LEARNING WHAT MATTERS: REWEIGHTING FROZEN EMBEDDINGS FOR DENSE RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Dense retrievers enable efficient semantic search using query and document embeddings. Once deployed, however, improving ranking quality of a dense retriever requires careful balancing of its performance with the cost of changing existing retrieval infrastructure. This motivates adapting how retrieved documents are scored while retaining the encoder, stored embeddings, and index. Yet fixed similarity scoring does not adapt dimension weights to the candidates competing for relevance. We introduce Shared Signed Gate (SSG), a lightweight reranker that learns which embedding dimensions help distinguish relevant documents from retrieved hard negatives. Using explicit dimension-level supervision, a small network predicts candidate-conditioned residual weights from the query and candidate statistics. Across nine encoders and seven BEIR datasets, SSG improves average nDCG@10 for every encoder and achieves the highest average among evaluated baselines for all three encoders used in detailed comparisons. It also improves rankings produced by other retrieval methods. With BGE-S, SSG adds approximately 7.8% to measured end-to-end latency while preserving the encoder, corpus embeddings, and retrieval index. These results show that candidate-conditioned dimensional weighting can improve ranking quality while reusing the representations and retrieval infrastructure already in place.

1 INTRODUCTION

Dense retrieval methods (Karpukhin et al., 2020; Xiong et al., 2021; Izacard et al., 2022) that map queries and documents to vectors are becoming very effective, but the best embedding models can depend on the retrieval domain and relevance definition (Thakur et al., 2021). For an existing retrieval system, improving ranking quality through a new embedding model can also require re-encoding the corpus and rebuilding its index. This makes better use of already-computed embeddings a practically important alternative. Simple dot-product scoring uses a uniform weighting of all the coordinates of the embeddings, and this may not be optimal for distinguishing between relevant and hard negative documents. However, the most useful dimensions to adapt may depend on the query and candidates.

Cross-encoders and late-interaction models capture richer query-document interactions at additional computational or storage cost (Nogueira & Cho, 2019; Khattab & Zaharia, 2020). Another direction adapts pretrained embeddings directly: Search-Adaptor learns representation transformations, while IMRNNs use query- and feedback-conditioned modulation (Yoon et al., 2024; Saxena et al., 2026). These trade-offs motivate a concrete question: *how can we improve discrimination among retrieved candidates while retaining the existing embeddings and index and adding little query-time computation?*

Retrieval feedback (e.g. relevance or pseudo-relevance) provides useful information about dimensional weighting adjustment. Recent approaches, such as Dimension Importance Estimation, exploit pseudo-relevant documents to determine which coordinates carry significant information (Faggioli et al., 2024; Campagnano et al., 2025). Similarly, ECLIPSE leverages feedback from pseudo-irrelevant documents to find noisy dimensions (D'Erasmo et al., 2025). Training relevance judgments provide direct evidence of which dimensions distinguish relevant documents from retrieved hard negatives. Learning to predict this evidence from the query and candidate set makes relevance-informed weighting available at inference, when those judgments are absent.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We introduce SHARED SIGNED GATE (**SSG**), a lightweight second-stage reranker operating on frozen query and document embeddings. SSG constructs a dimensional training target from differences in mean query-document contributions between relevant documents and retrieved hard negatives. A small network learns to predict this target from the query embedding and dimension-wise candidate statistics, connecting relevance-derived supervision to the local retrieval context. At inference, the predicted signed gate determines residual weights shared across the candidate list, allowing dimensions to be amplified or attenuated continuously. This adapts coordinate contributions without changing the stored embeddings or first-stage index. SSG therefore confines adaptation to scoring the retrieved candidate set: it requires neither corpus-wide embedding updates nor a second retrieval pass.

We evaluate separately trained SSG predictors across 9 encoders on 7 BEIR datasets and show that it consistently improves the average nDCG@10 of every encoder. We also compare with embedding-adaptation and feedback methods, as well as various controlled ablations. Finally, we show SSG can improve candidates retrieved by other methods.

Our main contributions are as follows:

- We introduce a candidate-conditioned reranking framework that predicts a shared residual gate over embedding dimensions, adapting the scoring function while preserving the frozen encoder and first-stage index.
- We derive explicit dimension-level supervision from relevant-hard-negative contribution contrasts and train a lightweight predictor to estimate these targets from the query and candidate summaries available at inference.
- Across nine encoders and seven BEIR datasets, we demonstrate improved average ranking quality for every encoder, and evaluate the inference cost and compatibility of this adaptation with other retrieval methods.

2 RELATED WORK

Dense Retrieval and Lightweight Reranking: Dense retrieval employs bi-encoders that map queries and documents separately to the shared vector space and pre-compute the document embeddings, as DPR for open-domain retrieval, ANCE and RocketQA with negative sampling and denoised supervision (Karpukhin et al., 2020; Xiong et al., 2021; Qu et al., 2021). While a cross-encoder can match its query and document by jointly encoding query-document pairs, it is computationally expensive. Late-interaction models like ColBERT and ColBERTv2 do not need repeated encoding to boost the interaction at token level (Khattab & Zaharia, 2020; Santhanam et al., 2022). Other solutions empirically improve retrieval and reranking via joint-training, distillation or decomposable architecture (Ren et al., 2021; Liao et al., 2024). Our SSG focuses on the aspect of lightweight reranking in the second stage over existing single-vector embeddings while keeping the encoder and first-stage index unchanged.

Embedding Adaptation and Dimension-Level Modulation: Post-hoc adaptation methods modify the retrieval process, without fully fine-tuning the embedding model. Examples include Search-Adaptor, which learns a supervised transformation of the frozen embeddings (Yoon et al., 2024); relevance filtering, which maps cosine similarities to query-dependent scores as a means of filtering out irrelevant mass (Rossi et al., 2024); and representation sharpening, which uses information from similar documents in the corpus to make document representations more distinguishable (Ashok et al., 2026).

A related avenue is adapting the individual coordinates in the embedding. For example, Matryoshka Representation Learning learns an embedding with information packed into a shorter prefix for the purpose of reducing its dimensionality (Kusupati et al., 2022). Dimension Importance Estimation identifies query-dependent dimensions that are particularly relevant given pseudo-relevant documents, leading to further questions regarding its generality (Faggioli et al., 2024; Campagnano et al., 2025). ECLIPSE instead uses pseudo-irrelevance feedback to mitigate noise in the selected dimensions (D'Erasmo et al., 2025). SSG shares the motivation of query-dependent adaptation at the level of individual dimensions, but differs in supervision and retrieval operation: it constructs continuous residual-weight targets from training relevance judgments and learns to predict them

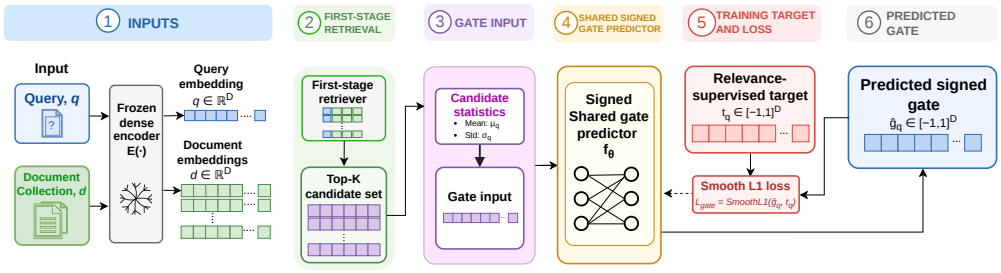


Figure 1: Overview of SSG gate prediction and training. A frozen retriever supplies query embeddings and top-K candidates, whose statistics condition a shared gate predictor. The predicted gate is supervised by a relevance-derived dimensional target using Smooth L1 loss.

from the query and candidate summaries, whereas DIME and ECLIPSE derive dimensional signals from pseudo-feedback at retrieval time without supervised training.

Candidate-Conditioned and Feedback-Based Retrieval: In pseudo-relevance feedback (PRF), the initially retrieved documents are used to improve upon later retrieval decisions. Neural pseudo-relevance feedback methods include NPRF (Li et al., 2018) and ANCE-PRF (Yu et al., 2021), which aggregate feedback signals to build refined query representations, while ColBERT-PRF (Wang et al., 2023) leverages retrieved passages for multi-vector retrieval using representative token embeddings. Other techniques rely on the use of cross-encoder pseudo-labels (Sung et al., 2023), explicit relevance feedback (Baumgärtner et al., 2022), or modelling the neighbourhood relationships (Zerveas et al., 2023).

CMC (Song et al., 2024) and IMRNNs (Saxena et al., 2026) explored approaches to capture candidate context through shallow self-attention over query and candidate embeddings, and query-conditioned document modulation with feedback-conditioned query refinement, respectively. In contrast, SSG summarizes the retrieved candidate set using permutation-invariant, dimension-wise mean and standard-deviation statistics and uses them to predict one shared gate for the entire candidate list. This avoids candidate-wise self-attention over the retrieved set and, unlike IMRNNs, does not require corpus-wide representation transformation or an additional retrieval index.

3 Methodology

SSG learns to adjust the contribution of embedding dimensions while keeping the dense retriever frozen. Figure 1 illustrates gate prediction and training, with one relevance-derived dimensional target constructed for each training query. We explicitly create these targets by assessing which dimensions contribute to distinguishing relevant candidates from retrieved hard negatives. We then train a lightweight predictor to estimate the target using the query embedding and candidate summary statistics. At inference time, we use the predicted gate to apply the same residual weighting to the scoring of all retrieved candidates. This allows us to keep the encoder and first-stage index frozen.

3.1 Problem Formulation

Let $E(\cdot)$ denote a frozen dense encoder that maps a query q and document d to L_2-normalized embeddings $\mathbf{q} = E(q) \in \mathbb{R}^D$ and $\mathbf{d} = E(d) \in \mathbb{R}^D$. The first-stage retriever scores documents using

$$s_0(q, d) = \mathbf{q}^\top \mathbf{d} = \sum_{j=1}^{D} q_j d_j, \tag{1}$$

which is equivalent to cosine similarity for these normalized representations. Given a document collection $\mathcal{D}$, it returns the top-K candidate set

$$\mathcal{C}_q = \mathrm{TopK}_{d \in \mathcal{D}}\ s_0(q, d). \tag{2}$$

Our objective is to improve the ordering within $\mathcal{C}_q$ while preserving its membership, the stored embeddings, and the retrieval index.

Equation 1 assigns a fixed, uniform weight to each coordinate-wise interaction $q_j d_j$. We instead predict a signed gate $\hat{\mathbf{g}}_q \in [-1, 1]^D$, conditioned on the query and its candidate distribution, to adjust these interactions through shared dimensional weights. The following subsections describe target construction, gate prediction, and residual scoring.

3.2 Relevance-Derived Dimensional Targets

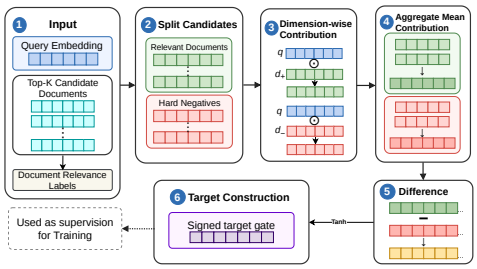


Figure 2: Dimensional target construction. Mean query-document contributions from relevant documents and hard negatives are contrasted, standardized across dimensions, and transformed by temperature-scaled $\tanh$.

We supervise this gate using dimension-level contrasts between relevant documents and hard negatives, yielding an explicit training signal regarding how much each coordinate contributes to the separation. This contrasts pairs of (potentially) useful inputs; Fig. 2 shows this construction.

For training query q, the set of documents in $\mathcal{P}_q$ are the judged relevant documents. The set of documents in $\mathcal{N}_q$ are the H highest-ranked candidates that are not labeled as relevant. These documents serve as hard negatives, since the base retriever gave them high scores. Judged relevant documents that are outside the top-K segment are used only for target construction — the gate input is computed from the original candidate set $\mathcal{C}_q$.

This vector $\mathbf{q} \odot \mathbf{d}$ consists of coordinate-wise contributions of a query-document pair to the base dot-product score. We average these contributions separately over relevant documents and hard negatives:

$$\mathbf{c}_q^+ = \frac{1}{|\mathcal{P}_q|} \sum_{d \in \mathcal{P}_q} \mathbf{q} \odot \mathbf{d}, \qquad \mathbf{c}_q^- = \frac{1}{|\mathcal{N}_q|} \sum_{d \in \mathcal{N}_q} \mathbf{q} \odot \mathbf{d}. \tag{3}$$

Their difference,

$$\boldsymbol{\Delta}_q = \mathbf{c}_q^+ - \mathbf{c}_q^-, \tag{4}$$

measures each dimension's contribution to the mean score gap between the two groups. A positive $\Delta_{q,j}$ favors relevant documents on average, whereas a negative value favors the selected hard negatives.

To reduce sensitivity to the scale of these contrasts, we standardize $\boldsymbol{\Delta}_q$ across dimensions for each query and apply a bounded transformation:

$$\mathbf{t}_q = \tanh\left(\frac{\boldsymbol{\Delta}_q - \mu_{\Delta,q}\mathbf{1}}{\tau(\sigma_{\Delta,q} + \epsilon)}\right), \tag{5}$$

where $\mu_{\Delta,q}$ and $\sigma_{\Delta,q}$ are the mean and standard deviation across the D coordinates, $\epsilon > 0$ ensures numerical stability, and $\tau > 0$ controls target sharpness. The resulting target $\mathbf{t}_q \in [-1, 1]^D$ expresses relative dimensional contrast: positive and negative values indicate above- and below-average contrasts within the query, respectively. Its sign therefore need not match that of the raw contrast $\boldsymbol{\Delta}_q$.

3.3 Candidate-Conditioned Gate Prediction

Since relevance judgments are unavailable at inference, the predictor estimates the dimensional target in Equation 5 from the query embedding and a compact summary of its retrieved candidates. For document embeddings $\mathbf{d}_1, \ldots, \mathbf{d}_K$ in $\mathcal{C}_q$, we compute the coordinate-wise mean and population standard deviation:

$$\boldsymbol{\mu}_q = \frac{1}{K}\sum_{i=1}^{K} \mathbf{d}_i, \qquad \boldsymbol{\sigma}_q = \sqrt{\frac{1}{K}\sum_{i=1}^{K} (\mathbf{d}_i - \boldsymbol{\mu}_q)^2}, \tag{6}$$

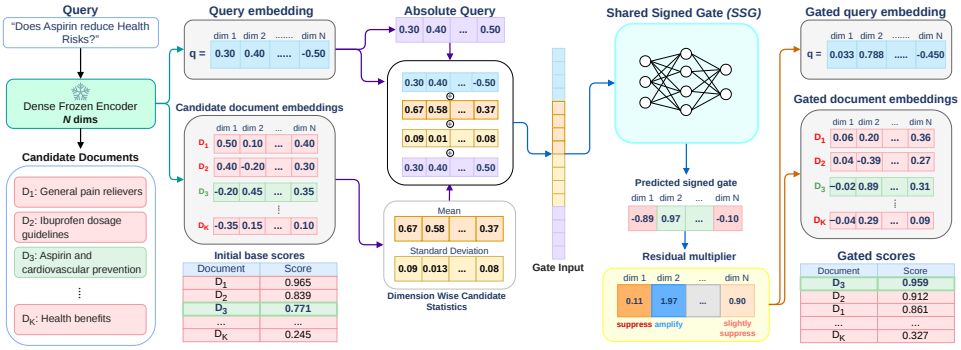


Figure 3: Illustrative SSG inference procedure. Query and candidate features condition a shared gate, whose residual weights adjust dimension contributions and reorder the retrieved candidates.

where the squaring and square root operations are done element-wise. This summary provides information about both the center and dispersion of the candidate embeddings along each coordinate. Furthermore, it is a permutation-invariant summary with dimensionality that is independent of the number of candidates.

The summary is then concatenated with the query embedding and its element-wise absolute values:

$$\mathbf{x}_q = [\mathbf{q}; \boldsymbol{\mu}_q; \boldsymbol{\sigma}_q; |\mathbf{q}|] \in \mathbb{R}^{4D}. \tag{7}$$

The absolute query embedding provides additional information about the coordinate magnitudes to complement its signed values. All of this information is passed into a lightweight multilayer perceptron f_θ with a LayerNorm, two hidden layers with GELU activations followed by dropout, and an output of dimensionality D, which, after applying $\tanh$ predicts:

$$\hat{\mathbf{g}}_q = \tanh\left(f_\theta(\mathbf{x}_q)\right), \qquad \hat{\mathbf{g}}_q \in [-1, 1]^D. \tag{8}$$

The gate predictor produces one gate that is shared for the query and the set of candidates. It takes inputs that are label-free during both training and inference.

3.4 Training Objective

We train the predictor to approximate the target derived from relevance, using a dimension-averaged Smooth L1 loss (Girshick, 2015):

$$\mathcal{L}_{\text{gate}}(q) = \frac{1}{D} \sum_{j=1}^{D} \text{SmoothL1}\left(\hat{g}_{q,j}, t_{q,j}\right). \tag{9}$$

The Smooth L1 loss penalizes small errors quadratically and larger errors linearly, limiting their influence on the loss in comparison to the squared-error loss. The objective of the training process is to minimize this loss over eligible training queries, while updating only the predictor's parameters θ, keeping the encoder, embeddings, and constructed targets fixed for a specific target temperature.

This formulation directly supervises the predicted dimensional contrasts, connecting gate learning to the contribution differences between relevant documents and hard negatives.

3.5 Residual Scoring and Inference

The predicted gate adjusts dimension weights through a residual multiplier:

$$\mathbf{m}_q = \mathbf{1} + \alpha \hat{\mathbf{g}}_q, \qquad 0 \le \alpha \le 1, \tag{10}$$

where α controls modulation strength. A zero gate value leaves the corresponding coordinate unchanged. For $\alpha > 0$, positive gate values amplify coordinates, while negative values attenuate them. Setting $\alpha = 0$ recovers the original scoring function. Applying the shared multiplier to the query

and candidate embeddings gives

$$\widetilde{\mathbf{q}} = \mathbf{q} \odot \mathbf{m}_q, \qquad \widetilde{\mathbf{d}} = \mathbf{d} \odot \mathbf{m}_q. \tag{11}$$

We score these vectors without renormalization:

$$s_{\text{gate}}(q, d) = \widetilde{\mathbf{q}}^\top \widetilde{\mathbf{d}} = \sum_{j=1}^{D} q_j d_j m_{q,j}^2. \tag{12}$$

The resulting score is a query-specific diagonal weighting of the original coordinate contributions, with effective weights $m_{q,j}^2 \geq 0$. These weights change contribution magnitudes without reversing their signs, although a zero weight removes a contribution entirely. Because the vectors are not renormalized, the score retains the direct effect of these coordinate weights. Equation 12 is also equivalent to applying $\mathbf{m}_q^2$ only to the query and taking its dot product with each original candidate embedding. Thus, two-sided application and query-only squared weighting implement the same scoring function.

At inference, we compute the candidate statistics, evaluate the gate predictor once, and sort the candidates by $s_{\text{gate}}(q, d)$. Computing the statistics and rescoring requires $O(KD)$ operations, in addition to the predictor evaluation and sorting. Figure 3 illustrates how the resulting dimension weights can change the candidate order.

4 Experimental Setup

Datasets and Evaluation. We evaluate SSG on seven BEIR datasets (Thakur et al., 2021), covering biomedical retrieval (NFCorpus and TREC-COVID), scientific claim verification (SciFact), financial question answering (FiQA-2018), argument retrieval (ArguAna and Touché 2020), and open-domain question answering (Natural Questions). Official splits are retained whenever available. When a validation split is absent, 10% of the training queries are held out for validation. Test-only collections are partitioned at the query level into 45% training, 5% validation, and 50% testing using a fixed seed. For the two very small collections, TREC-COVID and Webis-Touché 2020, we instead perform five-fold cross-validation on the development portion for model selection while keeping the test partition fixed for final evaluation. Per-dataset corpus sizes and the query counts are reported in Appendix Table 4. SSG reranks the top 100 documents returned by the frozen retriever, and effectiveness is measured using nDCG@10.

Base Embedding Models. We evaluate nine frozen dense embedding models spanning multiple model families and representation dimensionalities. Snowflake Arctic Embed-S (**Arctic-S**) and BGE-small-en (**BGE-S**) produce 384-dimensional embeddings, while Snowflake Arctic Embed-M-v2.0 (**Arctic-M**) produces 768-dimensional representations. The 1024-dimensional group comprises Qwen3-Embedding-0.6B (**Qwen3-0.6B**), GTE-large-en-v1.5 (**GTE-L**), BGE-large-en-v1.5 (**BGE-L**), and E5-large-v2 (**E5-L**). We further include Qwen3-Embedding-4B (**Qwen3-4B**) and Qwen3-Embedding-8B (**Qwen3-8B**), which produce 2560- and 4096-dimensional embeddings, respectively. Each of the nine encoders is evaluated on all seven retrieval datasets, yielding 63 model-dataset combinations and allowing us to assess SSG consistently across different model families, embedding scales, and retrieval domains. Dataset-level query and corpus statistics, including the number of queries and documents for each collection, are provided in the Appendix.

Baselines. We compare SSG with the original frozen-encoder ranking and Global Gate, a query-independent dimension weighting trained with SSG's target supervision. Published baselines include Search-Adaptor (Yoon et al., 2024), which learns residual embedding transformations; IMRNNs (Saxena et al., 2026), which use query- and feedback-conditioned modulation; and PRF-DIME (Faggioli et al., 2024; Campagnano et al., 2025) and ECLIPSE (D'Erasmo et al., 2025), which select dimensions using pseudo-relevance and contrastive feedback, respectively. We additionally evaluate a frozen-embedding CMC variant (Song et al., 2024), a simple fixed-candidate Vector PRF baseline (Yu et al., 2021), and an off-the-shelf MiniLM cross-encoder (Wang et al., 2020) evaluated zero-shot on the target BEIR collections. Implementation and tuning details are given in Appendix A.2.

Implementation Details. SSG uses fixed, precomputed embeddings and top-100 candidate lists. A separate predictor is trained for each dataset-encoder pair, with checkpoints and hyperparameters

selected using validation nDCG@10. Test results use the validation-selected configuration. Architecture, optimization, and selection settings are provided in Appendix A.3.

5 Results and Analysis

5.1 Main Results

Table 1 compares SSG against Base Retrieval, Global Gate, and seven additional retrieval and reranking baselines, using three different encoders on seven datasets. SSG improves the base retriever on all 21 combinations of encoder and dataset, and achieves the highest average across all seven datasets for each encoder. The average nDCG@10 increases from 0.519 to 0.559 for BGE-S, from 0.561 to 0.588 for Arctic-M, and from 0.564 to 0.603 for Qwen3-0.6B, exceeding the strongest competing averages of 0.526, 0.566, and 0.576, respectively.

SSG achieves the best result in 17 of the 21 settings. SSG is consistently better than the other methods on SciFact, ArguAna, FiQA, TREC-COVID, and NQ, leading for all three encoders. The four exceptions are the three NFCorpus settings and BGE-S on Touché-2020. However, SSG outperforms Global Gate in every case over the three encoders, suggesting that query-specific adaptation is more effective than dataset-level weighting. Overall, SSG demonstrates broadly effective reranking and the strongest average performance across the compared methods. Controlled ablations investigating the effect of training loss, candidate inputs, and scoring function are described in Appendix A.6.

Table 1: Test nDCG@10 across seven BEIR datasets for three representative encoders. Best and second-best results within each encoder and column are shown in **bold** and underlined, respectively, based on unrounded scores. Avg. denotes the mean across all seven datasets.

Model	Method	nDCG@10							
		NFCorpus	SciFact	ArguAna	FiQA	TREC-COVID	Touché-2020	NQ	Avg.
BGE-S	Base Retrieval	0.335	0.699	0.592	0.399	0.775	0.343	0.492	0.519
	Global Gate	0.339	0.699	0.604	0.399	0.777	0.343	0.509	0.524
	Vector PRF	0.338	0.703	0.619	0.401	0.742	0.324	0.491	0.517
	PRF-DIME	0.339	0.711	0.610	0.391	0.796	0.336	0.488	0.524
	ECLIPSE	0.344	0.704	0.558	0.391	0.647	**0.351**	0.494	0.498
	CMC	0.331	0.702	0.610	0.395	0.768	0.335	0.503	0.520
	IMRNNs	0.293	0.746	0.590	0.275	0.668	0.315	0.446	0.476
	Search-Adaptor	0.337	0.702	0.585	0.401	0.808	0.345	0.504	0.526
	MiniLM Cross-Encoder	**0.344**	0.697	0.418	0.376	0.811	0.306	0.510	0.495
	SSG (Ours)	0.343	**0.754**	**0.658**	**0.402**	**0.821**	0.350	**0.586**	**0.559**
Arctic-M	Base Retrieval	0.359	0.718	0.597	0.440	0.819	0.339	0.653	0.561
	Global Gate	0.359	0.721	0.602	0.440	0.819	0.339	0.654	0.562
	Vector PRF	0.371	0.724	0.628	0.441	0.761	0.323	0.653	0.557
	PRF-DIME	0.370	0.712	0.617	0.440	0.806	0.324	0.649	0.560
	ECLIPSE	**0.377**	0.709	0.594	0.440	0.725	0.331	0.651	0.547
	CMC	0.357	0.722	0.603	0.442	0.818	0.342	0.654	0.563
	IMRNNs	0.301	0.777	0.593	0.328	0.761	0.327	0.524	0.516
	Search-Adaptor	0.359	0.786	0.633	0.439	0.749	0.344	0.653	0.566
	MiniLM Cross-Encoder	0.351	0.692	0.417	0.376	0.837	0.311	0.599	0.512
	SSG (Ours)	0.372	**0.815**	**0.641**	**0.446**	**0.840**	**0.345**	**0.655**	**0.588**
Qwen3-0.6B	Base Retrieval	0.355	0.700	0.690	0.466	0.897	0.278	0.558	0.564
	Global Gate	0.357	0.700	0.688	0.469	0.897	0.278	0.559	0.564
	Vector PRF	0.368	0.684	0.701	0.462	0.885	0.261	0.559	0.560
	PRF-DIME	0.368	0.698	0.709	0.464	0.895	0.266	0.559	0.566
	ECLIPSE	**0.369**	0.700	0.691	0.457	0.887	0.274	0.556	0.562
	CMC	0.350	0.707	0.689	0.466	0.901	0.275	0.561	0.564
	IMRNNs	0.320	0.796	0.659	0.366	0.843	0.305	0.492	0.540
	Search-Adaptor	0.341	0.808	0.677	0.467	0.901	0.278	0.559	0.576
	MiniLM Cross-Encoder	0.353	0.700	0.420	0.383	0.863	0.320	0.561	0.514
	SSG (Ours)	0.368	**0.812**	**0.732**	**0.474**	**0.920**	**0.322**	**0.591**	**0.603**

5.2 Generality across Embedding Models

Figure 4 assesses the versatility of SSG across nine encoders, spanning multiple model families and dimensionalities (384-4096). For every encoder, SSG improves the seven-dataset average of

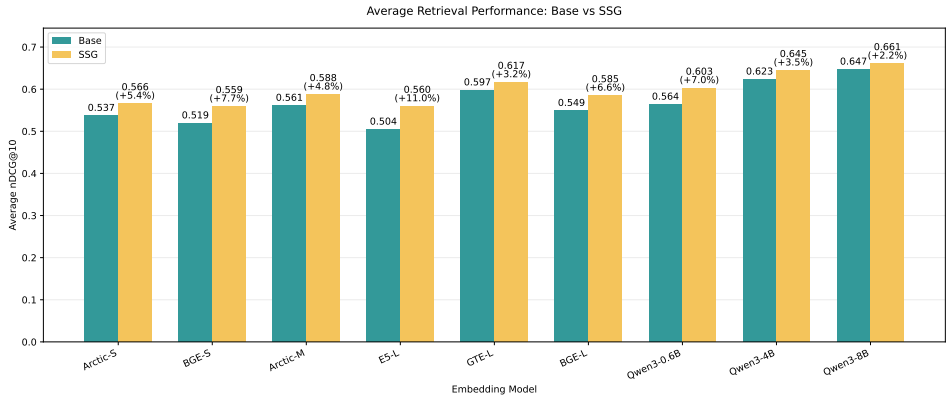


Figure 4: Average test nDCG@10 across seven BEIR datasets for nine frozen encoders, before and after SSG reranking. Percentage annotations show relative improvements over the corresponding base retriever.

nDCG@10, with gains ranging from 2.2% to 11.0%. The largest improvement is observed for E5-L, increasing from 0.504 to 0.560. Improvements also extend to stronger base retrievers: Qwen3-4B increases from 0.623 to 0.645, and Qwen3-8B from 0.647 to 0.661. No monotonicity in gains with respect to embedding dimension highlights the versatility of SSG.

Each dataset-encoder pair uses a separately trained predictor. Detailed dataset-level results are reported in Appendix A.4.

5.3 Compatibility with Other Retrieval Methods

Table 2: Test nDCG@10 before and after applying SSG to candidates retrieved by other methods, without retraining the SSG predictor. Bold indicates the higher score within each pair, including ties at three-decimal precision.

Model	Method	NFCorpus		SciFact		ArguAna		FiQA		TREC-COVID		Touché-2020		NQ	
		Orig.	+SSG	Orig.	+SSG	Orig.	+SSG	Orig.	+SSG	Orig.	+SSG	Orig.	+SSG	Orig.	+SSG
BGE-S	PRF-DIME	0.339	**0.346**	0.711	**0.729**	0.610	**0.617**	0.391	**0.395**	0.796	**0.818**	0.336	**0.339**	0.488	**0.512**
	IMRNNs	0.293	**0.338**	0.746	**0.750**	0.590	**0.611**	0.275	**0.389**	0.668	**0.808**	0.315	**0.350**	0.446	**0.508**
	Search-Adaptor	0.337	**0.342**	0.702	**0.748**	0.585	**0.612**	**0.401**	**0.401**	0.808	**0.813**	0.345	**0.350**	0.504	**0.511**
	ECLIPSE	**0.344**	0.341	0.704	**0.746**	0.558	**0.612**	0.391	**0.400**	0.647	**0.796**	**0.351**	0.350	0.494	**0.510**
Arctic-M	PRF-DIME	0.370	**0.377**	0.712	**0.774**	0.617	**0.638**	0.440	**0.445**	0.806	**0.822**	0.324	**0.335**	0.649	**0.652**
	IMRNNs	0.301	**0.368**	0.777	**0.830**	0.593	**0.629**	0.328	**0.441**	0.761	**0.838**	0.327	**0.344**	0.524	**0.648**
	Search-Adaptor	0.359	**0.371**	0.786	**0.833**	**0.633**	0.626	0.439	**0.446**	0.749	**0.845**	0.344	**0.345**	0.653	**0.654**
	ECLIPSE	**0.377**	0.372	0.709	**0.821**	0.594	**0.627**	0.440	**0.446**	0.725	**0.843**	0.331	**0.347**	0.651	**0.654**
Qwen3-0.6B	PRF-DIME	0.368	**0.373**	0.698	**0.765**	0.709	**0.716**	0.464	**0.469**	0.895	**0.915**	0.266	**0.292**	0.559	**0.561**
	IMRNNs	0.320	**0.364**	0.796	**0.830**	0.659	**0.700**	0.366	**0.467**	0.843	**0.923**	0.305	**0.318**	0.492	**0.559**
	Search-Adaptor	0.341	**0.367**	0.808	**0.832**	0.677	**0.700**	0.467	**0.476**	0.901	**0.920**	0.278	**0.319**	0.559	**0.561**
	ECLIPSE	**0.369**	0.367	0.700	**0.812**	0.691	**0.700**	0.457	**0.474**	0.887	**0.925**	0.274	**0.319**	0.556	**0.561**

Table 2 shows how well we can score candidates from PRF-DIME, IMRNNs, Search-Adaptor, and ECLIPSE using SSG. For each dataset-encoder, we take the trained SSG predictor without retraining and simply recompute the statistics for the top-100 documents retrieved by each method. SSG then assigns scores using the original frozen bi-encoder query embedding and the original frozen document embeddings corresponding to the retrieved candidate IDs; baseline-adapted or projected representations are not passed to SSG. Thus, SSG acts as a post-hoc reranker over the candidate sets produced by the alternative methods, rather than modifying their internal representation spaces.

At three-decimal precision, SSG scores improve in 78 of the 84 method-encoder-dataset comparisons, and are tied in one case. There is an improvement in all 21 comparisons of PRF-DIME and IMRNNs, in 19 of the comparisons for Search-Adaptor, and in 17 for ECLIPSE. In some encoder-

dataset combinations, using SSG even outperforms standalone SSG: the Arctic-M Search-Adaptor gives an SSG score of 0.833 on SciFact, ahead of standalone SSG at 0.815, while the Qwen3-0.6B ECLIPSE achieves 0.925 on TREC-COVID with SSG, up from 0.920.

The improvement is not universal: ECLIPSE decreases on NFCorpus for all three encoders and on Touché-2020 for BGE-S (0.351 to 0.350), while Arctic-M Search-Adaptor decreases on ArguAna from 0.633 to 0.626.

5.4 Efficiency and Computational Cost

Table 3 reports trainable parameter counts, end-to-end latency, and deployment requirements evaluated using the BGE-S encoder. We evaluate up to 200 test queries per dataset over three repeats after 10 warm-up queries and macro-average latency across the seven datasets. All retrieval outputs are matched at $K = 100$. Parameter counts exclude the frozen encoder. Higher-dimensional SSG profiling is reported in Appendix A.5.

Table 3: Efficiency and deployment requirements with BGE-S. End-to-end latency is macro-averaged across seven datasets with matched output depth $K = 100$.

Method	**Trainable params.**	**Latency (ms/query)**	**Corpus transformation**	**Additional index**
Base Retrieval	N/A	215.24	N/A	N/A
PRF-DIME (Rerank)	0	216.18	No	No
PRF-DIME (Retrieval)	0	350.79	No	No
ECLIPSE	0	351.26	No	No
CMC	4,336,384	247.30	No	No
IMRNNs	248,448	168.12	Yes	Yes
Search-Adaptor	591,360	211.18	Yes	Yes
MiniLM Cross-Encoder	0	363.22	No	No
SSG (Ours)	1,249,664	232.04	No	No

SSG has a measured latency of 232.04 ms/query, compared with 215.24 ms/query for Base Retrieval, adding 16.80 ms (approximately 7.8%). Its 1.25M trainable parameters preserve the stored corpus embeddings and retrieval index. IMRNNs (168.12 ms/query) and Search-Adaptor (211.18 ms/query) are faster online but require corpus-wide transformation and an additional index, whose offline cost is excluded. CMC reaches 247.30 ms/query with 4.34M parameters. The zero-shot MiniLM cross-encoder adds 147.98 ms/query for top-100 reranking, yielding approximately 363.22 ms/query end-to-end. PRF-DIME requires 216.18 ms/query for fixed-candidate reranking, but 350.79 ms/query with a full-corpus second search, ECLIPSE similarly requires 351.26 ms/query. Overall, SSG adds modest end-to-end overhead without corpus-wide adaptation.

6 Discussion

SSG provides a practical way to improve dense retrieval without replacing the underlying retriever, recomputing corpus embeddings, or rebuilding the retrieval index. Across nine encoders, it improves average retrieval effectiveness while operating directly on the representations already available in an existing system. Its compatibility with candidates produced by other retrieval methods further shows how the same lightweight reranking mechanism can be incorporated into existing retrieval pipelines.

The approach nevertheless requires relevance judgments and separate training for each dataset-encoder pair; transfer to unseen domains or encoders remains untested. As a top-K reranker, SSG cannot recover relevant documents missed by first-stage retrieval, and its dimensional targets do not guarantee optimal ranking weights.

Future work will explore transferable predictors, weaker supervision, and closer alignment between dimensional targets and ranking objectives. Multilingual evaluation and sensitivity to candidate depth and quality would further establish the scope of the approach.

AI USE STATEMENT

AI tools were employed exclusively to improve the quality of written English and for stylistic refinement, with attentive supervision by the authors. No scientific content, experimental design, data analysis, or any intellectual contribution were generated through LLMs, nor did such assistance influence any reported results.

REPRODUCIBILITY STATEMENT

Section 3 specifies SSG's target construction, gate prediction, and scoring procedure. The appendix documents datasets and encoder checkpoints (Appendix A.1), architecture and training settings (Appendix A.3), and baseline implementations (Appendix A.2). Controlled ablations and training variability are reported in Appendix A.6. Model and hyperparameter selection use validation data, with test results reported for the selected configurations.

REFERENCES

Dhananjay Ashok, Suraj Nair, Mutasem Al-Darabsah, Choon Hui Teo, Tarun Agarwal, and Jonathan May. A representation sharpening framework for zero shot dense retrieval. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2026.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Tim Baumgärtner, Leonardo F. R. Ribeiro, Nils Reimers, and Iryna Gurevych. Incorporating relevance feedback for information-seeking retrieval using few-shot document re-ranking. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2022.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

Cesare Campagnano, Antonio Mallia, and Fabrizio Silvestri. Unveiling DIME: Reproducibility, generalizability, and formal analysis of dimension importance estimation for dense retrieval. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 2025.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, 2007.

Giulio D'Erasmo, Giovanni Trappolini, Fabrizio Silvestri, and Nicola Tonellotto. ECLIPSE: Contrastive dimension importance estimation with pseudo-irrelevance feedback for dense retrieval. In *Proceedings of the 2025 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR)*. Association for Computing Machinery, 2025.

Guglielmo Faggioli, Nicola Ferro, Raffaele Perego, and Nicola Tonellotto. Dimension importance estimation for dense information retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 2024.

Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*, 2022.

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

Omar Khattab and Matei Zaharia. 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*. Association for Computing Machinery, 2020.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham M. Kakade, Prateek Jain, and Ali Farhadi. Matryoshka representation learning. In *Advances in Neural Information Processing Systems*, 2022.

Canjia Li, Yingfei Sun, Ben He, Le Wang, Kai Hui, Andrew Yates, Le Sun, and Jungang Xu. NPRF: A neural pseudo relevance feedback framework for ad-hoc information retrieval. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*, 2023.

Zihan Liao, Hang Yu, Jianguo Li, Jun Wang, and Wei Zhang. D2LLM: Decomposed and distilled large language models for semantic search. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Luke Merrick, Danmei Xu, Gaurav Nuti, and Daniel Campos. Arctic-Embed: Scalable, efficient, and accurate text embedding models. *arXiv preprint arXiv:2405.05374*, 2024.

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

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2021.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021.

Nicholas Rossi, Juexin Lin, Feng Liu, Zhen Yang, Tony Lee, Alessandro Magnani, and Ciya Liao. Relevance filtering for embedding-based retrieval. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. Association for Computing Machinery, 2024.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2022.

Yash Saxena, Ankur Padia, Kalpa Gunaratna, and Manas Gaur. IMRNNs: An efficient method for interpretable dense retrieval via embedding modulation. In *Findings of the Association for Computational Linguistics: EACL 2026*. Association for Computational Linguistics, 2026.

Jonghyun Song, Cheyon Jin, Wenlong Zhao, Andrew McCallum, and Jay-Yoon Lee. Comparing neighbors together makes it easy: Jointly comparing multiple candidates for efficient and effective retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2024.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 2014.

Mujeen Sung, Jungsoo Park, Jaewoo Kang, Danqi Chen, and Jinhyuk Lee. Optimizing test-time query representations for dense retrieval. In *Findings of the Association for Computational Linguistics: ACL 2023*. Association for Computational Linguistics, 2023.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, 2020.

Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. ColBERT-PRF: Semantic pseudo-relevance feedback for dense passage and document retrieval. *ACM Transactions on the Web*, 2023.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-Pack: Packed resources for general Chinese embeddings. *arXiv preprint arXiv:2309.07597*, 2023.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*, 2021.

Jinsung Yoon, Yanfei Chen, Sercan Arik, and Tomas Pfister. Search-Adaptor: Embedding customization for information retrieval. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024.

HongChien Yu, Chenyan Xiong, and Jamie Callan. Improving query representations for dense retrieval with pseudo relevance feedback. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*. Association for Computing Machinery, 2021.

Puxuan Yu, Luke Merrick, Gaurav Nuti, and Daniel Campos. Arctic-Embed 2.0: Multilingual retrieval without compromise. *arXiv preprint arXiv:2412.04506*, 2024.

George Zerveas, Navid Rekabsaz, and Carsten Eickhoff. Enhancing the ranking context of dense retrieval through reciprocal nearest neighbors. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2023.

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. mGTE: Generalized long-context text representation and reranking models for multilingual text retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Association for Computational Linguistics, 2024.

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

A Appendix

A.1 Datasets and Encoder Specifications

Table 4 summarizes the seven BEIR datasets (Thakur et al., 2021). Total queries denotes the combined number of queries in our training, validation, and test partitions, before applying SSG's training-eligibility requirements.

Table 4: Corpus sizes and total queries used in our experiments.

Dataset	Documents	Total Queries
NFCorpus	3,633	3,237
SciFact	5,183	1,109
ArguAna	8,674	1,406
FiQA-2018	57,638	6,648
TREC-COVID	171,332	50
Touché 2020	382,545	49
Natural Questions	2,681,468	3,452

Table 5 lists the encoder checkpoints and embedding dimensions, with links to their model cards. All nine encoders are included in the generality evaluation. Detailed baseline comparisons use BGE-S, Arctic-M, and Qwen3-0.6B, while controlled ablations and efficiency measurements use BGE-S. Additional efficiency profiling uses Qwen3-0.6B and Qwen3-8B."

Table 5: Frozen encoder checkpoints and embedding dimensions.

Model	Checkpoint	Dim.
Arctic-S (Merrick et al., 2024)	`Snowflake/snowflake-arctic-embed-s`	384
BGE-S (Xiao et al., 2023)	`BAAI/bge-small-en`	384
Arctic-M (Yu et al., 2024)	`Snowflake/snowflake-arctic-embed-m-v2.0`	768
E5-L (Wang et al., 2022)	`intfloat/e5-large-v2`	1024
GTE-L (Li et al., 2023; Zhang et al., 2024)	`Alibaba-NLP/gte-large-en-v1.5`	1024
BGE-L (Xiao et al., 2023)	`BAAI/bge-large-en-v1.5`	1024
Qwen3-0.6B (Zhang et al., 2025)	`Qwen/Qwen3-Embedding-0.6B`	1024
Qwen3-4B (Zhang et al., 2025)	`Qwen/Qwen3-Embedding-4B`	2560
Qwen3-8B (Zhang et al., 2025)	`Qwen/Qwen3-Embedding-8B`	4096

A.2 Baseline Implementation Details

All baselines use the same query splits and evaluation metric; embedding-based methods operate on the corresponding pretrained representations, while reranking methods use the same first-stage candidate sets where applicable. Implementation-specific settings are detailed below; where hyperparameters are searched, selection uses validation nDCG@10 rather than test performance.

Global Gate: To assess the benefit of conditional prediction, we replace the SSG predictor with a single trainable vector $\mathbf{g} = \tanh(\mathbf{u})$, shared across all queries within a dataset-encoder pair. The parameter vector $\mathbf{u} \in \mathbb{R}^D$ is initialized to zero, giving D trainable parameters. It receives supervision through SSG's cached dimensional targets, but takes no query or candidate features as input. Training follows the corresponding SSG optimization and early-stopping settings, including the temperature and residual-strength grids. The selected vector reranks the original top-100 candidates using SSG's residual scoring formulation.

Vector PRF: Our simple Vector PRF baseline, following the general dense pseudo-relevance-feedback formulation (Yu et al., 2021), forms a centroid $\mathbf{c}_k$ from the top-k initially retrieved document embeddings and constructs $\mathbf{q}_{\text{PRF}} = \text{norm}((1-\beta)\mathbf{q} + \beta\mathbf{c}_k)$. The resulting query reranks the same top-100 candidates by dot product and therefore performs no second full-corpus search. We select $k \in \{1, 2, 5, 10\}$ and $\beta \in \{0.1, 0.25, 0.5, 0.75, 1.0\}$ using validation nDCG@10. The method requires no training or corpus transformation.

PRF-DIME: Our PRF-DIME implementation (Faggioli et al., 2024; Campagnano et al., 2025) uses full-corpus second-search retrieval. Dimension importance is the element-wise product of the query embedding and the centroid of its top-k initially retrieved documents. The highest-scoring coordinates define a binary query mask; the masked query retrieves 100 documents from the existing FAISS index without renormalization. The search covers 12 combinations of feedback depth $k \in \{1, 2, 5\}$ and retained fraction $r \in \{0.2, 0.4, 0.6, 0.8\}$, with validation ties favoring smaller k, then smaller r. This baseline requires no training or corpus transformation. Only full-corpus effectiveness results are reported in Table 1 and Table 2; the fixed-candidate variant is included only in the latency analysis in Table 3.

ECLIPSE: ECLIPSE's contrastive dimension-selection procedure (D'Erasmo et al., 2025) assigns importance according to $\mathbf{q} \odot (\alpha \boldsymbol{\mu}_+ - \beta \boldsymbol{\mu}_-)$, using centroids of the highest- and lowest-ranked feedback documents. A binary mask retains the highest-importance coordinates, and the masked query searches the unchanged full-corpus index without renormalization, returning up to 1000 documents. No model training or corpus transformation is required. Our compact grid contains 80 configurations: $k_+, k_- \in \{2, 5\}$, $\alpha = 1$, $\beta \in \{0.25, 0.5, 1.0, 2.0\}$, and $r \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$. Validation ties favor more retained dimensions, followed by configuration order.

CMC: We implement a matched frozen-embedding variant of CMC (Song et al., 2024) over the same bi-encoder representations and candidate sets used by SSG. The query and candidates are jointly processed by two position-free Transformer encoder layers with 8 attention heads, feed-forward width 2048, dropout 0.1, and an outer residual connection. Training uses one judged positive and 63 retrieved hard negatives per query; for queries with multiple positives, one is sampled per epoch and the remaining positives are excluded from the negative pool. The objective combines cross-entropy with KL regularization toward the first-stage score distribution. We train for three epochs with batch size 8, AdamW, 10% linear warm-up, and learning rates in $\{10^{-5}, 5\times10^{-6}, 2\times10^{-6}\}$. Because the original CMC paper does not report the objective weights, we select $(\lambda_{\mathrm{CE}}, \lambda_{\mathrm{KL}})$ from $\{(0.9, 0.1), (0.75, 0.25), (0.5, 0.5)\}$ using validation nDCG@10. Only the contextualizer is trained; the underlying bi-encoder representations remain frozen. At inference, CMC jointly contextualizes the original top-100 candidates and scores them by the dot product between contextualized query and candidate vectors.

IMRNNs: The authors' implementation (Saxena et al., 2026) is integrated through a data loader for our cached embeddings. We retain the fixed configuration from the authors' implementation and use a 256-dimensional projection, hidden width 128, and dropout 0.1. The provided training routine runs for 20 epochs with batch size 32, learning rate 10^{-4}, weight decay 10^{-5}, and seed 42. Each example pairs the first available judged positive with 20 BM25-mined negatives from a pool of 200; shorter negative lists repeat the last available negative. At inference, normalized projected embeddings support top-100 retrieval through a separate 256-dimensional FAISS inner-product index. The retrieved documents' original embeddings then enter the model's candidate-scoring routine, which determines the final order.

Search-Adaptor: Our reimplementation of Search-Adaptor (Yoon et al., 2024) retains its shared residual adaptor and auxiliary document-to-query predictor. Both are instantiated as $D \to D \to D$ ReLU MLPs, an explicit architecture choice. The objective combines a mean relevance-weighted logistic loss over sampled positive-negative comparisons with L_1 recovery and query-prediction losses. Batches contain 128 positive relationships, with negative pairs randomly subsampled at a 10:1 ratio following the original Search-Adaptor training protocol. Unlike the general ranking objective, this sampling does not compare positives with different relevance grades.

Training uses Adam with learning rate 10^{-3} for at most 2000 iterations. Nine configurations combine recovery weights $\alpha \in \{0, 0.1, 1.0\}$ and prediction weights $\beta \in \{0, 0.01, 0.1\}$. Checkpoint selection uses fixed-top-100 validation as a computational proxy, evaluated every 25 iterations with patience 125 iterations. Full-corpus validation then selects among the retained checkpoints. Final retrieval searches a separately constructed FAISS index of adapted, L_2-normalized corpus embeddings using similarly adapted queries, returning 100 documents.

MiniLM Cross-Encoder: We evaluate the `cross-encoder/ms-marco-MiniLM-L6-v2` checkpoint (Wang et al., 2020) without target-domain fine-tuning. For each query, the cross-encoder

scores the original top-100 documents returned by the corresponding first-stage retriever and reranks this fixed candidate set. Query-document pairs use the document title and text, with a maximum sequence length of 512 tokens. Evaluation uses batches of 128 pairs with FP16 inference. Thus, this baseline is evaluated zero-shot on the target BEIR collections and cannot introduce documents outside the initial top-100.

A.3 SSG Implementation and Model Selection

Table 6 summarizes the SSG configuration. Training inputs use the original retrieved candidates, while targets use the separate labeled training-candidate table. Checkpoints, target temperature, and residual strength are selected by validation nDCG@10; the reported test result uses the selected configuration. For TREC-COVID and Webis-Touché 2020, model selection is performed using five-fold cross-validation over the development queries. Hyperparameters are selected from the fold results, after which the selected configuration is refit on the development data and evaluated once on the fixed test set.

Table 6: SSG configuration and hyperparameter settings.

Parameter	**Setting**
Inputs and targets	
Candidate depth	$K = 100$
Hard negatives	$H \in \{20, 50, 100\}$; selected $H = 20$
Input dimension	$4D$
Target standardization	Population z-score, $\epsilon = 10^{-6}$
Target temperature	$\tau \in \{1.0, 0.95, 0.90, 0.85, 0.80\}$
Architecture	
Input normalization	LayerNorm (Ba et al., 2016)
Layer dimensions	$4D \to 512 \to 512 \to D$
Hidden activation	GELU (Hendrycks & Gimpel, 2016)
Hidden dropout	$p = 0.1$ (Srivastava et al., 2014)
Output activation	$\tanh$
Optimization	
Loss	Smooth L1 (mean) (Girshick, 2015)
Optimizer	AdamW (Loshchilov & Hutter, 2019)
Initial learning rate	10^{-4}
Weight decay	10^{-3}
Batch size	32
Maximum epochs	160
Training seed	42
Model selection	
Residual strength	$\alpha \in \{0.5, 0.75, 1.0\}$
Scoring	Unnormalized residual dot product
Selection metric	Validation nDCG@10
Evaluation interval	Every epoch
Early-stopping patience	30 evaluations
Minimum training epochs	50

A.4 Detailed Results across Embedding Models

Table 7 reveals how improvements vary across individual retrieval tasks. At three-decimal precision, SSG improves 62 of the 63 encoder-dataset comparisons, with one decrease. SciFact gains range from 0.055 to 0.113, whereas Natural Questions gains range from 0.002 to 0.095. ArguAna illustrates stronger encoder-dependent variation: E5-L improves from 0.488 to 0.634, while GTE-L increases from 0.723 to 0.728.

Table 7: Generality of SSG across nine frozen dense encoders spanning 384-4096 dimensions. Test nDCG@10 is reported for the original base retriever and the corresponding SSG reranker. The better result within each encoder-dataset pair is shown in **bold**. Avg. denotes the mean across all seven datasets.

Model	Method	Dim.	nDCG@10							
			NFCorpus	SciFact	ArguAna	FiQA	TREC-COVID	Touché-2020	NQ	Avg.
Arctic-S	Base Retrieval	384	0.326	0.698	0.576	0.396	0.829	0.367	0.566	0.537
	SSG	384	**0.349**	**0.798**	**0.592**	**0.408**	**0.843**	**0.396**	**0.575**	**0.566**
BGE-S	Base Retrieval	384	0.335	0.699	0.592	0.399	0.775	0.343	0.492	0.519
	SSG	384	**0.343**	**0.754**	**0.658**	**0.402**	**0.821**	**0.350**	**0.586**	**0.559**
Arctic-M	Base Retrieval	768	0.359	0.718	0.597	0.440	0.819	0.339	0.653	0.561
	SSG	768	**0.372**	**0.815**	**0.641**	**0.446**	**0.840**	**0.345**	**0.655**	**0.588**
E5-L	Base Retrieval	1024	0.365	0.715	0.488	0.387	0.687	0.257	0.630	0.504
	SSG	1024	**0.378**	**0.809**	**0.634**	**0.423**	**0.735**	**0.298**	**0.642**	**0.560**
GTE-L	Base Retrieval	1024	0.370	0.827	0.723	**0.629**	0.817	0.258	0.557	0.597
	SSG	1024	**0.391**	**0.902**	**0.728**	0.626	**0.833**	**0.275**	**0.562**	**0.617**
BGE-L	Base Retrieval	1024	0.367	0.731	0.654	0.443	0.810	0.282	0.554	0.549
	SSG	1024	**0.401**	**0.825**	**0.676**	**0.455**	**0.871**	**0.300**	**0.566**	**0.585**
Qwen3-0.6B	Base Retrieval	1024	0.355	0.700	0.690	0.466	0.897	0.278	0.558	0.564
	SSG	1024	**0.368**	**0.812**	**0.732**	**0.474**	**0.920**	**0.322**	**0.591**	**0.603**
Qwen3-4B	Base Retrieval	2560	0.406	0.768	0.726	0.584	0.877	0.356	0.645	0.623
	SSG	2560	**0.419**	**0.852**	**0.733**	**0.589**	**0.904**	**0.367**	**0.648**	**0.645**
Qwen3-8B	Base Retrieval	4096	0.399	0.767	0.744	0.611	0.915	0.420	0.671	0.647
	SSG	4096	**0.409**	**0.845**	**0.750**	**0.612**	**0.916**	**0.423**	**0.673**	**0.661**

The only displayed decrease occurs for GTE-L on FiQA (0.629 to 0.626). The exception shows that the aggregate improvements do not imply gains in every setting; small score differences also do not by themselves establish statistical significance.

A.5 Higher-Dimensional Efficiency Profiling

Table 8: Per-dataset efficiency profiling of SSG with higher-dimensional encoders. Component timings are measured after first-stage retrieval. Vector access denotes loading the top-100 retrieved document embeddings. End-to-end (E2E) latency includes query encoding, retrieval, and reranking.

Model	**Dataset**	**Dim.**	**SSG params.**	**SSG component latency (ms/query)**				**E2E latency (ms/query)**	
				Vector access	**Statistics**	**Predictor**	**Rescore & sort**	**Base**	**+SSG**
Qwen3-0.6B	NFCorpus	1024	2,893,824	0.16	0.32	0.63	0.15	45.58	46.83
	SciFact			0.16	0.32	0.62	0.15	46.30	47.54
	ArguAna			0.20	0.34	0.63	0.15	49.38	50.56
	FiQA			0.54	0.37	0.85	0.16	74.94	77.01
	TREC-COVID			16.24	0.37	0.86	0.16	131.47	149.04
	Touché-2020			30.32	0.37	0.88	0.16	232.74	264.54
	NQ			113.15	0.38	0.92	0.16	1349.62	1463.99
Qwen3-8B	NFCorpus	4096	10,785,792	3.87	0.97	0.76	0.29	83.51	89.51
	SciFact			0.55	0.98	0.80	0.29	87.26	90.01
	ArguAna			0.69	1.02	0.86	0.29	139.78	143.38
	FiQA			7.20	1.01	0.90	0.29	202.53	212.02
	TREC-COVID			71.95	1.05	0.94	0.29	436.33	511.40
	Touché-2020			175.76	1.05	0.94	0.29	868.74	1045.98
	NQ			1431.26	1.24	1.37	0.32	5489.24	6917.68

To characterize SSG beyond the 384-dimensional BGE-S setting, we additionally profile Qwen3-0.6B ($D = 1024$) and Qwen3-8B ($D = 4096$). Measurements use an NVIDIA L4 GPU with CPU

FAISS FlatIP, up to 200 test queries per dataset, three repeats, and 10 warm-up queries. Because this profiling uses different hardware from the BGE-S benchmark in Table 3, absolute latencies should not be compared across the two experiments; the analysis here focuses on scaling within the higher-dimensional setting.

Across datasets, candidate-statistics computation, gate prediction, and rescoring remain small even at $D = 4096$, generally requiring only around 2-3 ms/query in total. Because SSG always operates on a fixed $K = 100$ candidate set, these costs remain nearly constant within each encoder. The larger overhead on TREC-COVID, Touché-2020, and especially NQ is instead dominated by accessing the retrieved document vectors.

This vector-access cost becomes more pronounced at higher dimensionality, increasing on average from 22.97 ms/query at $D = 1024$ to 241.61 ms/query at $D = 4096$. Consequently, end-to-end latency increases from 275.72 to 299.93 ms/query for Qwen3-0.6B (8.8%) and from 1043.91 to 1287.14 ms/query for Qwen3-8B (23.3%). Thus, the higher-dimensional overhead is driven primarily by candidate-vector access rather than gate prediction itself; parameter storage also grows with D.

A.6 Ablation Study

Table 9 evaluates SSG's target construction, training objective, input features, and scoring function using BGE-S across seven datasets. Ablations use seed 42 with matched training data and network capacity where applicable. Hyperparameters and checkpoints for trained variants are selected using validation nDCG@10. Full SSG achieves the highest average score, improving Base Retrieval from 0.519 to 0.559.

Table 9: Ablation study with BGE-S across seven BEIR datasets. Results are test nDCG@10; Avg. is the macro average. Best and second-best results in each column are shown in **bold** and underlined, respectively.

Variant	NFCorpus	SciFact	ArguAna	FiQA	TREC-COVID	Touché-2020	NQ	Avg.
Base Retrieval	0.335	0.699	0.592	0.399	0.775	0.343	0.492	0.519
Full SSG	**0.343**	**0.754**	**0.658**	0.402	**0.821**	**0.350**	**0.586**	**0.559**
Target construction								
Target w/o mean subtraction	0.332	0.748	0.575	0.400	0.777	0.349	0.506	0.527
Training objective								
Pairwise ranking loss	0.272	0.730	0.610	0.245	0.722	0.323	0.484	0.484
Listwise ranking loss	0.323	0.726	0.624	0.261	0.689	0.344	0.494	0.494
Gate input								
Without candidate statistics	0.337	0.748	0.601	**0.403**	0.793	0.347	0.498	0.532
Without absolute query embedding	0.335	0.739	0.609	0.398	0.785	0.345	0.512	0.532
Scoring function								
Linear gated dot product	0.341	0.749	0.618	0.400	0.803	0.349	0.511	0.539
Symmetric gated cosine	0.338	0.728	0.605	0.398	0.791	0.347	0.500	0.530

Removing mean subtraction from the dimensional target reduces the average from 0.559 to 0.527. This supports using the centered relevance contrast, which expresses each dimension relative to the query-specific distribution of positive-negative differences.

We compare target regression with pairwise logistic loss (Burges et al., 2005) and a listwise cross-entropy variant (Cao et al., 2007) using uniform probability over positives. For each ranking-loss variant, score temperature, residual strength, and checkpoints are independently selected by validation nDCG@10. Both use SSG's supervision documents and a score-temperature grid of $\{0.02, 0.05, 0.1, 0.2, 1.0\}$. Their averages of 0.484 and 0.494 fall below SSG's 0.559, supporting dimensional supervision under the evaluated settings.

Zeroing the candidate-statistics or absolute-query block before normalization reduces the average to 0.532 in both cases. Both inputs therefore improve aggregate performance, despite exceptions on individual datasets. In particular, removing candidate statistics slightly improves FiQA (0.403 vs. 0.402), indicating that their contribution can vary across retrieval tasks.

Replacing $m_{q,j}^2$ with $m_{q,j}$ or renormalizing the modulated vectors reduces the average to 0.539 and 0.530, respectively. These results favor unnormalized squared weighting under the evaluated settings. Residual strength is selected independently for each scoring variant using validation nDCG@10.

Label-Informed Diagnostic: We examine the ranking information in SSG's dimensional targets by constructing gates directly from test relevance labels using BGE-S. Targets use only the original top-100 candidates, requiring at least one judged positive and four negatives, with unjudged documents treated as nonrelevant. Temperature and residual strength remain validation-selected. All test queries are evaluated; queries without a valid target retain their base scores.

Table 10: Label-informed diagnostic with BGE-S. Scores are test nDCG@10 over all test queries. Coverage denotes the percentage of queries eligible for target construction. SSG scores match Table 1.

Dataset	Base	SSG	Label-informed	Coverage
NFCorpus	0.335	0.343	0.562	83.0%
SciFact	0.699	0.754	0.952	95.3%
ArguAna	0.592	0.658	0.962	98.6%
FiQA-2018	0.399	0.402	0.721	84.9%
TREC-COVID	0.775	0.821	0.909	100.0%
Touché 2020	0.343	0.350	0.576	100.0%
Natural Questions	0.492	0.586	0.897	91.3%

Table 10 shows substantially higher scores when the gate is constructed directly from test relevance labels. This diagnostic has privileged access to explicit positive and negative document identities, whereas the inference-time predictor observes only the query and permutation-invariant statistics of the unlabeled candidate set. The gap therefore reflects both this structural information asymmetry and the difficulty of predicting the corresponding dimensional weighting from candidate summaries alone.

Because it uses test relevance labels, the diagnostic is neither a deployable method nor a guaranteed upper bound on achievable ranking performance; rather, it measures the ranking potential of label-informed dimensional reweighting within the fixed candidate set.

Training Variability and Confidence Intervals: Table 11 reports BGE-S results across five seeds (41-45), with fixed splits, target temperatures, and optimization settings; checkpoints and residual strengths are selected on validation data for each seed. SSG improves Base Retrieval in all 35 seed-dataset settings, achieving a seven-dataset average of 0.5569 ± 0.0029 across seeds, versus 0.5190 for Base Retrieval. The 95% paired bootstrap interval excludes zero in 34 of the 35 comparisons. Per-checkpoint confidence intervals use 10,000 paired test-query bootstrap resamples without multiple-comparison correction.

Table 11: BGE-S test nDCG@10 across five training seeds. Δ denotes improvement over Base Retrieval, with 95% paired bootstrap confidence intervals. Summary rows report mean $\pm$ sample standard deviation across seeds.

Dataset	**Seed**	**SSG**	**Δ**	**95% CI for Δ**
NFCorpus	41	0.3444	+0.0099	[+0.0045, +0.0154]
	42	0.3432	+0.0087	[+0.0023, +0.0146]
	43	0.3459	+0.0114	[+0.0035, +0.0192]
	44	0.3481	+0.0135	[+0.0052, +0.0219]
	45	0.3422	+0.0077	[+0.0028, +0.0128]
	Mean $\pm$ SD	0.3448 ± 0.0023		
SciFact	41	0.7451	+0.0462	[+0.0308, +0.0628]
	42	0.7542	+0.0554	[+0.0272, +0.0587]
	43	0.7578	+0.0589	[+0.0227, +0.0659]
	44	0.7463	+0.0475	[+0.0293, +0.0667]
	45	0.7435	+0.0446	[+0.0047, +0.0552]
	Mean $\pm$ SD	0.7494 ± 0.0063		
ArguAna	41	0.6559	+0.0639	[+0.0613, +0.0656]
	42	0.6580	+0.0660	[+0.0630, +0.0689]
	43	0.6562	+0.0642	[+0.0630, +0.0686]
	44	0.6502	+0.0582	[+0.0568, +0.0639]
	45	0.6446	+0.0526	[+0.0492, +0.0543]
	Mean $\pm$ SD	0.6530 ± 0.0055		
FiQA-2018	41	0.4008	+0.0023	[+0.0012, +0.0046]
	42	0.4018	+0.0033	[+0.0023, +0.0059]
	43	0.4029	+0.0044	[+0.0032, +0.0068]
	44	0.4031	+0.0045	[+0.0019, +0.0077]
	45	0.4033	+0.0048	[−0.0001, +0.0059]
	Mean $\pm$ SD	0.4024 ± 0.0011		
TREC-COVID	41	0.8053	+0.0307	[+0.0101, +0.0794]
	42	0.8210	+0.0464	[+0.0104, +0.0631]
	43	0.8210	+0.0464	[+0.0101, +0.0868]
	44	0.7934	+0.0188	[+0.0089, +0.0761]
	45	0.8286	+0.0540	[+0.0182, +0.0667]
	Mean $\pm$ SD	0.8139 ± 0.0142		
Touché 2020	41	0.3470	+0.0040	[+0.0016, +0.0108]
	42	0.3495	+0.0065	[+0.0049, +0.0185]
	43	0.3553	+0.0123	[+0.0101, +0.0146]
	44	0.3526	+0.0096	[+0.0078, +0.0099]
	45	0.3487	+0.0058	[+0.0053, +0.0164]
	Mean $\pm$ SD	0.3506 ± 0.0033		
Natural Questions	41	0.5821	+0.0905	[+0.0823, +0.0993]
	42	0.5864	+0.0948	[+0.0893, +0.0956]
	43	0.5835	+0.0919	[+0.0854, +0.0985]
	44	0.5817	+0.0901	[+0.0825, +0.0977]
	45	0.5888	+0.0972	[+0.0886, +0.0997]
	Mean $\pm$ SD	0.5845 ± 0.0030		