Is Position Bias in Dense Retrievers Built In–or Learned from Data?

Anonymous ACL submission

Abstract

Dense retrievers exhibit positional bias, favoring documents whose query-relevant information appears near the beginning and degrading retrieval performance when the information appears later. While prior work on positional bias in dense retrievers has largely focused on architectural explanations, we study how the positional distribution of evidence in training data affects retrieval-level bias direction. To test this, we construct synthetic position-targeted training sets in which query-relevant evidence appears at the beginning, middle, or end of documents, and fine-tune eight architecturally diverse pretrained models under position-skewed and balanced training distributions. At the ranking level, we observe a strong directional pattern across the examined models: skewed training distributions favor evidence at the corresponding positions. Position-balanced training reduces positional sensitivity by 57–87% on position-aware benchmarks, with competitive mean retrieval performance in our controlled setting. Representation-level analyses further suggest that fine-tuning often reshapes learned positional preferences, although pre-existing architectural or pretraining-specific tendencies persist in some models. These results identify training-position distribution as a major controllable factor in retrieval-level position bias and suggest balanced data curation as a practical mitigation strategy.

1 Introduction

Dense retrievers (Karpukhin et al., 2020; Izacard et al., 2022) now serve as a core component in open-domain question answering and retrieval-augmented generation (Lewis et al., 2020; Jeong et al., 2024). Yet they exhibit a systematic *position bias*. Retrieval performance drops substantially when query-relevant information appears in the middle or end of a document rather than near the beginning (Coelho et al., 2024; Zeng et al., 2025). A retriever that disproportionately favors early positions risks missing critical information, potentially degrading downstream tasks such as retrieval-augmented generation (Fayyaz et al., 2025). Understanding the source of this bias is therefore important to prevent such performance degradation.

Prior work has largely examined position bias empirically: it has been observed across training stages (Coelho et al., 2024), positional encodings (Lee et al., 2025), and pooling-token attention patterns (Schuhmacher et al., 2026). Zeng et al. (2026) further show that positional sensitivity does not correlate with architectural factors. The underlying cause in dense retrievers thus remains unclear. In autoregressive transformers, causal attention has been identified as a primary cause of position bias (Wang et al., 2025; Wu et al., 2025). Yet encoder-based dense retrievers—which lack causal masking—still exhibit strong primacy bias (Coelho et al., 2024; Zeng et al., 2025), indicating that architectural factors alone may not fully explain position bias in dense retrievers.

This raises a fundamental question: to what extent can retrieval-level position bias be changed by the positional distribution of fine-tuning data, beyond tendencies induced by architecture and pretraining? In this work, we hypothesize that training-position distribution is an important factor in shaping retrieval-level position bias in dense retrievers. Two forms of positional skew motivate this hypothesis: in training corpora, texts such as news articles place key information in early positions (Po¨ttker, 2003; Catena et al., 2019), and in retrieval fine-tuning data, such as MS MARCO, query-relevant passages are heavily concentrated in early document positions (Hofstätter et al., 2021; Coelho et al., 2024). Yet no prior work has directly manipulated training data to isolate its role.

To test this hypothesis, we construct position-controlled datasets in which query-relevant information appears at the beginning, middle, or end

of documents, and fine-tune eight architecturally diverse pretrained models—covering encoder and decoder architectures, multiple positional encodings, and different pooling strategies—on them. If models with fundamentally different positional processing nevertheless develop bias patterns that mirror the training distribution, this would suggest that architecture alone cannot fully explain the bias. We evaluate on position-aware benchmarks to measure positional sensitivity and on standard retrieval benchmarks to examine how these training distributions affect performance under conventional evaluation settings.

Our key finding is that retrieval-level position bias direction follows the training data distribution across all eight models, despite their architectural differences: begin-skewed data produces begin-favoring retrieval, mid-skewed data produces mid-favoring retrieval, and end-skewed data produces end-favoring retrieval. Position-balanced training reduces positional sensitivity on position-aware benchmarks while preserving competitive retrieval performance, suggesting that data curation can reduce position bias.

Our contributions are as follows:

- We design a position-controlled data construction pipeline and release the datasets, enabling controlled experiments on the effect of training data on retrieval-level position bias.
- We show that training data distributions shape the direction of retrieval-level position bias, with controlled experiments on eight architecturally diverse models revealing predictable shifts in bias direction.
- We show that position-balanced training reduces positional sensitivity while preserving competitive retrieval performance, suggesting that position bias can be reduced through data curation.

2 Related Work

Position Bias in Dense Retrievers. Dense retrievers exhibit position bias, favoring evidence at the beginning of documents (Fayyaz et al., 2025; Lee et al., 2025; Zeng et al., 2025). Across retriever types, dense embedding and ColBERT-style models show performance degradation due to this bias, while BM25 and cross-encoder rerankers remain robust (Zeng et al., 2025). Zeng et al. (2026) evaluate embedding models on a position-aware benchmark and find that most exhibit primacy bias, though positional sensitivity does not correlate with architectural factors—model size, vector dimension, attention mechanism, or pooling strategy. Similarly, Lee et al. (2025) report that the bias persists across positional encodings—APE, ALiBi, and RoPE. These findings show that position bias is widespread in dense retrievers, but they do not explain its cause.

Architectural Explanations. Prior studies have examined architecture-based explanations for position bias in dense retrievers, but they do not fully explain the observed bias patterns. Schuhmacher et al. (2026) link primacy bias to front-loaded self-attention in pooling-token embeddings of encoder-based models, though its generality across the diverse architectures used in dense retrieval has not been established. In autoregressive transformers, by contrast, Wu et al. (2025) prove that causal attention favors earlier tokens with deeper layers amplifying the effect, and Wang et al. (2025) show that RoPE favors nearby tokens through distance-dependent attention decay. However, encoder-based dense retrievers, which lack causal masking, still exhibit strong primacy bias (Coelho et al., 2024; Zeng et al., 2025), and RoPE based decoder retrievers such as Qwen3-Embedding show primacy rather than recency bias (Zeng et al., 2025, 2026), indicating that architectural factors alone do not fully explain position bias in dense retrievers.

Training Data as a Source of Bias. Training data has also been implicated as a source of position bias. Coelho et al. (2024) show that position bias emerges during unsupervised contrastive pre-training and is amplified by MS MARCO fine-tuning, where relevant passages are disproportionately concentrated in early document positions. Similarly, Fayyaz et al. (2025) find that MS MARCO-trained models exhibit stronger position bias than unsupervised Contriever. Earlier work connects training data to position bias in rerankers: Hofstätter et al. (2021) show that rerankers trained on data with early-skewed answer positions inherit this bias. Across these studies, training data appears as a common factor, yet the evidence comes from observation rather than direct manipulation of the positional distribution. Our work addresses this gap by training eight architecturally diverse models on position-controlled datasets, providing direct evidence that training data distribution drives the direction of position bias in dense retrievers.

3 Method

Our approach has two components: a data construction pipeline that produces position-controlled training datasets (§3.1), and an experimental design that tests how changing the positional distribution of fine-tuning data affects retrieval-level position bias (§3.2).

3.1 Position-Controlled Data Construction

We construct datasets where the location of query-relevant information is controlled through a three-stage pipeline: corpus preparation with length-stratified binning, position-targeted query generation, and multi-reranker position verification.

3.1.1 Corpus Preparation

We use English Wikipedia as our source corpus for its topical diversity and wide range of article lengths. Within each pool, we stratify articles by character count into five length bins (256–512, 512–1024, 1024–2048, 2048–4096, and 4096–8192), using character count rather than token count for tokenizer-agnostic consistency across models. Each document is divided into three equal-length segments—*beginning*, *middle*, and *end*—following Zeng et al. (2025).

3.1.2 Position-Targeted Query Generation

For each document, we generate queries targeting each of the three positional segments using persona-conditioned prompting with GPT-4o-mini[1], following Zhang et al. (2025). A persona is sampled from PersonaHub (Ge et al., 2025) to encourage diverse information needs; the model then generates a query answerable from only the target segment. This yields three query subsets—q_{begin}, q_{mid}, and q_{end}—where the same document appears in all three, each time paired with a different position-targeted query. Details of the generation prompts are provided in Appendix A.

3.1.3 Multi-Reranker Position Verification

The generation prompt asks the LLM to produce a query answerable from the intended target segment, but this constraint is not guaranteed: a generated query may also be answerable from a non-target segment or from multiple segments. To filter such cases, we verify each generated candidate with a panel of three cross-encoder rerankers: *bge-reranker-v2-m3* (Chen et al., 2024), *gte-multilingual-reranker-base* (Zhang et al., 2024), and *jina-reranker-v2-base-multilingual*[2].

We use cross-encoder rerankers, rather than dense retrievers, as verifiers because full-interaction rerankers have been shown to be more robust to evidence position than dense embedding models (Zeng et al., 2025). This reduces the risk that the filtering step itself inherits the position bias that we aim to study.

The verification rule requires unanimous agreement across rerankers. Let q be a generated query for document d, and let $t \in \mathcal{P}$ be its intended target position, where $\mathcal{P} = \{\text{begin}, \text{middle}, \text{end}\}$. We denote the segment at position i by s_i.

For each reranker $R \in \mathcal{R}$, we score each segment as

$$r_{R,i} = R(q, s_i), \quad i \in \mathcal{P}. \tag{1}$$

The candidate is retained only if every reranker scores the target segment at least δ higher than the strongest non-target segment:

$$r_{R,t} - \max_{u \neq t} r_{R,u} \geq \delta, \quad \forall R \in \mathcal{R}. \tag{2}$$

The maximum is taken over the two non-target positions. Thus, even the least favorable reranker must prefer the intended target segment by at least the margin threshold δ.

All main experiments use a margin threshold of $\delta = 0.3$. Appendix B reports filtering statistics under different margin thresholds and an independent segment-wise LLM audit. We refer to the candidates that pass this rule as the retained pool.

3.1.4 Controlled Training Set Sampling

Applying the multi-reranker position-verification rule with margin threshold $\delta = 0.3$ yields 481,236 retained candidate examples for training. Table 1 reports the retained pool by length bin and target position.

The retained pool is not position-balanced, so we do not train on it directly. Instead, we construct the final training sets by downsampling within length-position cells. The smallest retained length-position cell is the middle-position cell in the 4096–8192 length bin, which contains 8,189 examples. This cell determines the sampling budget for the controlled training configurations defined in Section 3.2.

[1] https://developers.openai.com/api/docs/models/gpt-4o-mini

[2] https://huggingface.co/jinaai/jina-reranker-v2-base-multilingual

Length Bin	Begin	Middle	End
256–512	105,652	13,934	21,405
512–1024	86,495	16,660	21,427
1024–2048	60,357	13,594	16,691
2048–4096	43,946	10,527	13,363
4096–8192	39,200	**8,189**	9,796
Total	335,650	62,904	82,682

Table 1: Retained candidate examples by length bin and target position after the multi-reranker position-verification rule with margin threshold $\delta = 0.3$, before downsampling.

This downsampling step ensures that the final training sets use the same number of examples from each length bin, rather than inheriting the uneven length and position counts of the retained pool. As a result, later comparisons are not driven by differences in training size or document length.

3.2 Position-Controlled Experiment Design

Our experimental design tests whether retrieval-level position bias follows the positional distribution of training data across models with different architectural properties.

3.2.1 Model Selection and Initial Tendencies

We select eight pretrained models without retrieval-specific fine-tuning, spanning encoder and decoder architectures, multiple positional encodings, and different pooling strategies. This diversity is central to our design: if models with fundamentally different positional processing develop bias patterns that mirror their training data, the bias cannot be attributed to any single architectural property.

Before retrieval fine-tuning, these models are not perfectly position-neutral at the representation level: encoder models show mild primacy tendencies, while decoder models show recency tendencies (Appendix C). This makes the test stricter: a data-driven effect should appear despite different initial tendencies, and in some configurations must reverse them.

3.2.2 Controlled Training Configurations

Each model is fine-tuned as a dense retriever under four configurations that differ only in the target-position distribution of training queries, expressed as *begin:middle:end* ratios. Three concentrated configurations—100:0:0 (begin; $\mathcal{M}_B$), 0:100:0 (middle; $\mathcal{M}_M$), and 0:0:100 (end; $\mathcal{M}_E$)—restrict all queries to a single target position. The uniform configuration, 33:33:33 ($\mathcal{M}_U$), samples evenly across all three target positions.

Model	Type	PE	Pool	Params	Max Len
BERT-base	Enc	APE	CLS	110M	512
Longformer-b	Enc	APE	Mean	149M	4k
ModernBERT-b	Enc	RoPE	CLS	149M	8k
ModernBERT-l	Enc	RoPE	CLS	395M	8k
GPT-2-medium	Dec	APE	Last-t	355M	1k
BLOOM-560M	Dec	ALiBi	Last-t	560M	2k
TinyLlama-NoPE	Dec	NoPE	Last-t	1.1B	2k
Qwen3-0.6B	Dec	RoPE	Last-t	0.6B	32k

Table 2: Overview of the eight pretrained models used in controlled fine-tuning. PE denotes positional encoding; Pool denotes the document-pooling strategy; Max Len denotes the maximum input length.

All four configurations are sampled from the $\delta = 0.3$ retained pool using the per-bin budget defined in Section 3.1.4. Each concentrated configuration samples 8,189 examples from its target position in each length bin, yielding 40,945 training examples. The uniform configuration randomly samples 2,729 examples from each target position within each length bin, yielding 40,935 training examples. Thus, the configurations are matched in training scale and document-length distribution up to the integer split required by the uniform setting.

This yields 32 training runs: 8 base models $\times$ 4 training configurations. After training, we evaluate each model on position-aware benchmarks to measure how retrieval performance varies across target positions. If bias is data-driven, concentrated configurations should favor their respective target positions, while uniform training should reduce position sensitivity.

4 Experimental Setups

4.1 Base Models

Table 2 lists the eight pretrained base models and their architectural properties. On the encoder side, we include BERT-base (Devlin et al., 2019), ModernBERT-base and ModernBERT-large (Warner et al., 2025), and Longformer-base (Beltagy et al., 2020); on the decoder side, GPT-2-medium (Radford et al., 2019), BLOOM-560M (Workshop et al., 2023), TinyLlama-NoPE (Wang et al., 2024), and Qwen3-0.6B (Yang et al., 2025). ModernBERT-base and large share the same architecture at different scales, enabling a within-architecture scale comparison. TinyLlama-NoPE, which lacks positional encoding, tests whether positional encoding is a necessary condition for bias emergence.

4.2 Training Details

All eight models are fine-tuned as bi-encoder retrievers using InfoNCE loss with chunk-aware negatives: each batch is drawn from a single length bin so that all negatives share the same document length as the positive. We avoid hard negative mining, as mining strategies may introduce position-dependent confounds. All hyperparameters are held constant across the four configurations within each model; the only variable is the positional distribution of training data. Full training details are provided in Appendix D.

4.3 Evaluation

We evaluate all trained models on three position-aware benchmarks: SQuAD-PosQ, FineWeb-PosQ (Zeng et al., 2025), and PosIR (Zeng et al., 2026). Since FineWeb-PosQ and PosIR contain longer passages, we evaluate these benchmarks only on models with sufficient context length: ModernBERT-base, ModernBERT-large, and Qwen3-0.6B. We additionally evaluate on four BEIR datasets (Thakur et al., 2021)—SciFact, HotpotQA, FEVER, and CLIMATE-FEVER—where the provided annotations allow us to identify the position of evidence, enabling analysis of how the training distributions affect performance under standard retrieval settings.

We report nDCG@10 computed separately for each positional subset ($\mathcal{E}_{\text{begin}}$, $\mathcal{E}_{\text{mid}}$, $\mathcal{E}_{\text{end}}$). To summarize position sensitivity as a single scalar, we adopt the Position Sensitivity Index (PSI) proposed by Zeng et al. (2025):

$$\text{PSI} = 1 - \frac{\min(s)}{\max(s)}, \quad \text{where } \max(s) > 0 \quad (3)$$

and $s = \{s_{\text{begin}}, s_{\text{mid}}, s_{\text{end}}\}$ are the metric scores across positional subsets. A PSI of 0 indicates perfect positional robustness; higher values indicate greater sensitivity. We interpret PSI alongside mean performance to ensure that low PSI does not merely reflect uniformly poor retrieval.

5 Experimental Results

Skewed training distributions induce corresponding retrieval-level positional preferences. Figure 1 shows a consistent directional effect: retrieval performance peaks near the position emphasized during fine-tuning. Begin-trained retrievers ($\mathcal{M}_B$) favor early evidence, mid-trained retrievers ($\mathcal{M}_M$) favor middle evidence, and end-trained retrievers ($\mathcal{M}_E$) favor later evidence, consistently across all eight base models. In contrast, uniformly trained retrievers ($\mathcal{M}_U$) do not exhibit a comparable single-position preference; their position-wise curves are flatter, providing an initial indication that balanced training weakens the learned positional shortcut.

Representative cases illustrate the magnitude of this shift in both short- and long-passage position-aware benchmarks. On SQuAD-PosQ, Qwen3-0.6B scores 0.672 in the 0–100 position bucket under begin training but 0.415 under end training; in the 500–3120 bucket, the pattern reverses, with end training scoring 0.702 versus 0.407 for begin training. On FineWeb-PosQ, ModernBERT-large follows the same pattern: when evidence appears at the beginning, the $\mathcal{M}_B$ scores 0.778, compared with 0.475 for the $\mathcal{M}_E$; when evidence appears at the end, the $\mathcal{M}_E$ scores 0.743, compared with 0.447 for the $\mathcal{M}_B$. The pattern also appears in TinyLlama-NoPE, indicating that explicit positional encodings are not required for retrieval-level position bias to emerge.

Overall, these results show that retrieval-level bias direction can be redirected by the positional distribution of fine-tuning data, indicating that architecture alone does not fix the observed bias direction. Appendix E.1 provides an additional mirror-reversal diagnostic that confirms the same directional effect under document reversal.

Position-balanced training reduces sensitivity to answer location. Table 3 shows a consistent pattern across the position-aware benchmarks: the $\mathcal{M}_U$ is the least sensitive to evidence location. It achieves the lowest PSI for all eight models on SQuAD-PosQ and for all three evaluated models on FineWeb-PosQ, indicating that balanced training produces more stable retrieval performance across positions.

On SQuAD-PosQ, $\mathcal{M}_U$ reduces PSI by 57–87% relative to the worst skewed configuration for every model. For example, GPT-2-medium drops from 0.592 under begin training to 0.080 under uniform training, Qwen3-0.6B drops from 0.409 under end training to 0.068, and Longformer-base drops from 0.331 under end training to 0.143. The same pattern holds on FineWeb-PosQ: ModernBERT-base drops from 0.476 to 0.108, ModernBERT-large from 0.426 to 0.116, and Qwen3-0.6B from 0.359 to 0.116.

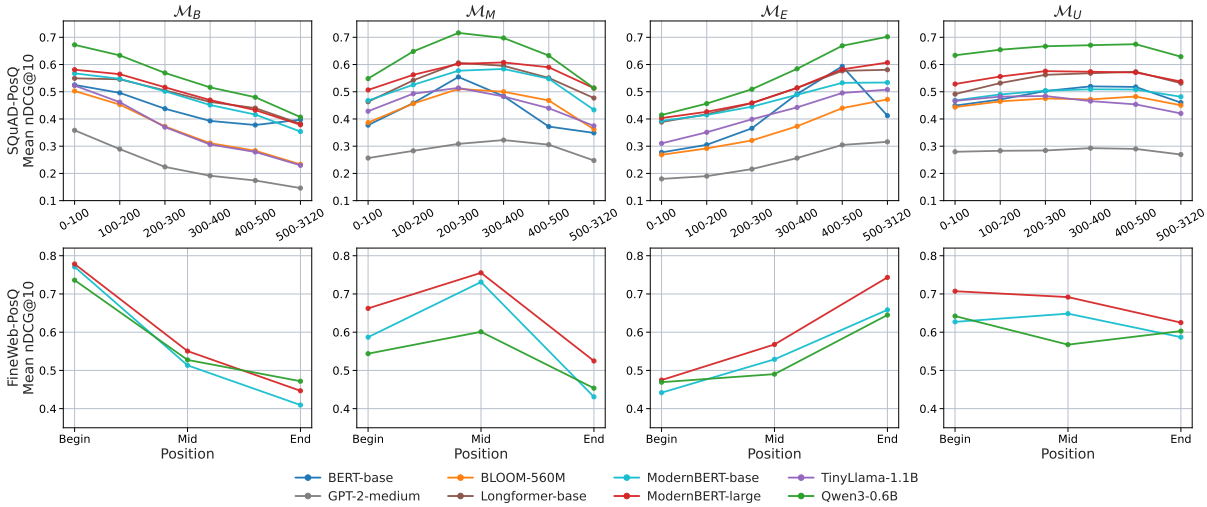

Figure 1: Position-wise nDCG@10 across training configurations. The top row reports SQuAD-PosQ, and the bottom row reports FineWeb-PosQ. Columns correspond to configurations, $\mathcal{M}_B$, $\mathcal{M}_M$, $\mathcal{M}_E$, and $\mathcal{M}_U$; lines denote evaluated base models.

	NDCG@10				PSI ↓			
Model	$\mathcal{M}_B$	$\mathcal{M}_M$	$\mathcal{M}_E$	$\mathcal{M}_U$	$\mathcal{M}_B$	$\mathcal{M}_M$	$\mathcal{M}_E$	$\mathcal{M}_U$
SQuAD-PosQ								
BERT-base	0.438	0.432	0.408	**0.487**	0.281	0.371	0.533	**0.136**
Longformer-base	0.481	0.539	0.489	**0.543**	0.304	0.236	0.331	**0.143**
ModernBERT-base	0.466	**0.520**	0.463	0.516	0.433	0.286	0.341	**0.088**
ModernBERT-large	0.490	**0.564**	0.499	0.557	0.348	0.166	0.335	**0.082**
GPT-2-medium	0.231	**0.287**	0.244	0.283	0.592	0.233	0.431	**0.080**
BLOOM-560M	0.359	0.447	0.361	**0.465**	0.536	0.290	0.431	**0.080**
TinyLlama-NoPE	0.362	0.455	0.418	**0.462**	0.561	0.271	0.389	**0.132**
Qwen3-0.6B	0.546	0.626	0.556	**0.655**	0.395	0.283	0.409	**0.068**
FineWeb-PosQ								
ModernBERT-base	0.554	0.570	0.571	**0.640**	0.476	0.422	0.343	**0.108**
ModernBERT-large	0.592	0.647	0.595	**0.675**	0.426	0.305	0.361	**0.116**
Qwen3-0.6B	0.578	0.533	0.535	**0.604**	0.359	0.245	0.272	**0.116**

Table 3: Mean nDCG@10 and Position Sensitivity Index (PSI) across training configurations. The upper block reports SQuAD-PosQ for all eight models; the lower block reports FineWeb-PosQ for models with sufficient context length. Higher is better for nDCG@10; lower is better for PSI. Best values for each model and metric are in **bold**.

These results show that position-balanced training does not merely move the bias to a different evidence position. Instead, it makes retrieval performance more consistent across positions, so the model is less affected by where the relevant evidence appears.

Position-balanced training reduces sensitivity with competitive retrieval performance. Table 3 shows that the lower PSI of the $\mathcal{M}_U$ consistently achieves the lowest PSI across position-aware benchmarks. On SQuAD-PosQ, $\mathcal{M}_U$ achieves the highest mean nDCG@10 for five of the eight models. For the remaining three models, its gap to the best skewed configuration is marginal (0.004–0.007). The pattern is even clearer on FineWeb-PosQ, where $\mathcal{M}_U$ achieves the highest mean nDCG@10 for all three evaluated models. Thus, position-balanced training reduces sensitivity to evidence location while maintaining competitive retrieval performance in this controlled setting.

These results also clarify the limitation of skewed training. A skewed model can perform well when the evidence appears at its trained position, but this gain often comes with larger drops at other positions. In contrast, M_U avoids relying on a single evidence location, leading to more stable retrieval across positions competitive retrieval performance.

Early-skewed benchmark subsets can favor early-position priors. After evaluating models

BEIR subset	$\mathcal{M}_B$	$\mathcal{M}_M$	$\mathcal{M}_E$	$\mathcal{M}_U$
SciFact	0.351	0.368	0.340	**0.393**
HotpotQA	**0.338**	0.192	0.165	0.284
FEVER	**0.491**	0.164	0.156	0.357
Climate-FEVER	0.153	0.125	0.109	**0.154**
Average	**0.333**	0.212	0.193	0.297

Table 4: BEIR nDCG@10 averaged over all eight models. Best values are in **bold**; second-best values are underlined. Full results are reported in Table 14.

Model	Config	Peak	Lowest	Δ
ModernBERT-base	$\mathcal{M}_B$	1	9	21.2
	$\mathcal{M}_M$	4	10	9.4
	$\mathcal{M}_E$	9	1	20.6
	$\mathcal{M}_U$	10	2	1.9
Qwen3-0.6B	$\mathcal{M}_B$	1	10	21.5
	$\mathcal{M}_M$	5	10	27.1
	$\mathcal{M}_E$	9	1	20.6
	$\mathcal{M}_U$	9	10	5.5

Table 5: Evidence-moving cosine analysis, where Peak and Lowest denote the insertion positions with the highest and lowest query-document cosine similarity, and Δ is the peak-minus-lowest cosine difference multiplied by 10^3. Full results are reported in Table 13.

on controlled position-aware benchmarks, we test whether training-induced positional priors also affect performance under the standard BEIR evaluation setting, where evidence location is not controlled. Figure 9 shows that the four BEIR subsets differ in their evidence-location distributions. HotpotQA and FEVER are strongly concentrated near the beginning, Climate-FEVER is early-skewed but has a longer tail, and SciFact is broader and less early-concentrated.

Table 4 aligns with these distributional patterns. Across the four BEIR subsets and all eight base models, the begin-trained model $\mathcal{M}_B$ achieves the highest average nDCG@10: 0.333, followed by $\mathcal{M}_U$ at 0.297, $\mathcal{M}_M$ at 0.212, and $\mathcal{M}_E$ at 0.193. The advantage of $\mathcal{M}_B$ over $\mathcal{M}_U$ is largest on the most early-skewed subsets, with a gap of +0.134 on FEVER and +0.054 on HotpotQA. In contrast, this advantage disappears when evidence is less concentrated near the beginning: the gap reverses on SciFact and is nearly zero on Climate-FEVER.

These results suggest that standard benchmark scores can partly reflect evidence-location skew. When evaluation data place much of the relevant evidence near the beginning, a model with an early-position prior can obtain higher scores even if it is less robust to evidence appearing elsewhere. The BEIR results therefore indicate benchmark-specific gains rather than evidence-location robustness.

6 Analyses

Do query-document representations encode positional preference? We first test whether the ranking-level bias observed in Section 5 is reflected in query-document similarity. Following Coelho et al. (2024), we use MS MARCO query-document pairs and relocate the query-relevant evidence to ten uniformly spaced positions within the same document while keeping the remaining content fixed.

Table 5 shows that the highest-similarity position follows the fine-tuning distribution. For both ModernBERT-base and Qwen3-0.6B, $\mathcal{M}_B$ models peak at position 1 and $\mathcal{M}_E$ models peak at position 9. $\mathcal{M}_M$ models peak in the middle range, at p4 for ModernBERT-base and p5 for Qwen3-0.6B. Uniform training produces a flatter pattern: the peak-to-lowest gap drops to 1.9 for ModernBERT-base and 5.5 for Qwen3-0.6B, much smaller than under the concentrated settings.

These results suggest that positional preference is reflected in query-document embedding similarities. Fine-tuning changes which evidence locations appear most similar to the query, rather than affecting only the final ranking output. Full results for all eight models are provided in Appendix F.

How does fine-tuning affect document representations? We next examine whether the document embedding itself reflects positional preference, independent of any query. For each document, we measure cosine similarity between the full-document embedding and embeddings of its ten equal-length segments. Because this analysis is query-free, it can be applied both before and after retrieval fine-tuning. Figure 3 shows ModernBERT-base and Qwen3-0.6B; full results are in Figure 7.

Before retrieval fine-tuning, the pretrained base models show only mild initial positional tendencies: ModernBERT-base is slightly closer to early segments, while Qwen3-0.6B is nearly flat across segments 1–9, with a final-segment spike likely caused by last-token pooling. After fine-tuning, however, the similarity profiles shift toward the training distribution. In Qwen3-0.6B, begin training raises similarity to segment 1 while lowering later non-final segments, end training reverses this pattern, and uniform training compresses the profile across positions. ModernBERT-base shows the same qualitative trend.

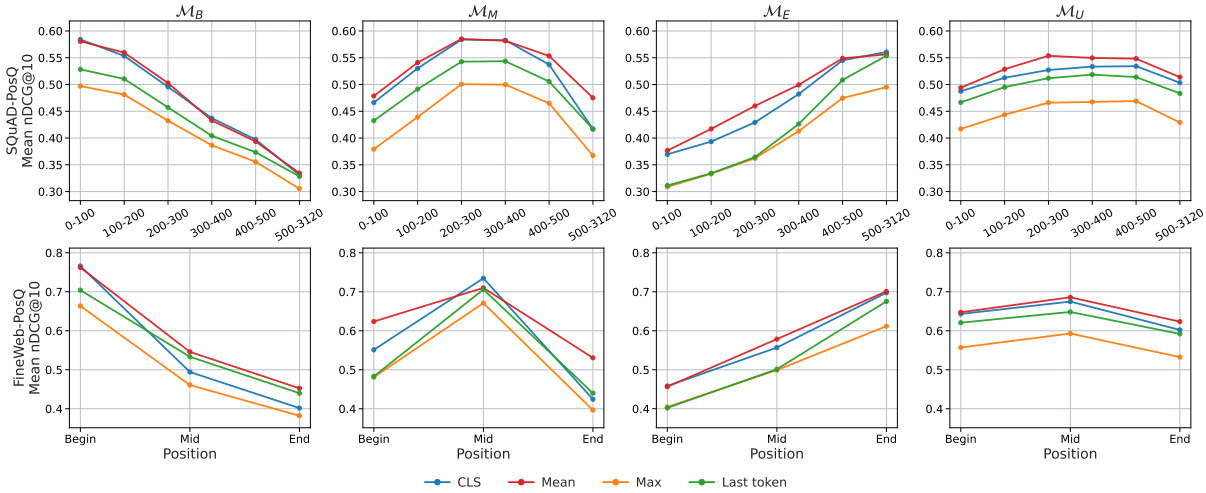

Figure 2: Position-wise nDCG@10 for ModernBERT-base under four pooling strategies. The top and bottom rows report SQuAD-PosQ and FineWeb-PosQ, respectively. Columns correspond to begin-, middle-, end-, and uniform-trained retrievers; lines denote pooling strategies.

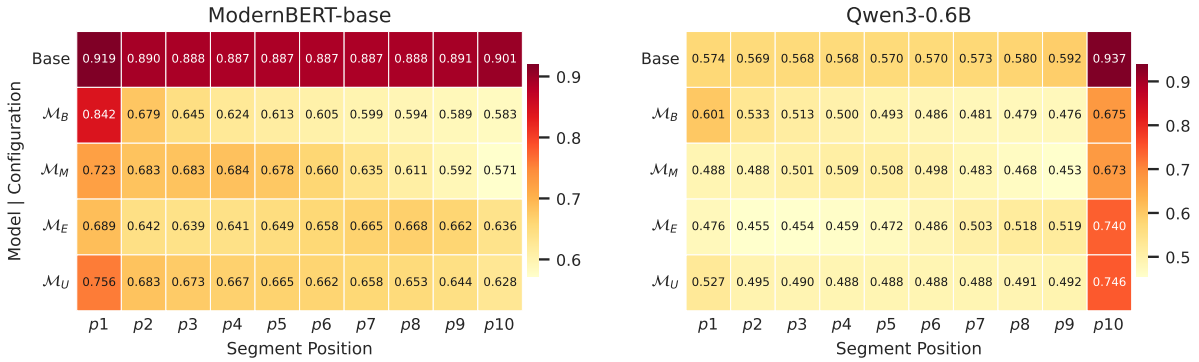

Figure 3: Mean cosine similarity between full-document embeddings and segment embeddings (p_1–p_{10}) for ModernBERT-base and Qwen3-0.6B. Each column corresponds to one of ten equal-length document segments. Full results for all eight models in Figure 7.

These results show that retrieval fine-tuning can reshape document representations, not only query-document matching scores. Although the base checkpoints may retain weak positional tendencies, retrieval fine-tuning largely redirects them toward the positional distribution of the training data.

Does the retrieval-level directional effect depend on pooling strategy? Finally, we examine whether the observed bias direction depends on the pooling strategy. We train ModernBERT-base under the same four positional training distributions using CLS, mean, max, and last-token pooling.

Figure 2 shows that, in this ModernBERT-base ablation, pooling changes absolute retrieval performance but the observed retrieval-level preference remains aligned with the fine-tuning position distribution. Across the four pooling choices tested, position-skewed training leads models to favor the corresponding positions on both SQuAD-PosQ and FineWeb-PosQ. Uniform training again yields a more position-balanced pattern.

These results suggest that the directional effect is not an artifact of a single pooling choice within ModernBERT-base. In this controlled ablation, changing the fine-tuning position distribution has a larger observed effect on retrieval-level bias direction than changing the pooling method.

7 Conclusion

We trained eight architecturally diverse dense retrievers on synthetic position-targeted data and found that skewed fine-tuning distributions induce corresponding ranking-level positional preferences. Representation analyses further suggest that fine-tuning can shift document embeddings toward emphasized evidence positions, while model-specific tendencies remain. Position-balanced training reduces positional sensitivity by 57–87% on controlled position-aware benchmarks with competitive retrieval performance, identifying training-position distribution as a major controllable factor for mitigating retrieval-level position bias.

Limitations

This study focuses on a synthetic, position-targeted fine-tuning setting built from English Wikipedia with LLM-generated queries. Although we match training scale and document-length distributions across configurations, beginning-, middle-, and end-targeted examples use different target segments and generated queries. Thus, physical position may remain partially entangled with segment content, discourse role, query semantics, and difficulty. Our results should therefore be interpreted as evidence that training-position distributions strongly influence retrieval-level bias, not as proof that physical position alone is the sole cause.

Our retained pool is filtered by multiplee rankers and checked with a held-out model-based LLM audit, but it is not human-annotated. Residual labeling errors or verifier-induced biases may remain. In addition, our experiments use a controlled single-seed setup without hard-negative mining, early stopping, or extensive hyperparameter sweeps, so small mean nDCG differences should be treated as point estimates. Finally, we evaluate retrieval-level behavior on position-aware benchmarks and four evidence-annotated BEIR subsets, but not end-to-end RAG or production retrieval systems; future work should test human-validated, multilingual, domain-specific, and downstream settings.

Ethics Statement

This work constructs synthetic position-targeted retrieval data from English Wikipedia and LLM-generated queries. We do not collect private user data, conduct human-subject experiments, or infer protected attributes. However, because Wikipedia contains articles about real individuals, organizations, and sensitive topics, derived examples may include public names or sensitive or offensive content from the source corpus. The artifacts are intended for research on retrieval robustness and position-aware evaluation, not for deployment decisions about individuals or groups. Any released data or code will document the source data, licenses or terms of use, intended use, and known limitations.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *Preprint*, arXiv:2004.05150.

Matteo Catena, Ophir Frieder, Cristina Ioana Muntean, Franco Maria Nardini, Raffaele Perego, and Nicola Tonellotto. 2019. Enhanced news retrieval: Passages lead the way! In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1269–1272, New York, NY, USA. Association for Computing Machinery.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. BGE M3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

João Coelho, Bruno Martins, Joao Magalhaes, Jamie Callan, and Chenyan Xiong. 2024. Dwell in the beginning: How language models embed long documents for dense retrieval. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 370–377, Bangkok, Thailand. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mohsen Fayyaz, Ali Modarressi, Hinrich Schuetze, and Nanyun Peng. 2025. Collapse of dense retrievers: Short, early, and literal biases outranking factual evidence. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9136–9152, Vienna, Austria. Association for Computational Linguistics.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. 2025. Scaling synthetic data creation with 1,000,000,000 personas. *Preprint*, arXiv:2406.20094.

Sebastian Hofstätter, Aldo Lipani, Sophia Althammer, Markus Zlabinger, and Allan Hanbury. 2021. Mitigating the position bias of transformer models in passage re-ranking. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 – April 1, 2021, Proceedings, Part I*, page 238–253, Berlin, Heidelberg. Springer-Verlag.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Preprint*, arXiv:2112.09118.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. 2024. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of*

the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 7036–7050, Mexico City, Mexico. Association for Computational Linguistics.

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.

Reagan J. Lee, Samarth Goel, and Kannan Ramchandran. 2025. Quantifying positional biases in text embedding models. *Preprint*, arXiv:2412.15241.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, 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 *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Horst Po¨ttker. 2003. News and its communicative quality: the inverted pyramid—when and why did it appear? *Journalism Studies*, 4(4):501–511.

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

Elias Schuhmacher, Andrianos Michail, Juri Opitz, Rico Sennrich, and Simon Clematide. 2026. Information representation fairness in long-document embeddings: The peculiar interaction of positional and language bias. *Preprint*, arXiv:2601.16934.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Jie Wang, Tao Ji, Yuanbin Wu, Hang Yan, Tao Gui, Qi Zhang, Xuanjing Huang, and Xiaoling Wang. 2024. Length generalization of causal transformers without position encoding. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14024–14040, Bangkok, Thailand. Association for Computational Linguistics.

Ziqi Wang, Hanlin Zhang, Xiner Li, Kuan-Hao Huang, Chi Han, Shuiwang Ji, Sham M. Kakade, Hao Peng, and Heng Ji. 2025. Eliminating position bias of language models: A mechanistic approach. *Preprint*, arXiv:2407.01100.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Griffin Thomas Adams, Jeremy Howard, and Iacopo Poli. 2025. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2526–2547, Vienna, Austria. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

BigScience Workshop, :, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, and 375 others. 2023. Bloom: A 176b-parameter open-access multilingual language model. *Preprint*, arXiv:2211.05100.

Xinyi Wu, Yifei Wang, Stefanie Jegelka, and Ali Jadbabaie. 2025. On the emergence of position bias in transformers. In *ICML*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Ziyang Zeng, Dun Zhang, Jiacheng Li, Zoupanxiang, Yudong Zhou, and Yuqing Yang. 2025. An empirical study of position bias in modern information retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 5069–5081, Suzhou, China. Association for Computational Linguistics.

Ziyang Zeng, Dun Zhang, Yu Yan, Xu Sun, Yudong Zhou, and Yuqing Yang. 2026. Posir: Position-aware heterogeneous information retrieval benchmark. *Preprint*, arXiv:2601.08363.

Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, Meishan Zhang, Wenjie Li, and Min Zhang. 2024. mgte: Generalized long-context text representation and reranking models for multilingual text retrieval. *Preprint*, arXiv:2407.19669.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren

Zhou. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

A Qwen3-Embedding Style Query Generation

We adopt a two-stage generation pipeline following Zhang et al. (2025). Stage 1 selects a shared configuration (persona, difficulty, query length) for each document, and Stage 2 generates position-conditioned query–answer pairs using that configuration. For each document, the persona candidate set is retrieved from PersonaHub (Ge et al., 2025) by embedding similarity using BGE-M3 (top-k=20). Both stages use GPT-4o-mini with temperature T=1.0 and top-p=1.0.

A.1 Prompt for Configuration Selection (Stage 1)

Given a document and a set of candidate personas, the model selects the most appropriate generation configuration. This configuration is shared across all three positional queries for the same document to keep persona, difficulty, and query length fixed across target positions. Here, `{CHARACTERS}` contains the 20 retrieved personas, each on a separate line. If any field fails to parse, the document is excluded from downstream processing.

A.2 Prompt for Position-Conditioned Query Generation (Stage 2)

For each document–position pair, the model generates a query–answer pair using the shared configuration from Stage 1. The model receives both the full document and the target segment, and is instructed to produce a query answerable only from the target segment. The positional constraint is enforced at the prompt level but is not guaranteed by the generator. Generated pairs are subsequently validated through the multi-reranker filtering pipeline described in Section 3.1.3.

Stage 1 user prompt: configuration selection listing options

```
[USER]
<task>
You are given a passage. Select the
    most appropriate configuration for
     generating a search query about
    this passage.
</task>

<passage> {PASSAGE} </passage>

<instructions>
Based on the passage content, select:
1. Character: A persona who would
    naturally search for this
    information
2. Difficulty: The education level
    appropriate for understanding this
     content
3. Query_Length: The appropriate
    length for the query
</instructions>

<options>
Character Candidates: {CHARACTERS}
Difficulties: high_school, university,
     phd
Query_Lengths: short (under 10 words),
     medium (10--20 words), long (over
     20 words)
</options>

Output as JSON: {"Character": "
    selected character description", "
    Difficulty": "selected difficulty",
     "Query_Length": "selected length"}
```

Figure 4: Stage 1 prompt for configuration selection. A single configuration is determined per document and reused across all positional queries.

Stage 2 user prompt: position-conditioned query generation listing options

```
[USER]
<task>
You are {CHARACTER}. Generate a search query to find information about the TARGET
    SEGMENT within the document. The query should be answerable ONLY by the target
     segment, not by other parts of the document.
</task>

<full_document> {FULL_DOCUMENT} </full_document>

<target_segment position="{POSITION}"> {TARGET_SEGMENT} </target_segment>

<configuration>
- Your persona: {CHARACTER}
- Difficulty level: {DIFFICULTY}
- Target query length: {QUERY_LENGTH}
</configuration>

<requirements>
- Generate a query from your persona's perspective
- The query MUST require information specifically from the target segment to
    answer
- The query should NOT be answerable using only the other parts of the document
- Match the specified difficulty and length
- The answer must be directly extractable from the target segment
</requirements>

Output as JSON: {"query": "the generated search query", "answer": "the answer
    extracted from target segment", "reasoning": "brief explanation of why this
    query targets the specific segment"}
```

Figure 5: Stage 2 prompt for position-conditioned query generation. `{POSITION}` $\in$ {beginning, middle, end}. The configuration fields are inherited from Stage 1.

B Reranker Filtering and Data Quality Audit

This appendix provides additional validation for the multi-reranker filtering step used to construct the position-controlled training data. The goal of this filtering step is to obtain a high-confidence retained pool of candidate examples whose generated queries are grounded in the intended target segment. The retained pool is not used directly as the final training distribution; instead, final training sets are constructed by downsampling within length-position cells, as described in Section 3.1.4.

We first define the consensus margin used for threshold analysis, then report filtering statistics under different margin thresholds. We next validate the retained candidates with an independent segment-wise LLM audit. Finally, we summarize how the final $\delta = 0.3$ training sets are sampled from the retained pool.

B.1 Consensus Margin for Threshold Analysis

Section 3.1.3 defines the multi-reranker position verification using a margin threshold δ. For threshold analysis, we summarize the same filtering rule with a scalar consensus margin. For a candidate example with query q, document d, and intended target position t, we define

$$m_{\text{cons}}(q, d, t) = \min_{R \in \mathcal{R}} \left[R(q, s_t) - \max_{u \neq t} R(q, s_u) \right],$$

where $\mathcal{R}$ is the set of three rerankers and $u \neq t$ ranges over the two non-target positions.

The consensus margin is the smallest target-vs-non-target score gap among the three rerankers. A candidate passes margin threshold δ if and only if $m_{\text{cons}}(q, d, t) \geq \delta$. Thus, $m_{\text{cons}} \geq 0$ means that all three rerankers rank the target segment above both non-target segments, while larger thresholds require stronger agreement that the query is grounded in the intended target segment.

We refer to candidates with $m_{\text{cons}} \geq 0$ as non-failing candidates. The final training sets use the retained pool obtained with margin threshold $\delta = 0.3$.

			Target position within retained pool					
			Begin		Middle		End	
Filter	Retained	% generated	N	Share	N	Share	N	Share
All generated candidates	2,948,006	100.00	–	–	–	–	–	–
Failed top-rank check	945,845	32.08	–	–	–	–	–	–
$m_{\text{cons}} \geq 0$	2,002,161	67.92	925,013	46.20	523,440	26.14	553,708	27.66
$m_{\text{cons}} \geq 0.1$	1,385,322	46.99	754,366	54.45	295,120	21.30	335,836	24.24
$m_{\text{cons}} \geq 0.2$	869,738	29.50	539,882	62.07	149,127	17.15	180,729	20.78
$m_{\text{cons}} \geq 0.3$	481,236	16.32	335,650	69.75	62,904	13.07	82,682	17.18

Table 6: Cumulative multi-reranker filtering statistics for training candidates. Position columns report the number of retained candidates and their share within each thresholded pool. Increasing the margin threshold reduces coverage but requires stronger agreement that the generated query is grounded in the intended target segment.

B.2 Filtering Statistics and Retained-Pool Skew

We first quantify how many generated candidates remain under different cumulative margin thresholds. Table 6 reports the number of retained candidates, their fraction among all generated candidates, and the target-position composition of the retained pool.

The final threshold $\delta = 0.3$ retains 481,236 candidates, including 62,904 middle-targeted and 82,682 end-targeted candidates. Although stricter thresholds make the retained pool increasingly begin-skewed, the $\delta = 0.3$ pool still contains enough examples in every target position for controlled sampling.

Together, these statistics show that the $\delta = 0.3$ retained pool is conservative under our model-based verification criteria and higher-confidence than lower-margin strata, but not position-balanced or length-neutral. We therefore use it only as a source pool for controlled sampling, rather than as the final training distribution.

B.3 Segment-Wise LLM Audit

To independently validate that the reranker margin corresponds to segment-exclusive answerability, we conduct a held-out LLM audit. The audit uses a single binary judge prompt: for each candidate example, we pair the query independently with each of the three document segments and ask whether that segment contains the answer. The intended target position is not revealed to the judge. This audit is used only for post-hoc validation and is not part of the training-set construction pipeline.

Figure 6 shows the exact binary prompt used for all reported audit results. The reported quantities—TargetYes, DistractorNo, and Exclusive—are aggregate metrics computed from independent segment-level yes/no judgments, not separate prompts.

Segment-level LLM audit prompt

```
[SYSTEM]
You are a precise reading
    comprehension evaluator. Given a
    question and a text segment,
    determine whether the segment
    contains the answer to the
    question.

[USER]
Question: {question}

Text:
{segment_text}

Does the text above contain the answer
     to the question?
Respond in JSON: {"reasoning": "...",
    "answer": "yes|no"}
```

Figure 6: Binary segment-level prompt used for the LLM audit. Each query–segment pair is evaluated independently, and the audit metrics are computed by aggregating the resulting yes/no judgments across the target and non-target segments.

Let $\mathcal{P} = \{\text{begin}, \text{middle}, \text{end}\}$. For candidate j, let $t_j \in \mathcal{P}$ be the labeled target position. We let $J_{j,i} \in \{0, 1\}$ denote the binary LLM judgment for whether segment $s_{j,i}$ contains the answer to query q_j, where $J_{j,i} = 1$ indicates `yes` and $J_{j,i} = 0$ indicates `no`. We also let $\bar{J}_{j,i} = 1 - J_{j,i}$, so $\bar{J}_{j,i} = 1$ means that segment $s_{j,i}$ is judged answer-absent. Let N be the number of audited candidate examples.

Reranker Condition	Target Yes	Distractor No	Exclusive
Failed top-rank check	87.7%	73.8%	51.4%
$0 \leq m_{\text{cons}} < 0.1$	93.4%	83.5%	67.0%
$0.1 \leq m_{\text{cons}} < 0.2$	93.7%	90.2%	77.7%
$0.2 \leq m_{\text{cons}} < 0.3$	94.7%	94.4%	85.3%
$0.3 \leq m_{\text{cons}}$	95.4%	97.0%	90.4%

Table 7: Segment-wise LLM audit across non-overlapping consensus-margin strata. Target Yes, Distractor No, and Exclusive are aggregate metrics computed from independent binary yes/no judgments. Exclusive requires the target segment to be judged as containing the answer and both non-target segments to be judged as not containing the answer. Higher values indicate better segment exclusivity.

We report three aggregate metrics:

$$\text{TargetYes} = \frac{1}{N}\sum_{j=1}^{N} J_{j,t_j},$$

$$\text{DistractorNo} = \frac{1}{2N}\sum_{j=1}^{N}\sum_{i\neq t_j} \bar{J}_{j,i},$$

$$\text{Exclusive} = \frac{1}{N}\sum_{j=1}^{N} J_{j,t_j}\prod_{i\neq t_j} \bar{J}_{j,i}.$$

Here, $i \neq t_j$ ranges over the two non-target positions. TargetYes measures whether the labeled target segment is judged answer-containing. DistractorNo measures whether the two non-target segments are judged answer-absent. Exclusive is the strictest metric: it requires the target segment to be judged answer-containing and both non-target segments to be judged answer-absent for the same candidate.

Table 7 reports the audit results across non-overlapping consensus-margin strata. Unlike Table 6, which reports cumulative retained pools, this audit uses disjoint margin ranges so that lower-margin groups are not mixed with candidates that would also pass stricter thresholds. The final $\delta = 0.3$ retained pool corresponds to the highest-margin stratum, $m_{\text{cons}} \geq 0.3$.

The exclusive rate increases monotonically across higher-margin strata, indicating that the reranker margin is an effective precision-control signal. Reranker-failed candidates have the lowest exclusive rate, while the $m_{\text{cons}} \geq 0.3$ group has the highest audited exclusivity. Combined with the retained-pool statistics in Table 6, this supports $\delta = 0.3$ as a conservative high-precision setting: it yields the strongest audited segment exclusivity while still leaving enough candidates for controlled training-set construction.

Length bin	Begin pool	Middle pool	End pool	Budget
256–512	105,652	13,934	21,405	8,189
512–1024	86,495	16,660	21,427	8,189
1024–2048	60,357	13,594	16,691	8,189
2048–4096	43,946	10,527	13,363	8,189
4096–8192	39,200	8,189	9,796	8,189
Total	335,650	62,904	82,682	40,945

Table 8: Final sampling budget from the $\delta = 0.3$ retained pool. The smallest length-position cell is the 4096–8192 middle cell with 8,189 examples, which sets the per-bin budget for concentrated configurations. The uniform configuration samples 2,729 examples from each target position within each length bin, yielding 40,935 examples in total.

B.4 Final Sampling from the $\delta = 0.3$ Retained Pool

The $\delta = 0.3$ retained pool is used as a high-confidence source pool, not as the final training distribution. As shown above, the retained pool is begin-skewed and retention varies by length bin. To prevent these raw counts from becoming confounding factors, we construct final training sets by downsampling within length-position cells.

Table 8 reports the retained cell sizes and the sampling budget used for the final training configurations. The smallest retained length-position cell is the middle-position cell in the 4096–8192 length bin, which contains 8,189 examples. This cell sets the common per-bin sampling budget for concentrated configurations.

Each concentrated configuration samples 8,189 examples from its target position in each length bin, yielding 40,945 training examples. The uniform configuration samples 2,729 examples from each target position within each length bin, yielding 40,935 training examples. The slight difference in total size comes from the integer split of 8,189 examples into three target positions.

This sampling design prevents the final training sets from inheriting the uneven position and length counts of the retained pool. As a result, comparisons across training configurations are not driven by differences in training size or document-length distribution.

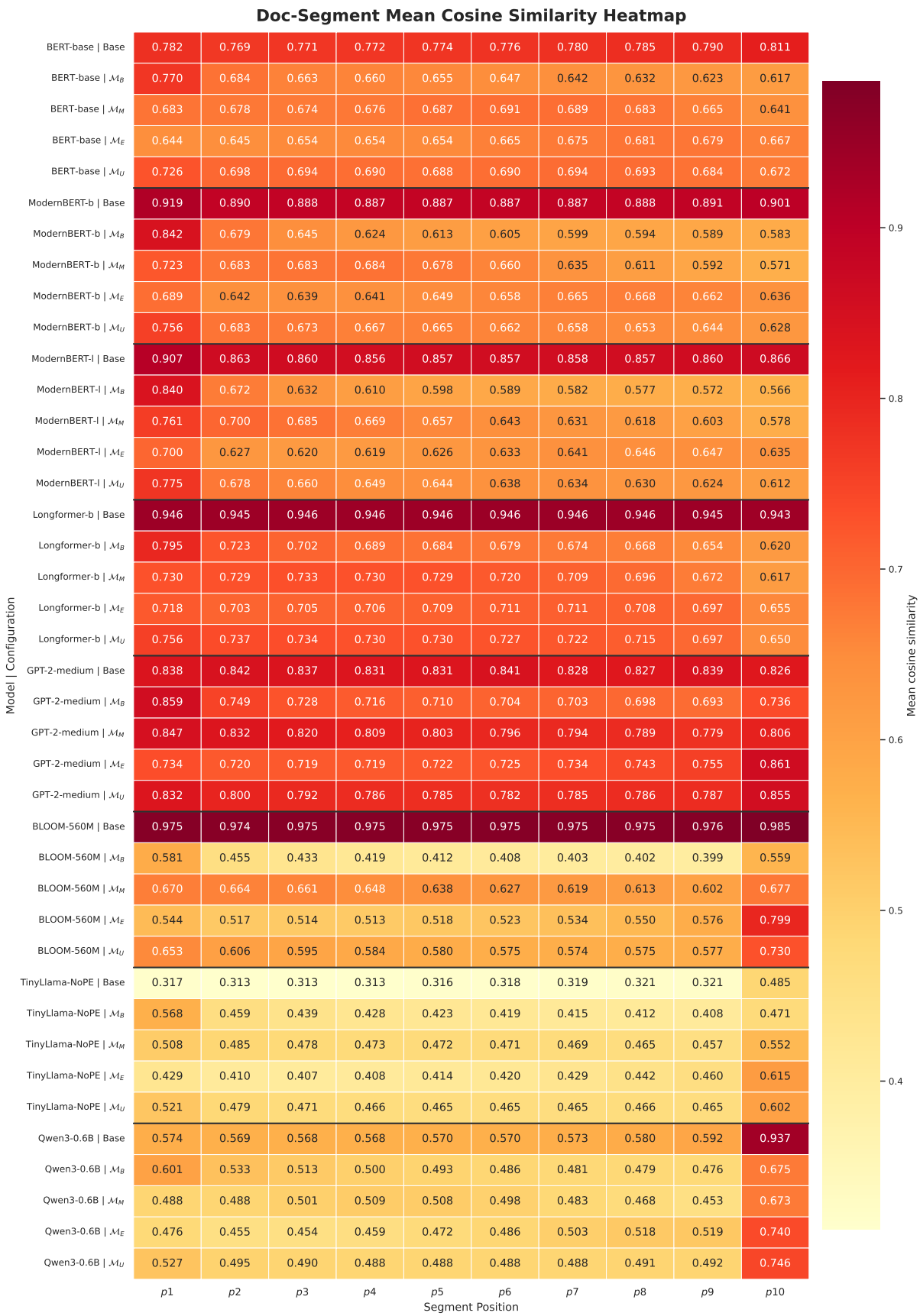

Figure 7: Full-document–segment cosine similarity for all eight models across ten equal-length document segments. "Base" denotes pretrained base models before retrieval fine-tuning; "$\mathcal{M}_B$," "$\mathcal{M}_M$," "$\mathcal{M}_E$," and "$\mathcal{M}_U$" denote retrievers fine-tuned under the corresponding training configuration. Columns p_1–p_{10} denote segment positions.

C Pre-Existing Positional Tendencies in Pretrained Models

As noted in Section 3.2.1, pretrained models are not strictly position-neutral at the representation level. To quantify these pre-existing positional tendencies, we compute cosine similarity between each full-document embedding and the embedding of each of its ten equal-length segments. Figure 7 ("Base" rows, denoting pretrained models before retrieval fine-tuning) reports the results for all eight models. Here, range denotes the maximum minus minimum similarity across the ten document segments.

The base checkpoints show weak, model-specific tendencies rather than a consistent directional bias. Among encoders, Longformer-base is nearly flat (range 0.003), ModernBERT-base and ModernBERT-large show mild early preference (ranges 0.032 and 0.051), and BERT-base is shallowly U-shaped (range 0.042). Among decoders, GPT-2-medium is also nearly flat (range 0.016). BLOOM-560M, TinyLlama-NoPE, and Qwen3-0.6B show a spike at segment 10. Excluding segment 10, their profiles are almost flat, with ranges of 0.002, 0.008, and 0.024, respectively.

These initial tendencies are much smaller than the changes induced by retrieval fine-tuning. ModernBERT-base's range increases from 0.032 before fine-tuning to 0.259 under begin training, and Qwen3-0.6B's range over segments 1–9 increases from 0.024 to 0.125 under begin training and 0.065 under end training. Thus, the main experiments are conservative because the observed bias must arise despite weak, model-specific tendencies already present before fine-tuning.

D Training Details

Table 9 lists the public base checkpoints used in our controlled fine-tuning experiments. All 32 model configurations are trained using the Sentence Transformers library (Wolf et al., 2020) with identical hyperparameters except for the learning rate, which varies by model scale. Table 10 lists the shared training configuration.

We set the learning rate to 4×10^{-5} for base-scale models with fewer than 400M parameters (BERT-base, ModernBERT-base, Longformer-base, GPT-2-medium) and 2×10^{-5} for larger models (ModernBERT-large, BLOOM-560M, TinyLlama-NoPE, Qwen3-0.6B). All runs use the final checkpoint after three epochs with no early stopping or checkpoint selection.

Model	Checkpoint identifier
BERT-base	`google-bert/bert-base-uncased`
Longformer-base	`allenai/longformer-base-4096`
ModernBERT-base	`answerdotai/ModernBERT-base`
ModernBERT-large	`answerdotai/ModernBERT-large`
GPT-2-medium	`openai-community/gpt2-medium`
BLOOM-560M	`bigscience/bloom-560m`
TinyLlama-NoPE	`AntNLP/TinyLlama-NoPE-1.1B`
Qwen3-0.6B	`Qwen/Qwen3-0.6B`

Table 9: Public base checkpoint identifiers used for controlled fine-tuning.

Parameter	**Value**
Loss function	InfoNCE (CachedMNRL)
Optimizer	AdamW
Batch size	256
Epochs	3
Warmup ratio	0.1
Similarity scale	20.0 ($\tau = 0.05$)
Chunk-aware negatives	Enabled
Hard negative mining	None
Seed	42

Table 10: Shared training hyperparameters for all 32 fine-tuning runs.

We prepend an instruction prefix to all query inputs at both training and inference time: `"query: "` for encoder-only models and `"Retrieve a relevant passage: "` for decoder-only models. No prefix is applied to document inputs.

Fine-tuning all 32 model configurations took approximately 6 hours on 8 NVIDIA A100-SXM4-80GB GPUs, or about 48 GPU-hours, excluding data generation, filtering, and evaluation.

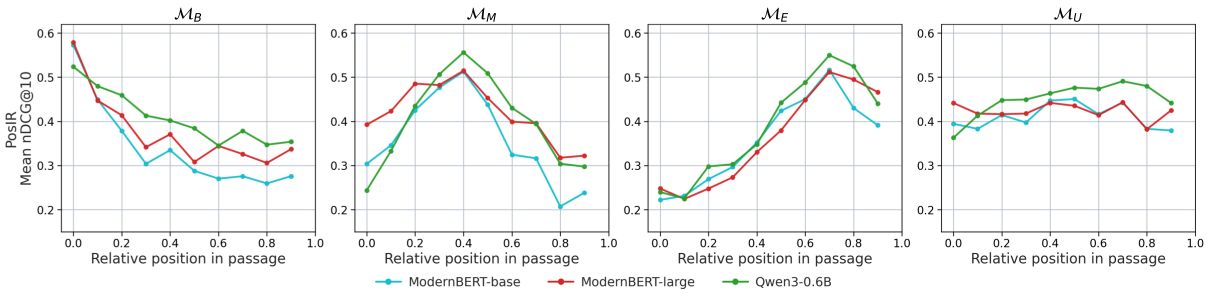

Figure 8: Position-wise nDCG@10 on four selected PosIR domains: Subject Education, News Media, Law Judiciary, and Finance Economics. Columns correspond to configurations, $\mathcal{M}_B$, $\mathcal{M}_M$, $\mathcal{M}_E$, and $\mathcal{M}_U$; lines denote evaluated base models.

	NDCG@10				PSI ↓			
Model	$\mathcal{M}_B$	$\mathcal{M}_M$	$\mathcal{M}_E$	$\mathcal{M}_U$	$\mathcal{M}_B$	$\mathcal{M}_M$	$\mathcal{M}_E$	$\mathcal{M}_U$
PosIR								
ModernBERT-base	0.341	0.359	0.358	**0.411**	0.547	0.596	0.570	**0.158**
ModernBERT-large	0.377	0.419	0.362	**0.423**	0.472	0.383	0.562	**0.138**
Qwen3-0.6B	0.409	0.401	0.386	**0.450**	0.341	0.562	0.590	**0.261**

Table 11: Mean nDCG@10 and Position Sensitivity Index (PSI) on PosIR across training configurations for models with sufficient context length. Higher is better for nDCG@10; lower is better for PSI. Best values for each model and metric are in **bold**.

E Additional Experimental Results: PosIR

Figure 8 reports position-wise nDCG@10 on the PosIR four selected subset domains: Subject Education, News Media, Law Judiciary, and Finance Economics. The same directional pattern observed on SQuAD-PosQ and FineWeb-PosQ also appears on PosIR: begin-trained retrievers ($\mathcal{M}_B$) favor earlier evidence, mid-trained retrievers ($\mathcal{M}_M$) peak around the middle, and end-trained retrievers ($\mathcal{M}_E$) improve toward later evidence. Uniformly trained retrievers ($\mathcal{M}_U$) produce a flatter curve, indicating lower sensitivity to the physical location of the reference evidence.

Table 11 summarizes mean nDCG@10 and PSI. The $\mathcal{M}_U$ achieves the highest mean nDCG@10 for all three evaluated long-context models: 0.411 for ModernBERT-base, 0.423 for ModernBERT-large, and 0.450 for Qwen3-0.6B. It also achieves the lowest PSI for all three models, reducing PSI relative to the worst skewed configuration by 73.5% for ModernBERT-base, 75.4% for ModernBERT-large, and 55.8% for Qwen3-0.6B.

These results show that the main position-aware findings extend to PosIR. Position-skewed training still induces position-specific preferences, while position-balanced training reduces sensitivity to evidence location with higher mean nDCG@10 on the evaluated PosIR domains.

E.1 Mirror-Reversal Diagnostic on PosIR

The original PosIR evaluation measures retrieval performance as a function of the physical position of the reference evidence. As an additional counterfactual diagnostic, we reverse the document order while keeping the query and relevance label fixed. Specifically, each document is divided into five equal contiguous segments and reordered from $1, 2, 3, 4, 5$ to $5, 4, 3, 2, 1$.

We stratify queries by the original location of their reference evidence. Front-origin queries are those whose evidence appears in segment 1 or 2; back-origin queries are those whose evidence appears in segment 4 or 5; and mid-origin queries are those whose evidence appears in segment 3. After reversal, front-origin evidence is mirrored to the back, denoted F→B, while back-origin evidence is mirrored to the front, denoted B→F. Mid-origin evidence remains near the middle.

This setup lets us compare the same origin groups before and after their physical position changes. We define the reversal front–back gap as

$$\Delta_{\text{rev}} = \text{B}{\rightarrow}\text{F} - \text{F}{\rightarrow}\text{B}.$$

Positive Δ_{rev} indicates that the model performs better when originally back evidence is moved to the front; negative Δ_{rev} indicates that the model performs better when originally front evidence is moved to the back.

	Front-origin queries		Back-origin queries		
Model	Orig. front	Rev. F→B	Orig. back	Rev. B→F	Δ_{rev}
MODERNBERT-BASE					
$\mathcal{M}_B$	0.435	0.187	0.272	0.382	+0.194
$\mathcal{M}_M$	0.364	0.238	0.288	0.375	+0.138
$\mathcal{M}_E$	0.238	0.458	0.458	0.331	-0.126
$\mathcal{M}_U$	0.385	0.353	0.413	0.413	+0.060
MODERNBERT-LARGE					
$\mathcal{M}_B$	0.447	0.238	0.326	0.443	+0.205
$\mathcal{M}_M$	0.431	0.310	0.371	0.463	+0.153
$\mathcal{M}_E$	0.232	0.497	0.478	0.360	-0.137
$\mathcal{M}_U$	0.415	0.387	0.416	0.477	+0.090
QWEN3-0.6B					
$\mathcal{M}_B$	0.475	0.353	0.352	0.589	+0.236
$\mathcal{M}_M$	0.354	0.417	0.372	0.531	+0.114
$\mathcal{M}_E$	0.254	0.680	0.502	0.450	-0.230
$\mathcal{M}_U$	0.409	0.581	0.469	0.620	+0.039

Table 12: Original and mirror-reversed PosIR performance for front- and back-origin queries. Queries are stratified by the original location of their reference evidence. Front-origin queries have evidence in segments 1–2; after reversal, this evidence moves to the back, denoted F→B. Back-origin queries have evidence in segments 4–5; after reversal, this evidence moves to the front, denoted B→F. $\mathcal{M}_B$, $\mathcal{M}_M$, and $\mathcal{M}_E$ denote retrievers trained on begin-, middle-, and end-concentrated configurations, respectively, and $\mathcal{U}$ denotes the uniform configuration. We define $\Delta_{\text{rev}} = \text{B}{\rightarrow}\text{F} - \text{F}{\rightarrow}\text{B}$, so positive values indicate a preference for evidence currently placed near the front after reversal, while negative values indicate a preference for evidence currently placed near the back.

F Evidence-Moving Analysis Full Results

Table 13 reports the full position-wise cosine similarity for all eight models under the evidence-moving experiment described in Section 6. The table uses insertion positions $p1$–p10, where $p1$ denotes the earliest insertion position and p10 denotes the latest insertion position.

Five of the eight models—ModernBERT-base, ModernBERT-large, Longformer-base, BLOOM-560M, and Qwen3-0.6B—show clear directional alignment with the fine-tuning distribution. In these models, $\mathcal{M}_B$ models peak at $p1$, $\mathcal{M}_E$ models peak at $p9$ or $p10$, and $\mathcal{M}_M$ models peak in the middle range, between $p3$ and $p5$. These results indicate that the evidence location preferred in the embedding space generally follows the target-position distribution used during fine-tuning.

Uniform training produces the smallest or near-smallest Range for all eight models. It yields the smallest Range for BERT-base, ModernBERT-base, ModernBERT-large, Longformer-base, GPT-2-medium, TinyLlama-NoPE, and Qwern3-0.6B. The only exception is BLOOM-560M, where the mid-trained Range is 4.7 and the uniform-trained Range is 5.4; this difference is small in absolute terms and both values are much lower than the begin- and end-trained ranges of 23.1 and 21.4 Overall, the uniform setting compresses the peak-to-lowest cosine differences and weakens position-specific preference.

The remaining three models exhibit model-specific deviations from clean directional alignment. GPT-2-medium shows a persistent late-position preference: cosine similarity peaks at $p10$ under the begin-, mid-, and end-trained configurations, suggesting that fine-tuning does not fully redirect this pre-existing or architecture-specific tendence. Under uniform training, however, the peak shifts to $p2$ and the Range decreases to 6.0. BERT-base aligns with begin and middle training, peaking at $p1$ under $\mathcal{M}_M$, but its end-trained configuration peaks at $p6$ rather than at $p9$ or $p10$, indicating incomplete alignment with the end-trained distribution. TinyLlama-NoPE aligns under begin and end training, peaking at $p1$ under $\mathcal{M}_B$ and $p10$ under $\mathcal{M}_E$, but fails to shift under mid-training, where the peak remains at $p1$.

Base Model	Config	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	Range
BERT-base	$\mathcal{M}_B$	**.6832**	.6720	.6777	.6805	.6733	.6713	.6730	.6750	.6778	.6803	11.9
	$\mathcal{M}_M$	.6599	.6596	.6577	.6686	**.6869**	.6732	.6722	.6666	.6613	.6722	29.3
	$\mathcal{M}_E$	.6326	.6393	.6371	.6332	.6366	**.6605**	.6582	.6547	.6399	.6337	27.8
	$\mathcal{M}_U$	.6702	.6717	.6715	.6716	.6739	.6775	**.6796**	.6780	.6755	.6746	9.4
ModernBERT-base	$\mathcal{M}_B$	**.6582**	.6454	.6420	.6403	.6387	.6380	.6375	.6369	.6366	.6370	21.5
	$\mathcal{M}_M$	.6045	.6064	.6077	**.6078**	.6074	.6062	.6048	.6027	.6008	.5985	9.4
	$\mathcal{M}_E$	.5622	.5626	.5637	.5653	.5674	.5706	.5759	.5802	**.5829**	.5803	20.6
	$\mathcal{M}_U$	.6144	.6143	.6147	.6148	.6149	.6148	.6151	.6147	.6150	**.6162**	1.9
ModernBERT-large	$\mathcal{M}_B$	**.6452**	.6364	.6345	.6331	.6327	.6322	.6321	.6318	.6315	.6323	13.7
	$\mathcal{M}_M$	.6054	.6062	**.6067**	.6064	.6059	.6051	.6046	.6033	.6024	.6010	5.7
	$\mathcal{M}_E$	.5732	.5747	.5757	.5766	.5777	.5791	.5814	.5834	.5856	**.5881**	14.9
	$\mathcal{M}_U$	.6160	.6168	.6169	.6165	.6164	.6166	.6169	.6169	.6169	**.6175**	1.6
Longformer-base	$\mathcal{M}_B$	**.6261**	.6204	.6188	.6174	.6174	.6180	.6192	.6203	.6183	.6164	9.7
	$\mathcal{M}_M$	.5838	.5856	.5871	.5877	**.5878**	.5870	.5869	.5855	.5849	.5807	7.1
	$\mathcal{M}_E$	.5622	.5641	.5656	.5660	.5671	.5681	.5691	.5696	.5712	**.5779**	15.7
	$\mathcal{M}_U$	.6030	.6033	.6043	.6042	.6049	.6050	.6056	.6058	.6057	**.6061**	3.1
GPT-2-medium	$\mathcal{M}_B$	.6637	.6684	.6666	.6655	.6686	.6693	.6658	.6711	.6687	**.6713**	7.6
	$\mathcal{M}_M$	.6984	.6989	.6969	.6945	.6983	.6994	.6960	.7004	.6979	**.7009**	6.5
	$\mathcal{M}_E$	.7044	.7040	.7029	.7011	.7023	.7047	.7030	.7049	.7060	**.7109**	9.8
	$\mathcal{M}_U$	.6831	**.6876**	.6846	.6816	.6861	.6868	.6822	.6861	.6851	.6819	6.0
BLOOM-560M	$\mathcal{M}_B$	**.3289**	.3174	.3164	.3167	.3185	.3146	.3163	.3149	.3143	.3058	23.1
	$\mathcal{M}_M$	.5459	.5483	.5487	**.5489**	.5482	.5465	.5467	.5458	.5460	.5442	4.7
	$\mathcal{M}_E$	.5576	.5562	.5608	.5627	.5629	.5601	.5673	.5686	**.5776**	.5579	21.4
	$\mathcal{M}_U$	.6026	.6024	.6032	.6037	.6032	.6004	.6029	.6034	.6054	**.6058**	5.4
TinyLlama-NoPE	$\mathcal{M}_B$	**.4401**	.4266	.4254	.4238	.4231	.4235	.4228	.4238	.4228	.4207	19.5
	$\mathcal{M}_M$	**.4514**	.4483	.4479	.4477	.4471	.4472	.4476	.4477	.4472	.4378	13.6
	$\mathcal{M}_E$	.4068	.4034	.4035	.4042	.4055	.4068	.4081	.4095	.4155	**.4182**	14.8
	$\mathcal{M}_U$	**.4363**	.4324	.4323	.4322	.4322	.4321	.4326	.4331	.4334	.4276	8.7
Qwen3-0.6B	$\mathcal{M}_B$	**.6085**	.6035	.6003	.5986	.5989	.5981	.5986	.5977	.5981	.5870	21.5
	$\mathcal{M}_M$	.5401	.5428	.5459	.5489	**.5509**	.5484	.5451	.5417	.5397	.5238	27.1
	$\mathcal{M}_E$	.5161	.5174	.5172	.5177	.5216	.5254	.5309	.5353	**.5368**	.5278	20.6
	$\mathcal{M}_U$	.5498	.5519	.5513	.5505	.5518	.5516	.5521	.5516	**.5522**	.5467	5.5

Table 13: Full evidence-moving cosine similarity across all eight models within each document ($p1$–$p10$). **Bold** marks the highest cosine (peak); underline marks the lowest cosine. **Range** is the cosine difference between peak and lowest ($\times 10^3$).

These model-specific deviations do not change the ranking-level conclusions in Section 5: across all eight models, position-skewed fine-tuning induces retrieval behavior aligned with the corresponding training-position distribution. Instead, Table 13 shows that the strength and exact embedding-level location of this prefernece can vary by architecture, with some models retaining residual positional tendencies even after controlled fine-tuning.

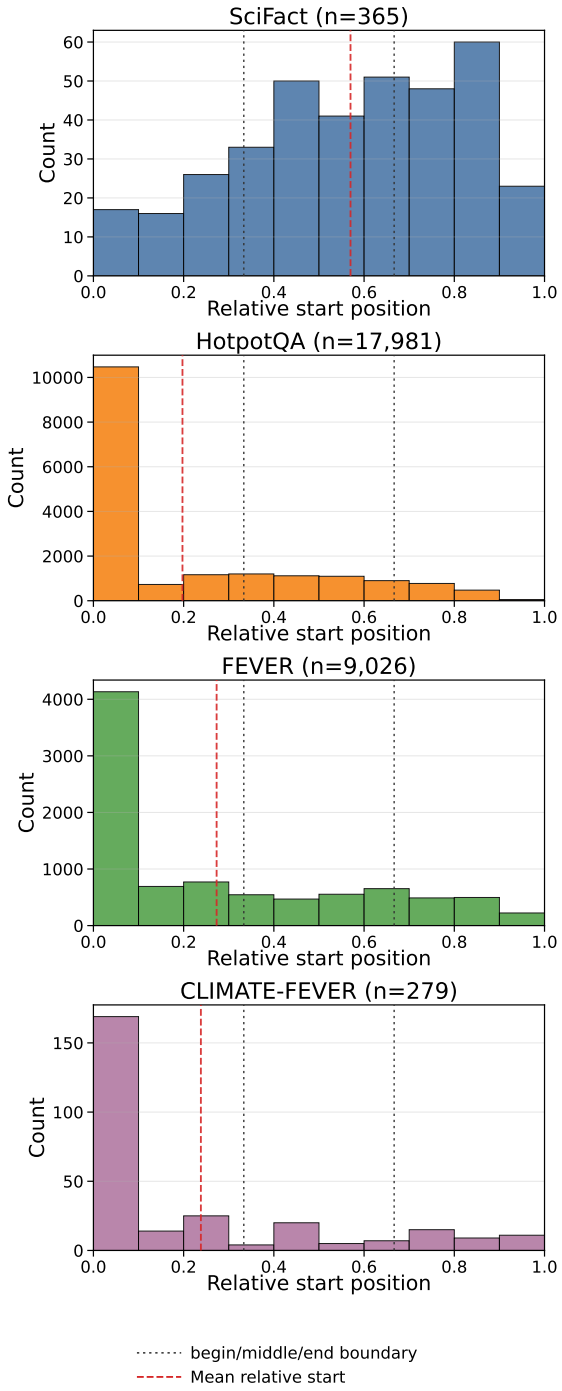

Figure 9: Relative evidence start-position distributions for the BEIR subsets used in our analysis, restricted to examples where the evidence location could be identified. Evidence start positions are normalized by relevant-document length. Dashed red lines indicate mean relative start positions, and dotted gray lines mark the begin/middle/end boundaries.

BEIR subset	$\mathcal{M}_B$	$\mathcal{M}_M$	$\mathcal{M}_E$	$\mathcal{M}_U$
BERT-BASE				
SciFact	**0.298**	0.154	0.138	0.273
HotpotQA	**0.271**	0.110	0.077	0.190
FEVER	**0.432**	0.088	0.047	0.225
Climate-FEVER	**0.137**	0.082	0.074	0.130
Average	**0.285**	0.108	0.084	0.205
MODERNBERT-BASE				
SciFact	0.422	0.379	0.383	**0.434**
HotpotQA	**0.402**	0.251	0.209	0.335
FEVER	**0.604**	0.199	0.200	0.443
Climate-FEVER	0.176	0.136	0.115	**0.185**
Average	**0.401**	0.241	0.227	0.349
MODERNBERT-LARGE				
SciFact	0.472	**0.499**	0.440	0.497
HotpotQA	**0.432**	0.254	0.224	0.354
FEVER	**0.660**	0.259	0.229	0.497
Climate-FEVER	**0.217**	0.175	0.140	0.205
Average	**0.445**	0.297	0.258	0.388
LONGFORMER-BASE				
SciFact	**0.371**	0.370	0.319	0.370
HotpotQA	**0.408**	0.255	0.253	0.358
FEVER	**0.559**	0.235	0.297	0.470
Climate-FEVER	0.186	0.185	0.168	**0.204**
Average	**0.381**	0.262	0.259	0.351
GPT-2-MEDIUM				
SciFact	0.084	**0.148**	0.122	0.119
HotpotQA	**0.096**	0.050	0.031	0.053
FEVER	**0.213**	0.049	0.030	0.086
Climate-FEVER	0.048	0.054	**0.056**	0.037
Average	**0.110**	0.075	0.060	0.074
QWEN3-0.6B				
SciFact	0.550	0.604	**0.628**	0.624
HotpotQA	**0.485**	0.299	0.299	0.481
FEVER	**0.657**	0.286	0.296	0.577
Climate-FEVER	**0.231**	0.171	0.182	0.215
Average	**0.481**	0.340	0.351	0.474
BLOOM-560M				
SciFact	0.365	0.458	0.384	**0.470**
HotpotQA	**0.331**	0.133	0.098	0.251
FEVER	**0.508**	0.082	0.082	0.346
Climate-FEVER	0.131	0.097	0.071	**0.141**
Average	**0.334**	0.192	0.159	0.302
TINYLLAMA-1.1B				
SciFact	0.243	0.329	0.308	**0.354**
HotpotQA	**0.280**	0.183	0.129	0.254
FEVER	**0.291**	0.116	0.069	0.213
Climate-FEVER	0.103	0.104	0.070	**0.117**
Average	0.229	0.183	0.144	**0.234**

Table 14: Full BEIR nDCG@10 by base model and training configuration. Averages are computed over the four BEIR subsets. Higher is better. Best values within each model–subset row are in bold; second-best values are underlined.