Online-Efficient Long-Document Reranking via Block-Level Embeddings and Top-k Interaction Refinement

Anonymous ACL submission

Abstract

Representation-based rerankers are attractive for long-document search because document-side embeddings can be computed offline, yet a single vector often washes out sparse relevance signals in long and heterogeneous documents. Interaction-heavy rerankers alleviate this problem, but their online cost grows with the number and length of candidate documents. We present BReps-TIR, a fixed-candidate long-document reranking framework that preserves the offline/online separation of dense rerankers while retaining localized block evidence. BReps segments each candidate document into short blocks, encodes each block independently, and scores the document by a descending weighted aggregation over its top-k query–block similarities. To calibrate the selected blocks, BReps-TIR adds a lightweight query-conditioned setwise module that refines only the top-k block scores with bounded residual corrections. Across TREC DL'19/22/23 and MLDR-zh, BReps-TIR consistently improves early-ranking effectiveness over single-vector LLM encoders and reduces online reranking computation relative to segmentation-and-cross-encoder baselines, at the cost of a substantially larger document-side vector store.

1 Introduction

Large language models (LLMs) have become increasingly useful for neural information retrieval, both as rerankers and as representation learners for dense retrieval (Ma et al., 2024; Zhao et al., 2024). However, long-document reranking remains a difficult regime. The relevant evidence for a query may occupy only a small portion of a document, while the remaining content introduces topical drift and irrelevant context. Encoding the entire document into a single vector can therefore dilute fine-grained relevance signals; jointly encoding each query–document pair with an interaction model can recover these signals, but is often too expensive for reranking many long candidates online.

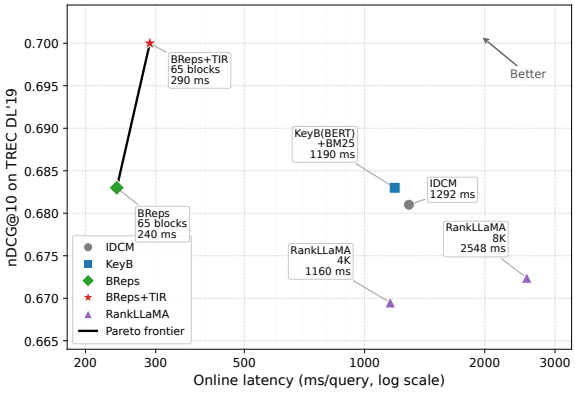

Figure 1: Effectiveness–efficiency trade-off on TREC DL'19. We plot NDCG@10 against online reranking latency for reranking the top-100 candidates on a single NVIDIA A100 GPU; document-side block embeddings are precomputed offline. Top-left is better.

This paper studies *fixed-candidate long-document reranking*: given the same candidate set for every method, the goal is to assign better document scores while keeping online cost practical. This positioning is important. We do not claim to replace all interaction-heavy rerankers in unconstrained settings; instead, we ask whether representation-based LLM rerankers can be made substantially more faithful to localized evidence without giving up their main deployment advantage: precomputable document-side representations.

Figure 1 previews this trade-off on TREC DL'19: BReps-TIR improves early-rank effectiveness while keeping online latency low because document-side block embeddings are computed offline. The same design also increases the stored document representation, so we explicitly quantify the vector-store footprint in Section 6.3.

We propose BReps, a block-level representation framework for long-document reranking. Instead

of compressing a document into a single embedding, BReps represents it with multiple short block embeddings. At query time, the query vector is compared with all stored blocks of a candidate document, and the document score is computed from the top-k block similarities using a simple descending weighted aggregation. This design keeps document representations offline while allowing document scores to be computed from the most relevant local blocks rather than from a single global vector.

We further introduce BReps-TIR (Top-k Interaction Refinement), a lightweight module that operates only on the selected top-k blocks. The motivation is that top-k block aggregation treats blocks independently: repeated blocks can be over-counted, while complementary medium-scoring blocks may be under-valued. TIR uses query-conditioned attention over the selected blocks to compute a setwise context and applies a bounded residual correction to each block score before aggregation. Because TIR never re-encodes full documents and only processes a small selected set, it adds modest online overhead.

This design makes three practical trade-offs explicit. First, it targets fixed-candidate reranking rather than unconstrained first-stage retrieval, where all methods rerank the same candidate documents. Second, it moves most document-side computation offline while still using localized block scores at query time. Third, it exposes the central deployment trade-off: a larger document-side vector store in exchange for lower online model computation and stronger local-evidence modeling.

Our contributions are:

- We propose BReps, a block-level representation framework for online-efficient long-document reranking with precomputed document-side embeddings.
- We introduce BReps-TIR, a lightweight top-k setwise refinement module that calibrates selected block scores via bounded residual corrections.
- We evaluate on TREC DL'19/22/23 and MLDR-zh under fixed candidate sets, showing consistent gains over single-vector LLM encoders and lower online model computation than segmentation-and-cross-encoder rerankers, while quantifying the larger document-side vector store required by block representations.
- We provide ablations on segmentation, aggregation, block budget, and TIR components, together with latency and storage analyses.

2 Related Work

Representation-based and interaction-based reranking. Dense bi-encoders and representation-based rerankers encode queries and documents separately, enabling efficient similarity computation and offline document indexing. RepLLaMA adapts decoder-only LLMs to retrieval by producing single-vector representations for queries and documents (Ma et al., 2024). This efficiency comes at the cost of compressing long documents into one vector. In contrast, cross-encoders jointly encode query–document pairs and generally provide strong relevance estimation (Nogueira and Cho, 2019), but their online cost is high for long-document reranking. Our work targets the middle ground: multiple document-side representations are precomputed offline, while only lightweight top-block interaction is performed online. Recent work constructs compact textual evidence contexts for efficient interaction-based long-document reranking (Li et al., 2024). In contrast, we precompute block-level document embeddings and refine only top-k scores online, trading larger offline storage for substantially lower online model computation.

Late interaction and multi-vector retrieval. Late-interaction models such as ColBERT preserve token-level document representations and perform fine-grained max-sim matching at query time (Khattab and Zaharia, 2020). They demonstrate the value of multi-vector document representations, but token-level storage and interaction can be expensive for very long documents. BReps instead stores block-level vectors, providing a coarser but more compact multi-vector representation. This choice is aimed at the long-document reranking regime where the main constraint is not only matching quality, but also the cost of materializing and querying many document-side vectors.

Long-document retrieval and block selection. Long-document retrieval has a long history of passage-level scoring, segmentation, and evidence aggregation. Recent long-document neural

rerankers such as IDCM and KeyB select or aggregate informative segments before scoring (Hofstätter et al., 2021; Li et al., 2023). Our method shares the premise that long documents should not be treated as indivisible units, but differs by making the block representation itself precomputable and by refining only top selected block scores with a small setwise module.

Long-context embedding models. Long-context embedding models such as E5-Base-4K and BGE-M3 extend the input window for representation learning (Zhu et al., 2024; Chen et al., 2024). They are strong baselines for long-document representation, but a single long-context representation can still mix relevant and irrelevant regions. BReps explicitly preserves multiple local evidence vectors and therefore avoids forcing all document evidence into a single embedding.

3 Method

Figure 2 summarizes BReps-TIR. Let q be a query and d be a candidate document. The document is segmented into blocks $b_1, \ldots, b_n$. The encoder produces a query representation $\mathbf{v}_q \in \mathbb{R}^H$ and block representations $\mathbf{v}_{b_i} \in \mathbb{R}^H$. All block representations can be computed and stored offline.

3.1 Block-Level Representation

We segment each document into short blocks using the CogLTX dynamic-programming segmentation strategy (Ding et al., 2020), with a maximum block length of 63 tokens. This encourages boundaries near punctuation while respecting a fixed token budget. Each block is encoded independently using the same decoder-only LLM encoder style as RepLLaMA:

$$\mathbf{v}_{b_i} = \mathrm{Enc}(\texttt{passage: } b_i \texttt{ </s>})[-1], \quad (1)$$

$$\mathbf{v}_q = \mathrm{Enc}(\texttt{query: } q \texttt{ </s>})[-1]. \quad (2)$$

We L2-normalize representations and compute scaled cosine similarities:

$$s_i = \cos(\mathbf{v}_q, \mathbf{v}_{b_i})/\tau_s, \quad (3)$$

where $\tau_s = 0.01$ follows the RepLLaMA-style score scaling used in training.

3.2 Weighted Top-k Aggregation

Let $s_{(1)} \geq \cdots \geq s_{(k)}$ denote the top-k block scores for document d. BReps computes the initial document score as:

$$S_{\mathrm{BReps}}(q, d) = \sum_{i=1}^{k} w_i s_{(i)}, \quad (4)$$

$$w_1 \geq \cdots \geq w_k, \quad \sum_i w_i = 1. \quad (5)$$

In the default setting, we use a fixed descending weight schedule. This simple aggregation is deliberately conservative: it gives priority to the strongest local evidence while still allowing multiple blocks to support the document-level score. Unlike max pooling, it does not discard secondary evidence; unlike uniform averaging, it does not treat weak tail blocks as equally important.

3.3 Top-k Interaction Refinement

The aggregation in Eq. 5 treats selected blocks independently. This can over-count redundant evidence and under-value complementary blocks. TIR therefore refines only the top-k selected block scores before aggregation. Crucially, TIR does not replace the original relevance signal; it applies a bounded correction to calibrate it.

Let $\mathbf{B} = [\mathbf{b}_1; \ldots; \mathbf{b}_k] \in \mathbb{R}^{k \times H}$ be the selected block representations sorted by initial score, and let $\mathbf{s} \in \mathbb{R}^k$ be their scores. After layer normalization, TIR computes query-to-block attention in a low-dimensional projection space $d \ll H$:

$$\hat{\mathbf{q}} = W_q^a \mathrm{LN}(\mathbf{v}_q), \quad (6)$$

$$\hat{\mathbf{B}} = \mathrm{LN}(\mathbf{B})(W_b^a)^\top, \quad (7)$$

$$\boldsymbol{\alpha} = \mathrm{softmax}\left(\frac{\hat{\mathbf{B}}\hat{\mathbf{q}}}{\sqrt{d}\tau_a}\right), \quad (8)$$

$$\mathbf{c} = \sum_{i=1}^{k} \alpha_i \mathrm{LN}(\mathbf{b}_i). \quad (9)$$

The context vector $\mathbf{c}$ summarizes which selected blocks the query attends to. For each block, TIR fuses the projected query, block, and setwise context:

$$\mathbf{z}_i = \tanh(W_q \mathrm{LN}(\mathbf{v}_q) + W_b \mathrm{LN}(\mathbf{b}_i) + W_c \mathrm{LN}(\mathbf{c}) + g(s_i)), \quad (10)$$

where $g(\cdot)$ is a small score-gating MLP. The raw correction is mapped to a bounded residual:

$$\delta_i = \gamma \tanh(\mathbf{w}_\delta^\top \mathbf{z}_i), \quad s_i' = s_i + \delta_i. \quad (11)$$

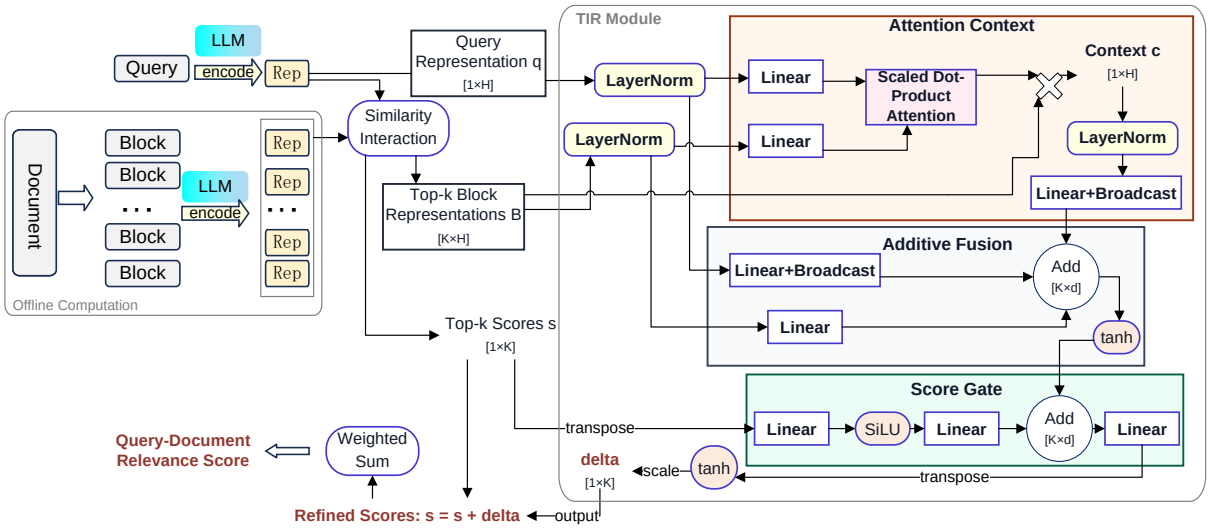

Figure 2: Overview of BReps-TIR. Documents are segmented and encoded into block embeddings offline. At query time, the query is encoded once, candidate blocks are scored independently, and the top-k block scores are aggregated. TIR refines only the top-k selected blocks with query-conditioned setwise interaction and bounded residual corrections.

The refined document score is:

$$S_{\mathrm{TIR}}(q, d) = \sum_{i=1}^{k} w_i s'_i. \tag{12}$$

The residual scale γ ensures that TIR calibrates rather than overwrites the original similarity scores.

3.4 Training

We train with triplets (q, d^+, d^-) and a pairwise hinge loss:

$$\mathcal{L} = \max(0, m - S(q, d^+) + S(q, d^-)). \tag{13}$$

For BReps, S is Eq. 5; for BReps-TIR, S is the refined score. We keep the hinge objective to match the training protocol of the single-vector LLM reranker baseline, so that the main comparison isolates the effect of block-level representation and top-block refinement.

4 Experimental Setup

4.1 Datasets

We evaluate on four long-document reranking benchmarks: TREC DL 2019 document ranking based on MS MARCO v1 (Craswell et al., 2020; Nguyen et al., 2016), TREC DL 2022/2023 document ranking based on MS MARCO v2 (Craswell et al., 2025a,b), and MLDR-zh, a Chinese long-document benchmark built from open-domain corpora including Wudao (Yuan et al., 2021). Table 1 summarizes dataset statistics.

Table 1: Dataset statistics. Average length is measured with the Gemma-2-2B tokenizer.

Dataset	genre	#documents	#test query	Avg. Num. of Tokens
TREC 2019 DL doc	Web Documents	3,213,835	49	1670
TREC 2022 DL doc	Web Documents	11,959,635	76	3542
TREC 2023 DL doc	Web Documents	11,959,635	82	3542
MLDR-zh	Wikipedia + Wudao	200000	800	9030

4.2 Baselines

We compare against four groups of baselines. **Lexical and short-context neural baselines**: BM25 (Robertson and Zaragoza, 2009), BERT-dot (Reimers and Gurevych, 2019), and ColBERT (Khattab and Zaharia, 2020). **Single-vector LLM rerankers**: RepLLaMA results where available from prior fixed-candidate evaluations, and our RankLLaMA-style Gemma2-2B single-vector control, evaluated with 2k, 4k, and 8k inference token budgets. **Long-context embedding models**: E5-Base-4K (Zhu et al., 2024) and BGE-M3 (Chen et al., 2024). **Segmentation-and-interaction long-document rerankers**: IDCM (Hofstätter et al., 2021) and KeyB(BERT)$_{\mathrm{BM25}}$ (Li et al., 2023).

4.3 Implementation Details

We use Gemma-2-2B as the base encoder and fine-tune with LoRA on a single NVIDIA V100 32GB. For TREC DL'19 and the MS MARCO v2-based DL'22/DL'23 setting, we train on 20k triplets; for MLDR-zh, we use the same first-negative MLDR-zh training file as the BReps training pipeline. We use LoRA rank $r = 32$, $\alpha = 64$, learning

rate 5×10^{-5}, query length 32, mixed precision, and pairwise hinge margin $m = 10$ in the scaled score space. Unless otherwise specified, documents are segmented into 63-token blocks, trained with $k = 20$ blocks, and evaluated with up to $k = 65$ blocks under a 4k-token budget. TIR uses projection dimension $d = 256$, attention temperature $\tau_a = 0.07$, and residual scale $\gamma = 0.3$. Our RankLLaMA-style Gemma2-2B control follows the same backbone, LoRA configuration, learning rate, hinge loss, and training triplet budget as the corresponding BReps run, but encodes each candidate document into one representation rather than block-level representations. At inference time, we report 2k, 4k, and 8k token budgets for this baseline to test whether simply exposing the single-vector encoder to more document context closes the gap to block-level reranking.

Reranking protocol. All methods rerank identical candidate sets. For TREC DL, we rerank top-100 candidates per query. For MLDR-zh, we rerank the official candidate set of one positive and seven negatives per query. We report NDCG@10 and MAP for TREC DL, and P@1, MAP, and NDCG@8 for MLDR-zh.

Efficiency protocol. We report online reranking latency in ms/query for reranking the top-100 candidates per query on TREC DL'19. For efficiency evaluation, the online latency in Figure 1 is measured on a single NVIDIA A100 GPU and includes tokenization, host-to-device transfer, inference, and output writing, but excludes model loading and any offline document embedding precomputation. For BReps and BReps-TIR we use batch size 8; for the single-vector Gemma2-2B baseline, the batch size is reduced for longer inference contexts when required by GPU memory. Offline representation computation is measured separately on a single NVIDIA V100 32GB GPU. For storage, we distinguish the true stored block embedding dimension H from the TIR internal projection dimension $d = 256$; storage estimates are based on H and report vector payloads only, excluding metadata and ANN-index overhead.

5 Main Results

On TREC DL'19 (Table 2), BReps-TIR achieves the strongest NDCG@10 among the reported methods. The improvement over the RankLLaMA-style Gemma2-2B single-vector control is numerically consistent but not statistically significant on this small DL'19 test set, while the single-vector control obtains slightly higher MAP. This difference is expected because MAP averages precision over the whole reranked list, whereas NDCG@10 emphasizes the top ranks most visible to users; longer single-vector contexts can improve recall-oriented ranking depth, while BReps-TIR concentrates evidence in the highest-ranked positions. Compared with IDCM and KeyB, BReps-TIR achieves a favorable balance: it benefits from localized evidence while avoiding full query–text re-encoding for every segment at query time.

Table 2: TREC 2019 DL (Document Ranking). Superscripts ^{b} and ^{t} denote statistically significant difference from **BReps-65** and **BReps-TIR-65**, respectively (paired t-test on per-query scores, $p < 0.05$).

Model	Context	NDCG@10	MAP
Baseline models			
BM25	Full document	0.488	0.234
BERT-dot	110M, 512 tokens	0.638	0.241
ColBERT	110M, 512 tokens	0.642	0.268
PLMs and LLMs for long documents			
E5-Base-4K	110M, 4096 tokens	0.643^{t}	0.274
BGE-M3	560M, 4096 tokens	0.639^{t}	0.261^{bt}
BGE-M3	560M, 8192 tokens	0.632^{bt}	0.258^{bt}
RankLLaMA	Gemma2-2B, 2k tokens	0.675	0.279
RankLLaMA	Gemma2-2B, 4k tokens	0.670	0.275
RankLLaMA	Gemma2-2B, 8k tokens	0.672	0.278
IDCM	4096 tokens	0.681	0.277
KeyB(BERT)$_{BM25}$	full doc	0.683	**0.281**
Proposed: BReps and BReps-TIR			
BReps-65	$\leq$4095 tokens	0.683	0.274
BReps-TIR-65	$\leq$4095 tokens	**0.698**	0.275

Table 3: TREC 2022 DL (Document Ranking). Superscripts ^{b} and ^{t} denote statistically significant difference from **BReps-65** and **BReps-TIR-65**, respectively (paired t-test on per-query scores, $p < 0.05$).

Model	Context	NDCG@10	MAP
Baseline models			
BM25	Full document	0.299	0.080
BERT-dot	110M, 512 tokens	0.324	0.088
ColBERT	110M, 512 tokens	0.349	0.096
PLMs and LLMs for long documents			
E5-Base-4K	110M, 4096 tokens	0.399^{t}	0.107^{t}
BGE-M3	560M, 4096 tokens	0.401^{t}	0.107^{t}
BGE-M3	560M, 8192 tokens	0.393^{t}	0.106^{t}
RepLLaMA	2B, 2k tokens	0.374^{t}	0.098^{t}
RepLLaMA	2B, 4k tokens	0.375^{t}	0.099^{t}
RankLLaMA	Gemma2-2B, 2k tokens	0.385^{t}	0.102^{t}
RankLLaMA	Gemma2-2B, 4k tokens	0.385^{t}	0.102^{t}
RankLLaMA	Gemma2-2B, 8k tokens	0.380^{t}	0.101^{t}
IDCM	4096 tokens	0.402^{t}	0.103^{t}
KeyB(BERT)$_{BM25}$	full doc	0.362^{bt}	0.099^{t}
Proposed: BReps and BReps-TIR			
BReps-65	$\leq$4095 tokens	0.398^{t}	0.106^{t}
BReps-TIR-65	$\leq$4095 tokens	$\mathbf{0.436}^{b}$	$\mathbf{0.115}^{b}$

Tables 3 and 4 show that the same pattern holds on MS MARCO v2-based DL22/DL23, whose doc-

Table 4: TREC 2023 DL (Document Ranking). Superscripts ^{b} and ^{t} denote statistically significant difference from **BReps-65** and **BReps-TIR-65**, respectively (paired t-test on per-query scores, $p < 0.05$).

Model	Context	NDCG@10	MAP
Baseline models			
BM25	Full document	0.295	0.105
BERT-dot	110M, 512 tokens	0.296	0.101
ColBERT	110M, 512 tokens	0.315	0.111
PLMs and LLMs for long documents			
E5-Base-4K	110M, 4096 tokens	0.391^t	0.133^t
BGE-M3	560M, 4096 tokens	0.415^t	0.141
BGE-M3	560M, 8192 tokens	0.400^t	0.136^t
RankLLaMA	Gemma2-2B, 2k tokens	0.355^{bt}	0.120^{bt}
RankLLaMA	Gemma2-2B, 4k tokens	0.353^{bt}	0.122^{bt}
RankLLaMA	Gemma2-2B, 8k tokens	0.350^{bt}	0.120^{bt}
IDCM	4096 tokens	0.400^t	0.132^t
KeyB(BERT)$_{BM25}$	full doc	0.352^{bt}	0.123^{bt}
Proposed: BReps and BReps-TIR			
BReps-65	$\leq$4095 tokens	0.395^t	0.133^t
BReps-TIR-65	$\leq$4095 tokens	$\mathbf{0.450}^b$	$\mathbf{0.147}^b$

uments are substantially longer than DL19. The RankLLaMA-style Gemma2-2B single-vector control reaches 0.385 NDCG@10 on DL22 with 2k–4k inference budgets, but remains substantially below BReps-TIR with statistically significant differences on both NDCG@10 and MAP. Notably, BReps-TIR improves over BReps across both datasets, indicating that TIR contributes beyond block-level representation alone. The improvement is especially clear on DL23, where BReps-TIR significantly outperforms the single-vector Gemma2-2B baseline and obtains a large NDCG@10 gain over BGE-M3 under comparable 4k-token budgets.

Table 5: Results on MLDR-zh (Document Ranking). For models truncated at 65 blocks, superscripts ^{b} and ^{t} denote statistically significant difference from **BReps-65** and **BReps-TIR-65**, respectively. For comparisons at 8192 tokens (and KeyB) and 130 blocks, significance is measured against **BReps-130** and **BReps-TIR-130** (paired t-test on per-query scores, $p < 0.05$).

Model	Context	P@1	MAP	NDCG@8
Baseline models				
BM25	Full document	0.201	0.259	0.277
BERT-dot	110M, 512 tokens	0.600	0.733	0.798
ColBERT	110M, 512 tokens	0.586	0.723	0.790
PLMs and LLMs for long documents				
E5-Base-4K	110M, 4096 tokens	0.406^{bt}	0.576^{bt}	0.677^{bt}
BGE-M3	560M, 4096 tokens	0.877^b	0.926^b	0.944^b
BGE-M3	560M, 8192 tokens	0.906	0.945	0.959
RankLLaMA	Gemma2-2B, 2k tokens	0.527^{bt}	0.646^{bt}	0.729^{bt}
RankLLaMA	Gemma2-2B, 4k tokens	0.593^{bt}	0.695^{bt}	0.767^{bt}
RankLLaMA	Gemma2-2B, 8k tokens	0.516^{bt}	0.652^{bt}	0.734^{bt}
KeyB(BERT)$_{BM25}$	full doc	0.422^{bt}	0.586^{bt}	0.685^{bt}
Proposed: BReps and BReps-TIR				
BReps-65	2B, $\leq$4095 tokens	0.849	0.900^t	0.925^t
BReps-TIR-65	2B, $\leq$4095 tokens	0.869	0.918^b	0.938^b
BReps-130	2B, $\leq$8190 tokens	0.920	0.945	0.959
BReps-TIR-130	2B, $\leq$8190 tokens	**0.928**	**0.955**	**0.966**

MLDR-zh (Table 5) stresses multilingual and very long-document settings. BReps-TIR consistently improves over BReps at 65 blocks and substantially outperforms the single-vector Gemma2-2B baseline. With 130 blocks, BReps-TIR achieves the best results in this table, outperforming BGE-M3 at the 8k-token setting on P@1, MAP, and NDCG@8, although it uses a larger 2B backbone and incurs a larger document-side representation footprint. This suggests that block-level representation is particularly effective when relevant evidence may be deeply buried in long documents.

6 Analysis

6.1 Effectiveness–Efficiency Trade-off

Figure 1 shows that BReps-TIR improves effectiveness while keeping online latency practical in our fixed-candidate protocol, including against the newly measured single-vector Gemma2-2B reranker. This result follows from the offline/online decomposition: document block embeddings are precomputed, while online computation consists of query encoding, block similarity scoring, and a small top-k refinement module. Latency includes tokenization, host-to-device transfer, inference, and output writing, but excludes model loading and offline document embedding precomputation. On a single NVIDIA A100 GPU, BReps takes 239.63 ms/query to rerank 100 candidates on DL'19, and BReps-TIR takes 289.91 ms/query, a $1.21\times$ overhead for the refinement module. By contrast, interaction-heavy segmentation baselines require over one second per query under the same top-100 protocol, e.g., 1,189.81 ms/query for KeyB(BERT)$_{\mathrm{BM25}}$ and 1,291.75 ms/query for IDCM with a 4,096-token input cap.

6.2 Offline Representation Computation

We also measure the offline cost of computing and materializing document representations. For offline representation computation, we use 5,000 TREC DL'22 documents because MS MARCO v2 documents are substantially longer than MS MARCO v1 documents. With Gemma2-2B, the single-vector control requires 73.0 minutes to encode up to 4,096 tokens per document, whereas BReps requires 23.4 minutes to encode up to 65 blocks under the same 4k-token budget, reducing offline computation time by 67.9%. This improvement is expected because self-attention cost grows superlinearly with sequence length; processing a long document as multiple shorter blocks avoids

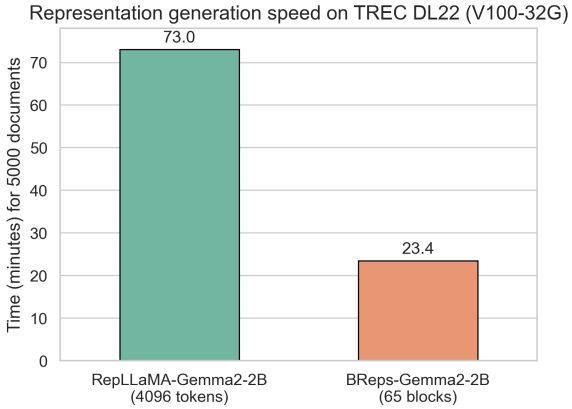

Figure 3: Time to compute representations for 5,000 TREC DL'22 documents on one NVIDIA V100 32GB GPU. The single-vector Gemma2-2B control encodes up to 4,096 tokens; BReps encodes up to 65 blocks under the same 4k-token budget.

repeatedly attending over a single long sequence.

6.3 Embedding Footprint

Beyond latency, we estimate the document representation footprint under fp16 storage. For a single-vector reranker with stored embedding dimension H, the per-document storage is

$$\text{Bytes/doc} = 2H. \tag{14}$$

For BReps, each document stores b_d block embeddings after applying the evaluation cap, yielding

$$\text{Bytes/doc} = 2b_dH. \tag{15}$$

We report the sampled average $\bar{b}$ and corpus-level extrapolations in Table 6. These are not full-corpus materialized indexes; they are estimates from the unique documents appearing in the fixed reranking candidate files, with the number of sampled documents shown explicitly in the table. For MS MARCO v2, we merge the DL'22 and DL'23 candidate samples because they share the same corpus. The observed average is about 47 stored blocks per document, below the 65-block worst case but still much larger than a single-vector index. With Gemma2-2B ($H = 2304$), this corresponds to roughly 2.37 TiB of fp16 vector payload for the 11.96M-document MS MARCO v2 corpus, or 1.18 TiB with int8 vector storage. For online reranking, fetching the fp16 vectors for the top-100 candidates requires about 20.7 MiB/query on DL'22/DL'23, compared with 0.44 MiB/query for a single-vector fp16 index.

This estimate deliberately uses the true stored encoder dimension H, not the TIR projection dimension $d = 256$. The projection dimension only affects the small online TIR module, whereas the offline index stores block embeddings before this projection. Thus BReps trades a substantially larger document-side footprint and vector-fetch bandwidth for lower online model computation and more localized evidence. MS MARCO v2 should be viewed as the large-corpus stress case in our evaluation: the storage multiplier is manageable for smaller corpora such as MLDR-zh, but at MS MARCO v2 scale practical deployment would require quantization, caching, or dimensionality reduction. Cross-encoder-style baselines such as IDCM and KeyB make the opposite trade-off: they store little document-side state but primarily incur cost at query time by scoring query-text pairs.

6.4 Segmentation Strategy

Table 7 shows that CogLTX segmentation outperforms fixed-length segmentation on NDCG@10. This supports the intuition that block boundaries matter: naive windows may split coherent evidence or mix relevant and irrelevant content. The result also suggests that the gains are not merely from using more document chunks, but from exposing semantically coherent local evidence to the top-block scorer.

6.5 Aggregation Function

Table 8 compares descending weights with average and learned weights. Average aggregation underperforms because it treats lower-confidence blocks as equally important. Learned weights do not improve consistently, likely because the small training set does not provide enough signal to learn stable rank-position weights. The fixed descending schedule is therefore a simple and robust default. We also checked standard pooling over the same Gemma2-2B block scores: AvgP obtains 0.553 NDCG@10 and MaxP obtains 0.669 on DL'19, both below weighted Top-k aggregation at 0.683. Thus the gain is not simply from splitting documents into many chunks; the document score benefits from retaining several high-confidence local matches while down-weighting weaker blocks.

6.6 Block Budget and TIR

Figure 4 varies the number of processed blocks on DL'19 and DL'23. The trend confirms that longer

Table 6: Estimated document-side vector storage for BReps. Average block counts are measured from sampled unique documents in the reranking candidate files using the same tokenizer, segmentation, and token caps as evaluation, then extrapolated to the corpus sizes in Table 1. DL'22 and DL'23 are merged because both use MS MARCO v2; separate rows would only reflect different query-specific candidate samples. Storage counts vector payloads only and excludes metadata/ANN overhead. "Top-100 read" estimates fp16 block-vector bytes fetched per query for reranking 100 candidates.

Corpus / sample source	Corpus docs	Sample docs	Avg. blocks /doc	Cap	At cap	Single fp16 TiB	BReps fp16 TiB	BReps int8 TiB	Top-100 read MiB/query
MS MARCO v1 (DL'19 cand.)	3.21M	4,295	36.5	65	25.7%	0.014	0.491	0.246	16.0
MS MARCO v2 (DL'22/23 cand.)	11.96M	5,000	47.2	65	40.8%	0.050	2.366	1.183	20.7
MLDR-zh-65	0.20M	5,000	65.0	65	100.0%	0.001	0.055	0.027	28.6
MLDR-zh-130	0.20M	5,000	124.1	130	53.8%	0.001	0.104	0.052	54.5

Table 7: Segmentation ablation on TREC DL'19. CogLTX segmentation is compared with fixed-length block segmentation.

block length	NDCG@10	MAP
default (CogLTX)	0.684	0.272
50	0.656	0.274
60	0.672	0.270
66	0.675	0.271
80	0.665	0.270
100	0.676	0.267

Table 8: Aggregation ablation on TREC DL'19. Descending top-k weights are more robust than uniform average or learned weights in this setup.

	NDCG@10	MAP
BReps-20Blocks-default	0.684	0.272
BReps-20Blocks-avgWeight	0.671	0.271
BReps-20Blocks-LearnWeight	0.664	0.269
BReps-TIR-20Blocks-default	0.691	0.273
BReps-TIR-20Blocks-avgWeight	0.682	0.271
BReps-TIR-20Blocks-LearnWeight	0.679	0.274

documents benefit from exposing more local evidence, while TIR provides an additional gain at the same block budget.

6.7 TIR Component Analysis

TIR is designed to make a limited correction rather than replacing block similarity scores. Ablations in Tables 8 and 9, together with the main results, show that BReps-TIR improves over BReps across datasets. The attention component provides a query-conditioned view of the selected block set, while the score gate lets TIR condition its correction on the strength of the original score. Removing either component hurts DL'19 NDCG@10, suggesting that setwise context and score-aware calibration are complementary. Measured in PyTorch, TIR contains 3.03M parameters, about 0.15% of the Gemma2-2B encoder, so the refinement head remains small relative to the backbone and only runs on the selected top-k blocks.

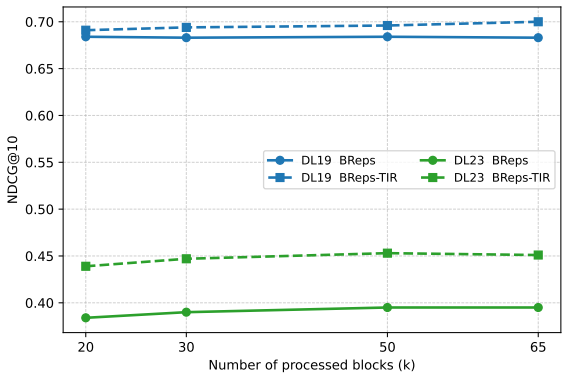

Figure 4: Effect of block budget on NDCG@10. Increasing the number of processed blocks generally improves effectiveness, and BReps-TIR consistently improves over BReps at matched budgets.

Table 9: Core TIR ablation on TREC DL'19. Δ is relative to full TIR.

Variant	NDCG@10	Δ
Full TIR	0.691	–
w/o attention context	0.676	-0.015
w/o score gate	0.672	-0.019

7 Conclusion

We presented BReps-TIR, a long-document reranking framework that represents long documents with block-level embeddings and calibrates top-block scores with lightweight top-k interaction refinement. Across TREC DL and MLDR-zh, BReps-TIR improves over single-vector LLM rerankers and trades a larger document-side vector store for lower online model computation and stronger localized evidence modeling. The results suggest that block-level document representations, combined with small setwise refinement over only the most salient blocks, are a promising direction for scalable long-document reranking.

Limitations

BReps-TIR reduces online model computation by precomputing document block vectors, but this shifts cost to document-side storage and vector fetch. On the merged DL'22/DL'23 candidate sample, our estimate corresponds to a roughly 47× fp16 vector-payload multiplier over a single-vector index, or about 2.37 TiB for the 11.96M-document MS MARCO v2 corpus before metadata and ANN-index overhead. Large-scale deployment would therefore require compression, quantization, caching, or projection to preserve the latency profile reported here. Our experiments also focus on fixed-candidate reranking rather than first-stage retrieval or end-to-end RAG, and broader comparisons with larger or commercial listwise rerankers remain useful future work under identical candidate sets and latency accounting.

References

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 2318–2335.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, Jimmy Lin, Ellen M Voorhees, and Ian Soboroff. 2025a. Overview of the trec 2022 deep learning track. *arXiv preprint arXiv:2507.10865*.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Hossein A. Rahmani, Daniel Campos, Jimmy Lin, Ellen M. Voorhees, and Ian Soboroff. 2025b. Overview of the trec 2023 deep learning track. *Preprint*, arXiv:2507.08890.

Ming Ding, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Cogltx: Applying bert to long texts. *Advances in Neural Information Processing Systems*, 33:12792–12804.

Sebastian Hofstätter, Bhaskar Mitra, Hamed Zamani, Nick Craswell, and Allan Hanbury. 2021. Intra-document cascading: Learning to select passages for neural document ranking. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1349–1358.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Minghan Li, Eric Gaussier, Juntao Li, and Guodong Zhou. 2024. Evirerank: Adaptive evidence construction for long-document llm reranking. *arXiv preprint arXiv:2411.06254*.

Minghan Li, Diana Nicoleta Popa, Johan Chagnon, Yagmur Gizem Cinar, and Eric Gaussier. 2023. The power of selecting key blocks with local pre-ranking for long document information retrieval. *ACM Transactions on Information Systems*, 41(3):1–35.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016)*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Sha Yuan, Hanyu Zhao, Zhengxiao Du, Ming Ding, Xiao Liu, Yukuo Cen, Xu Zou, Zhilin Yang, and Jie Tang. 2021. Wudaocorpora: A super large-scale chinese corpora for pre-training language models. *AI Open*, 2:65–68.

Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji-Rong Wen. 2024. Dense text retrieval based on pretrained language models: A survey. *ACM Transactions on Information Systems*, 42(4):1–60.

Dawei Zhu, Liang Wang, Nan Yang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2024. Longembed: Extending embedding models for long context retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 802–816.

A Reproducibility and Protocol Details

A.1 Training and Evaluation Configuration

Table 10 summarizes the experimental configuration used for the controlled comparison between the single-vector reranker, BReps, and BReps-TIR. The block-level variants and the controlled single-vector reranker share the Gemma-2-2B backbone family, LoRA configuration, similarity scaling, pairwise loss, and training budget. This design reduces confounding effects from model capacity or supervision size when assessing the value of block-level representations and top-k refinement.

A.2 Budget-Capped Block Store and Online Scoring

Our evaluation concerns fixed-candidate reranking under a specified document budget. For a candidate document d, CogLTX segmentation first produces a sequence of blocks $b_1, \ldots, b_n$. At document-indexing time, we materialize representations for the budget-capped prefix of blocks,

$$\mathcal{B}_d^{(K)} = \{\mathbf{v}_{b_i}\}_{i=1}^{\min(n,K)}, \qquad K \in \{65, 130\}, \tag{16}$$

where $K = 65$ is used for the standard 4k setting and $K = 130$ is used for the extended MLDR-zh 8k setting. At query time, the query is encoded once, its similarity with each stored block of a candidate document is computed, and the strongest local matches are aggregated.

This protocol is important for interpreting the reported results. BReps and BReps-TIR preserve localized evidence within a matched input budget, but they do not claim access to evidence beyond the stored budget-capped prefix. Their efficiency gain arises because the block vectors are computed offline and only vector scoring plus lightweight top-block calibration is performed online.

A.3 Controlled Single-Vector Comparison and Accounting

The controlled single-vector reranker uses the same Gemma-2-2B backbone family, LoRA configuration, learning rate, training triplet budget, and pairwise objective as the corresponding block-level run, but stores one document representation rather than multiple block representations. We additionally evaluate this model with longer document contexts at inference time to test whether a larger single-vector input window alone closes the gap to local-evidence reranking.

Online latency is measured for reranking the fixed candidate set of one query. In the TREC DL'19 latency experiment, each query has 100 candidate documents. The reported runtime includes tokenization, host-to-device transfer, neural inference, query–block similarity computation, TIR computation when applicable, score aggregation, and output writing. It excludes model loading and offline document representation computation.

Storage estimates count vector payloads only. Under fp16 storage, a single-vector reranker with output dimension H stores $2H$ bytes per document, whereas BReps stores

$$\text{Bytes/doc} = 2\bar{b}H, \tag{17}$$

where $\bar{b}$ is the sampled average number of stored blocks after applying the budget cap. For Gemma-2-2B, $H = 2304$. The TIR projection dimension $d = 256$ affects only the online refinement head and does not reduce the stored block-vector dimension. Metadata, block offsets, ANN-index structures, and compression overhead are not included in the payload estimates.

B Additional Details of Top-k Interaction Refinement

B.1 Selection–Calibration Boundary

TIR is a calibration module over already selected local evidence rather than an additional selector. Let $\pi = (\pi_1, \ldots, \pi_k)$ be the indices of the top-k blocks sorted by their original similarities, with $s_{\pi_1} \geq \cdots \geq s_{\pi_k}$. The selected set and its original order are held fixed during refinement:

$$\pi = \text{TopKSort}(s_1, \ldots, s_n), \tag{18}$$

$$\mathbf{B} = [\mathbf{v}_{b_{\pi_1}}; \ldots; \mathbf{v}_{b_{\pi_k}}] \in \mathbb{R}^{k \times H}, \tag{19}$$

$$s'_{\pi_i} = s_{\pi_i} + \delta_i, \tag{20}$$

$$S_{\text{TIR}}(q, d) = \sum_{i=1}^{k} w_i s'_{\pi_i}. \tag{21}$$

The aggregation weights satisfy $w_1 \geq \cdots \geq w_k$ and $\sum_i w_i = 1$, exactly as in the base BReps aggregation. Thus, TIR modifies the contributions of the selected blocks through bounded score corrections;

Table 10: Training, inference, and evaluation configuration of the controlled experiments.

Configuration item	Setting
Encoder backbone	Gemma-2-2B for the controlled single-vector reranker, BReps, and BReps-TIR.
Parameter-efficient tuning	LoRA with rank $r = 32$ and scaling $\alpha = 64$; mixed-precision training on one NVIDIA V100 32GB GPU.
Optimization	Learning rate 5×10^{-5}; batch size 1 with gradient accumulation of 4.
Objective	Pairwise hinge loss $\mathcal{L} = \max(0, m - S(q, d^+) + S(q, d^-))$, with margin $m = 10$ in the scaled-score space.
Similarity scaling	L2-normalized cosine similarity divided by $\tau_s = 0.01$.
TREC DL training data	20,000 (q, d^+, d^-) triplets; negatives are uniformly sampled from non-positive documents.
MLDR-zh training data	10,000 triplets following the official training files; one positive and the first labeled negative are used for each query.
Input limits	Query length capped at 32 tokens; CogLTX segmentation with a maximum block length of 63 tokens.
Block budgets	Training uses $k = 20$ blocks; the standard evaluation setting processes up to 65 blocks (≤ 4095 tokens); MLDR-zh additionally reports up to 130 blocks (≤ 8190 tokens).
TIR configuration	Projection dimension $d = 256$, attention temperature $\tau_a = 0.07$, and bounded residual scale $\gamma = 0.3$.
TREC DL protocol	Reranking the same top-100 candidate documents for each query; metrics are NDCG@10 and MAP.
MLDR-zh protocol	Reranking the official set of one positive and seven negatives per query; metrics are P@1, MAP, and NDCG@8.
Efficiency hardware	Online reranking latency is measured on one NVIDIA A100 GPU; offline document encoding is measured separately on one NVIDIA V100 32GB GPU.

it neither retrieves additional blocks nor reorders the selected blocks after calibration.

B.2 Query-Conditioned Setwise Calibration

Let $\mathbf{v}_q \in \mathbb{R}^H$ be the query representation and $\mathbf{B} \in \mathbb{R}^{k \times H}$ the selected block matrix. TIR first summarizes the selected evidence with query-to-block attention in a low-dimensional space:

$$\hat{\mathbf{q}} = W_q^a \mathrm{LN}(\mathbf{v}_q) \in \mathbb{R}^d, \tag{22}$$

$$\hat{\mathbf{B}} = \mathrm{LN}(\mathbf{B})(W_b^a)^\top \in \mathbb{R}^{k \times d}, \tag{23}$$

$$\boldsymbol{\alpha} = \mathrm{softmax}\left(\frac{\hat{\mathbf{B}}\hat{\mathbf{q}}}{\sqrt{d}\tau_a}\right), \tag{24}$$

$$\mathbf{c} = \sum_{i=1}^{k} \alpha_i \mathrm{LN}(\mathbf{v}_{b_{\pi_i}}). \tag{25}$$

For each selected block, the query vector, the local block vector, the query-conditioned context, and the original similarity score are fused:

$$\mathbf{z}_i = \tanh\left(W_q \mathrm{LN}(\mathbf{v}_q) + W_b \mathrm{LN}(\mathbf{v}_{b_{\pi_i}}) + W_c \mathrm{LN}(\mathbf{c}) + g(s_{\pi_i})\right), \tag{26}$$

where $g(\cdot)$ is a shared score-gating MLP implemented with a Linear–SiLU–Linear transformation. A final projection predicts a bounded residual:

$$\delta_i = \gamma \tanh(\mathbf{w}_\delta^\top \mathbf{z}_i), \qquad |\delta_i| \leq \gamma, \tag{27}$$

$$s'_{\pi_i} = s_{\pi_i} + \delta_i. \tag{28}$$

The residual bound ensures that the original query–block similarity remains the dominant scoring signal and that TIR acts as a local calibration head rather than an unconstrained replacement ranker.

B.3 Online Inference Procedure

For each query and candidate document, online scoring proceeds in the following sequence:

1. Encode the query once to obtain $\mathbf{v}_q$.
2. Read the precomputed, budget-capped block vectors for the candidate document.
3. Compute scaled cosine similarities between the query representation and stored block representations.
4. Select the top-k blocks and record their original similarity-based order.
5. Apply query-conditioned TIR over the selected block set to obtain bounded residuals.
6. Add each residual to its corresponding original block score, without changing the selected set or order.
7. Apply the same descending aggregation rule as BReps to obtain the document-level score.

The shared blockwise projections and the setwise attention do not depend on a fixed positional parameter for a particular value of k. Consequently, the model trained with $k = 20$ can be directly applied to larger inference-time block sets in the budget analyses. Measured on TREC DL'19 under the top-100 protocol, TIR increases latency from 239.63 to 289.91 ms/query, an absolute increment of 50.28 ms/query and a $1.21\times$ latency ratio. The refinement head contains 3.03M parameters, approximately 0.15% of the Gemma-2-2B encoder.

C Qualitative Visualization of Block-Level Representations

Figure 5 provides a qualitative view of the representation granularity induced by BReps. We randomly compute representations for 500 query–relevant-document pairs from the MS MARCO v1 document training set and display three representative query–document pairs. Query, single-vector document, and block-level document representations are projected to two dimensions using t-SNE (van der Maaten and Hinton, 2008).

In a single-vector reranker, all information in a long document must be compressed into one representation. When only a small portion of the document is relevant, this representation can mix useful evidence with unrelated content. By contrast, BReps yields multiple block representations, allowing a query to match strongly with localized evidence without requiring the entire document to be mapped close to the query.

The visualization is intended as an illustrative analysis of representation granularity, not as a quantitative proof of ranking superiority: two-dimensional projections may distort distances and do not directly measure retrieval effectiveness. It complements the controlled ranking experiments in the main paper by showing why block-level matching can preserve query-aligned local signals that a single document vector may obscure.

D Evidence Attribution and Relative TIR Calibration

TIR produces an inspectable scoring trace because it operates on selected evidence blocks and adds bounded residual corrections to their scores. For a relevant document d^+ and a nearby competing document d^-, we define the ranking-boundary margins before and after refinement as

$$M_{\text{before}} = S_{\text{BReps}}(q, d^+) - S_{\text{BReps}}(q, d^-), \quad (29)$$

$$M_{\text{after}} = S_{\text{TIR}}(q, d^+) - S_{\text{TIR}}(q, d^-). \quad (30)$$

A relevant document can improve in rank even when its absolute score decreases, provided that TIR decreases the competing document's score more strongly. Tables 11 and 12 report representative rank-boundary corrections on TREC DL'19.

These examples clarify two complementary forms of interpretability in BReps-TIR. First, BReps localizes the text blocks that contribute to a document score. Second, TIR reveals how the rank boundary changes through score calibration among close competitors. This is not a free-form rationale generated after ranking; it is an auditable trace of the evidence blocks and score changes used by the reranking pipeline.

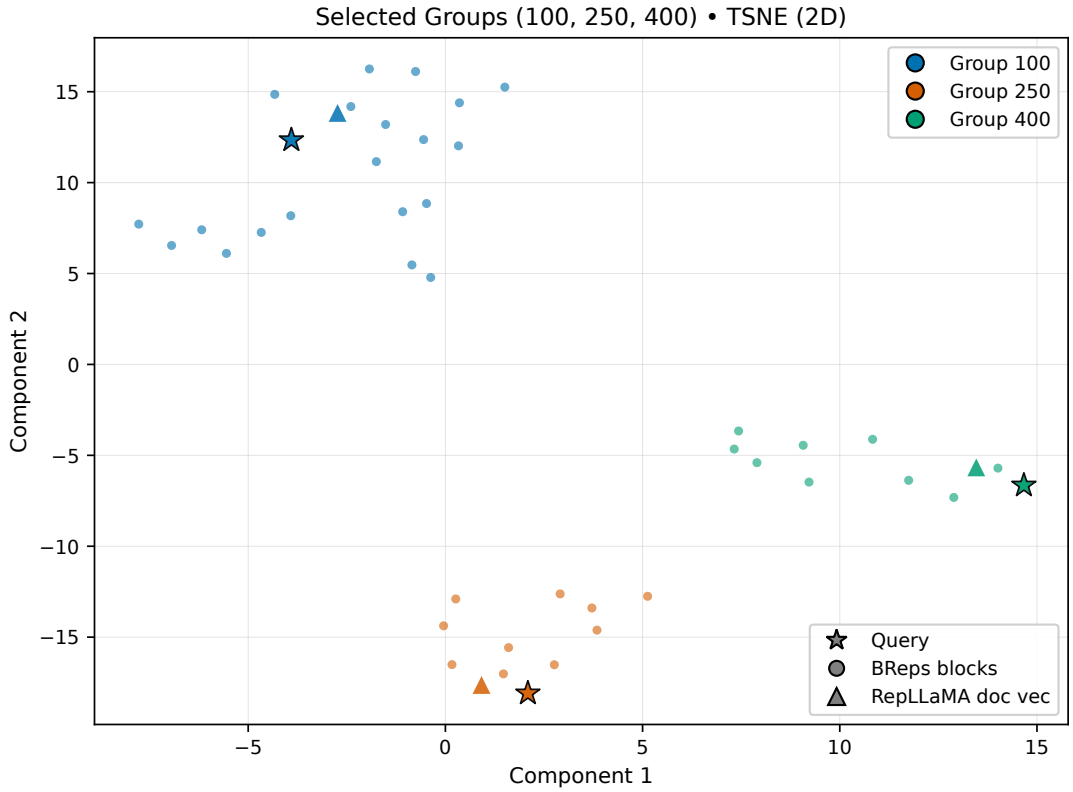

Figure 5: Qualitative visualization of three query–relevant-document pairs from MS MARCO v1. Representations are projected from the original embedding space to two dimensions using t-SNE. A single-vector document encoder yields one coarse document point, whereas BReps exposes multiple block-level points; several block representations are located closer to the corresponding query and occupy different regions of the projected space, illustrating localized and potentially complementary matching signals.

Table 11: Representative rank corrections induced by TIR on TREC DL'19. Scores are shown as base BReps score → calibrated BReps-TIR score.

Query	Relevant document	Nearest competitor	Margin change	Interpretation
medicare's definition of mechanical ventilation	D1209401: #11 → #10; 76.62 → 76.38	D1667011: #10 → #11; 76.62 → 76.31	0.00 → +0.06	The Medicare-focused document overtakes a terminology-heavy ventilation competitor.
difference between rn and bsn	D1086662: #17 → #16; 68.00 → 67.75	D3034044: #16 → #17; 68.00 → 67.69	0.00 → +0.06	Direct RN-to-BSN evidence is retained above more diffuse nursing-degree content.
how to find the midsegment of a trapezoid	D1783252: #45 → #44; 64.00 → 63.72	D318113: #44 → #45; 64.00 → 63.69	0.00 → +0.03	Formula-bearing trapezoid evidence narrowly overtakes triangle-midsegment evidence.
how long is life cycle of flea	D3208112: #4 → #3; 80.50 → 80.19	D1988715: #3 → #4; 80.50 → 80.12	0.00 → +0.07	Flea life-cycle evidence overtakes remedy- and control-oriented content.

Table 12: Evidence-level traces for the rank corrections in Table 11. Block snippets are shortened for readability.

Query	Relevant-document evidence	Competitor evidence	Evidence-level interpretation
medicare's definition of mechanical ventilation	D1209401. B25: patient-safety problems and hospital expenses. B23: prospective payment and imperfect risk adjustment. B21: coding infections and conditions as "present on admission."	D1667011. B3: assisted ventilation and artificial respiration. B37: intermittent mandatory ventilation. B6: airway pressure and barotrauma.	The competitor contains strong mechanical-ventilation terminology, but the relevant document better matches the Medicare-policy aspect of the query. TIR suppresses the competitor more strongly and changes the rank order.
difference between rn and bsn	D1086662. B0: "RN to BSN Majors Guide." B1: typical RN-to-BSN students are registered nurses employed full- or part-time. B50: nursing organizations and certification references.	D3034044. B48: associate and baccalaureate degree holders can sit for NCLEX-RN. B35: professional values for the BSN nurse. B40: nursing philosophy across degrees.	The relevant document provides direct RN-to-BSN program evidence, whereas the competitor is relevant to nursing degrees but less focused on the requested distinction.
how to find the midsegment of a trapezoid	D1783252. B29: $MN \parallel AB$, $MN \parallel DC$, and $MN = \frac{1}{2}(AB + CD)$. B21 and B24: supporting trapezoid relations.	D318113. B1: a triangle midsegment joins the midpoints of two sides. B12 and B15: triangle-midsegment examples and calculations.	The relevant document contains the exact trapezoid midsegment formula, while the competitor's evidence concerns a related but different geometric object.
how long is life cycle of flea	D3208112. B9: Oriental rat flea and flea species. B3: flea feeding on mammals and birds. B24: flea maturation and reproduction around the host nest.	D1988715. B17: over-the-counter flea remedies. B38: chemical flea and tick control. B9: flea jumping ability.	Both documents discuss fleas, but the relevant document provides a more coherent biological evidence chain, while the competitor emphasizes treatment and general descriptions.