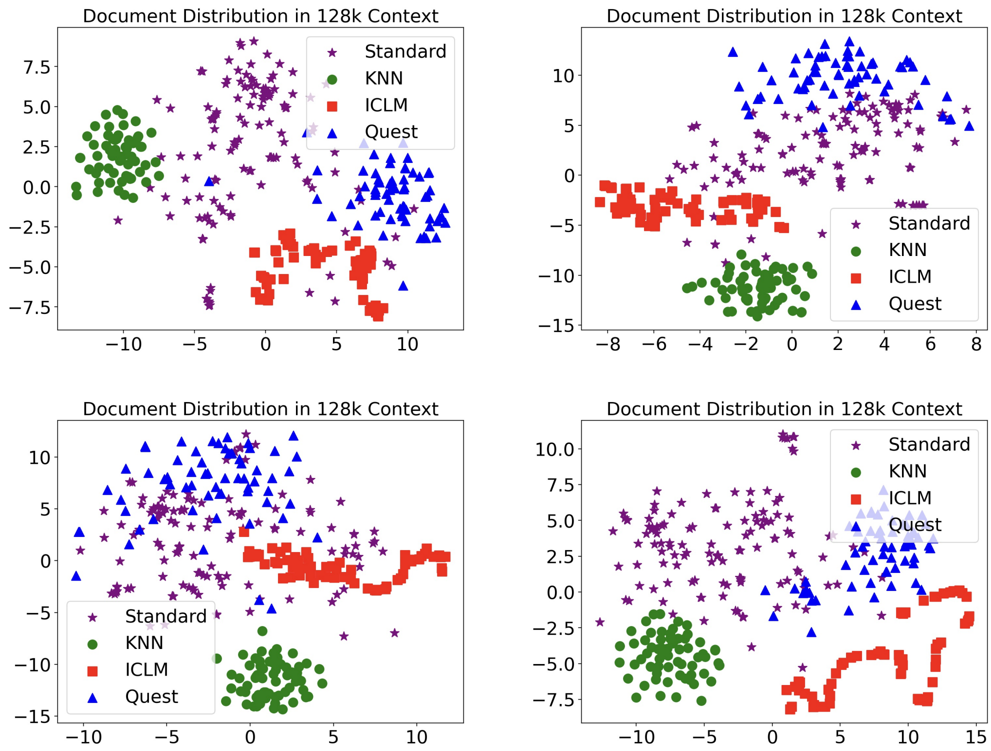

Figure 11: Visualizing Documents Comprising a 128k Context.

Model Size	Method	Few-shot Learning				Synthetic Tasks		Code Completion	
		trec	triviaqa	samsum	nq	passage_count	passage_retrieval_en	lcc	repobench-p
1.4B	Standard	32.29	19.99	24.75	29.95	2.47	1.62	33.66	39.16
	KNN	33.50	17.13	21.88	24.13	1.09	3.58	37.69	40.75
	ICLM	31.75	17.91	16.08	26.60	1.00	2.88	34.56	36.07
	Quest	40.75	18.68	19.17	33.65	0.96	3.71	40.89	43.60
6.9B	Standard	38.75	22.38	23.06	36.72	2.97	4.75	41.17	40.66
	KNN	35.83	24.29	16.54	36.07	2.69	4.83	42.86	40.89
	ICLM	38.08	18.90	15.47	30.97	2.15	0.27	43.37	38.93
	Quest	38.50	21.59	24.22	36.26	2.41	3.50	49.95	51.16
12B	Standard	39.25	26.87	26.97	35.12	3.12	4.44	42.03	45.43
	KNN	39.42	21.64	20.22	35.46	2.11	2.83	36.62	38.25
	ICLM	41.08	23.05	22.24	37.60	1.75	1.25	34.65	39.52
	Quest	38.21	26.60	21.81	41.03	2.86	3.58	46.91	46.69

Table 8: Performance of different methods across various Longbench subtasks.

Model Size	Method	Single-Doc QA			Multi-Doc QA			Summarization		
		narrativeqa	qasper	multifieldqa_en	hotpotqa	2wikimqa	musique	gov_report	qmsum	multi_news
1.4B	Standard	13.55	14.8	29.38	21.71	22.96	7.7	23.28	14.53	24.13
	KNN	13.29	12.24	26.24	16.72	16.33	5.98	26.29	15.31	27.32
	ICLM	14.66	15.79	29.59	16.56	20.13	7.44	25.28	14.33	26.25
	Quest	12	12.77	29.14	21.63	24.7	7.62	25.53	14.35	25.86
6.9B	Standard	13.83	10.78	29.61	21.66	21.52	7.31	24.08	16.66	26.25
	KNN	16.10	10.41	29.00	19.66	17.30	3.97	26.70	17.06	23.91
	ICLM	14.62	10.65	28.19	19.48	20.44	6.11	26.08	16.41	24.56
	Quest	17.77	8.63	31.23	19.46	17.60	5.33	26.69	16.11	24.56
12B	Standard	20.32	13.85	32.36	28.79	22.09	14.94	24.57	18.56	23.78
	KNN	17.91	12.23	31.50	24.79	23.63	13.01	28.43	18.05	24.04
	ICLM	20.33	16.84	30.84	31.92	23.83	14.11	28.01	18.97	23.24
	Quest	19.12	14.17	33.72	27.53	21.76	13.94	28.22	19.33	23.68

Table 9: Performance of different methods across various Longbench subtasks.

Split Ratio (%)	Few-shot Learning				Synthetic Tasks		Code Completion	
	trec	triviaqa	samsum	nq	passage_count	passage_retrieval_en	lcc	repobench-p
0	33.46	21.94	22.50	29.30	1.48	2.50	32.29	37.06
10	39.21	23.39	21.47	30.44	1.07	3.77	33.96	39.11
30	36.75	18.06	27.06	27.85	0.89	3.50	33.74	38.32
50	36.17	22.02	24.29	31.60	1.64	4.69	32.28	37.00
70	33.83	19.17	25.58	27.13	1.81	3.67	29.78	35.66
90	32.33	21.23	26.34	23.98	2.41	2.60	30.81	34.52
100	33.46	21.94	22.50	29.30	1.48	2.50	32.29	37.06

Table 10: Performance Change with Split_Ratio across various Longbench subtasks.

Split Ratio (%)	Single-Doc QA			Multi-Doc QA			Summarization		
	narrativeqa	qasper	multifieldqa_en	hotpotqa	2wikimqa	musique	gov_report	qmsum	multi_news
0	12.41	13.13	27.87	20.05	16.65	8.05	24.57	15.50	22.64
10	13.79	16.22	27.15	18.38	18.74	9.11	26.72	15.01	22.69
30	14.04	17.19	27.77	20.39	20.70	7.87	26.20	14.94	23.99
50	13.31	13.92	27.80	18.69	21.84	7.14	24.70	15.12	23.18
70	11.32	14.34	28.37	17.85	22.21	9.20	25.18	14.89	24.35
90	13.18	18.30	29.68	19.71	21.14	7.14	24.06	15.09	23.51
100	12.41	13.13	27.87	20.05	16.65	8.05	24.57	15.50	22.64

Table 11: Performance Change with Split_Ratio across various Longbench subtasks.