Decoding Dense Embeddings: Sparse Autoencoders for Interpreting and Discretizing Dense Retrieval

Seongwan Park
SungKyunKwan University
Republic of Korea
waniboyy@gmail.com

Taeklim Kim
SungKyunKwan University
Republic of Korea
kimtaeklim2002@gmail.com

Youngjoong Ko*
SungKyunKwan University
Republic of Korea
yjko@skku.edu

Abstract

Despite their strong performance, Dense Passage Retrieval (DPR) models suffer from a lack of interpretability. In this work, we propose a novel interpretability framework that leverages Sparse Autoencoders (SAEs) to decompose previously uninterpretable dense embeddings from DPR models into distinct, interpretable latent concepts. We generate natural language descriptions for each latent concept, enabling human interpretations of both the dense embeddings and the query-document similarity scores of DPR models. We further introduce Concept-Level Sparse Retrieval (CL-SR), a retrieval framework that directly utilizes the extracted latent concepts as indexing units. CL-SR effectively combines the semantic expressiveness of dense embeddings with the transparency and efficiency of sparse representations. We show that CL-SR achieves high index-space and computational efficiency while maintaining robust performance across vocabulary and semantic mismatches[1].

1 Introduction

Traditional information retrieval methods have relied on exact lexical matching between query and document terms to determine document relevance (Craswell et al., 2018). Despite their efficiency and transparency, these sparse retrieval techniques suffer from *vocabulary mismatch*, where a query and the relevant documents use different terms (e.g., *cat* vs. *kitty*), and *semantic mismatch*, where the same term can refer to different concepts (e.g., *bank of river* vs. *bank in finance*) (Gao et al., 2021).

The advent of Pre-trained Language Models (PLMs) have led to the emergence of dense retrieval approaches as promising alternatives for overcoming the limitations of sparse methods (Zhao et al., 2024). Dense retrieval methods embed queries and documents onto a continuous vector space by utilizing dense embeddings to represent contextualized semantics and enabling similarity computations beyond simple keyword matches. Consequently, dense retrieval effectively addresses the vocabulary and semantic mismatch issues inherent in sparse retrieval, achieving state-of-the-art (SOTA) performance across various information retrieval (IR) benchmarks (Huang and Chen, 2024; Xu et al., 2024). However, dense retrieval suffers from a fundamental limitation: the difficulty of interpreting the dense embeddings and the ranking results. This lack of interpretability poses a significant challenge in applications where transparency and user trust in search results are critical, leading to various attempts to interpret dense retrieval models (Anand et al., 2022). Recently, sparse autoencoders (SAEs) have garnered significant attention as a method to disentangle the complex semantic structures inherent in the dense embeddings of decoder-only transformer models, into distinct and interpretable conceptual units (i.e., latent concepts) (Bricken et al., 2023; Templeton et al., 2024; Huben et al., 2024).

In this work, we propose a novel explainable AI (XAI) framework that extends SAEs to the field of information retrieval by applying them to the embeddings of Dense Passage Retrieval (DPR) models. Our method is summarised in Figure 1.

First, we train SAEs and present qualitative measures to evaluate SAEs' ability of decomposing the semantic information embedded in the original dense embeddings into linear combinations of latent concepts. Furthermore, we qualitatively analyze if each extracted individual latent concepts hold distinct semantic meanings (Section 3).

We then generate natural language descriptions for each latent concept, enabling interpretation of both the semantic content of dense embeddings and the similarity computation between queries and documents. We qualitatively evaluate our frame-

*Corresponding author

[1]The code is available at https://github.com/Tro-fish/Decoding-Dense-Embeddings

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 26468–26485
November 4-9, 2025

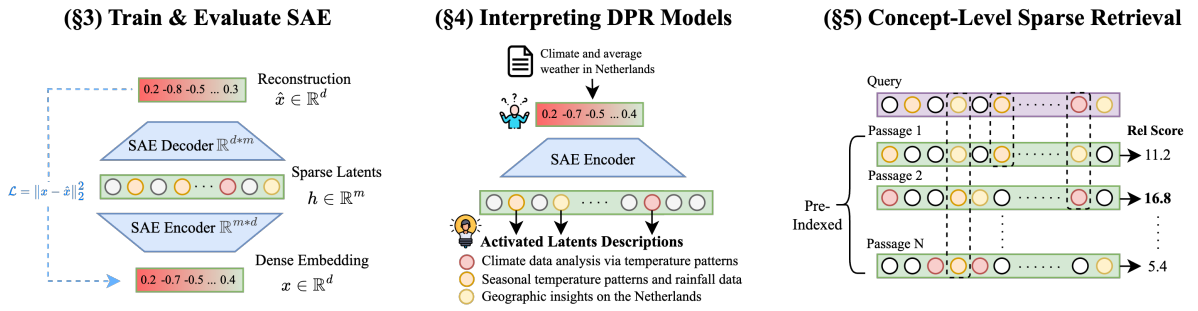

Figure 1: Overview of our method. We first train a SAE to decompose dense embeddings into latent concepts (§3). Given a query or a passage, the SAE encoder sparsely activates latent concepts, which are mapped to natural language descriptions allowing human interpretability tasks (§4). In CL-SR, queries and passages are represented as sets of activated latent concepts (§5). We also demonstrate its effectiveness on subsets where traditional sparse retrieval methods struggle.

work by performing multiple human interpretability tasks (Section 4).

Building on this, we futher introduce Concept-Level Sparse Retrieval (CL-SR), a retrieval framework that treats each latent concept as a fundamental unit of retrieval. Unlike traditional term-based sparse methods, which are prone to vocabulary and semantic mismatches, CL-SR mitigate these limitations by leveraging semantically generalized latent concepts. CL-SR also achieves higher retrieval efficiency with fewer matching units compared to term-based retrieval (Section 5).

2 Related Work

2.1 Interpretability in Dense Passage Retrieval

Dense passage retrieval employs PLM-based encoders to embed queries and passages[2] into dense vectors—typically obtained from the [CLS] token representation or mean pooling over token embeddings—and measures relevance via dot product or cosine similarity between these embeddings, exhibiting superior performance across a variety of IR tasks. However, DPR inherently suffers from a lack of interpretability due to the implicit semantics encoded within the uninterpretable dense embedding space. To address this shortcoming, various attempts have been made to interpret the inner workings of DPR. Völske et al. (2021), analyzes the ranking determinants between document pairs based on axioms defined in traditional IR theory—e.g., term frequency, document length, semantic similarity, and term proximity—by formalizing each axiom for rank comparison and training a simple explanatory model that approximates DPR's ranking. A different approach, Llordes et al. (2023) generate an "equivalent query" through discrete state-space search that reproduces DPR's ranking results under a sparse retrieval model, thereby explaining the semantic information used by DPR at the term level. However, while these approaches provide a proxy-based interpretation that approximates neural behavior through input manipulation and external alignment, the internal representational structure of the DPR model remains unexplored.

2.2 Sparse Autoencoders

The "superposition hypothesis" suggests that neural networks are able to encode more features than their available dimensions, exploiting sparsity of feature activation (Elhage et al., 2022). This results in superposed representations that are difficult to interpret directly due to polysemanticity. Recent work applies a set of methods called sparse coding or sparse dictionary learning to identify underlying true features actually used by neural networks (Sharkey et al., 2025). One of the simplest and widely explored methods is SAE. SAE is a single-layer feedforward autoencoder with a hidden layer larger than the input dimension, incorporating a sparsity constraint to ensure that only a small subset of hidden neurons (i.e., latent concepts) activate for any given input. Recent studies have empirically demonstrated that SAEs can effectively extract interpretable features from the activations of decoder-only large language models(LLMs) (Huben et al., 2024; Marks et al., 2025). While there have been attempts to apply SAEs to dense embeddings gen-

[2]In this paper, the terms "passage" and "document" are used interchangeably.

Model	NMSE	MSMARCO Dev		TREC DL 2019			TREC DL 2020		
		MRR@10	Recall@1k	NDCG@10	Recall@1k	Spearman	NDCG@10	Recall@1k	Spearman
Baseline (SimLM)	–	0.411	0.986	0.714	0.767	–	0.697	0.772	–
Reconstructed (k=32)	0.1903	0.328 (×0.80)	0.964 (×0.98)	0.589 (×0.83)	0.666 (×0.87)	0.928 ±7.7E-3	0.582 (×0.84)	0.681 (×0.88)	0.927 ±8.1E-3
Reconstructed (k=48)	0.1643	0.338 (×0.83)	0.968 (×0.98)	0.624 (×0.87)	0.686 (×0.89)	0.936 ±6.7E-3	0.613 (×0.88)	0.704 (×0.92)	0.933 ±7.2E-3
Reconstructed (k=64)	0.1458	0.347 (×0.84)	0.973 (×0.98)	0.640 (×0.90)	0.692 (×0.90)	0.949 ±6.2E-3	0.608 (×0.87)	0.718 (×0.93)	0.948 ±6.8E-3
Reconstructed (k=128)	0.1069	**0.371** (×0.90)	**0.980** (×0.99)	**0.664** (×0.93)	**0.726** (×0.95)	**0.959** ±5.2E-3	**0.629** (×0.90)	**0.734** (×0.95)	**0.956** ±5.4E-3

Table 1: Quantitative evaluation of the reconstruction capability of SAEs trained on SimLM embeddings. Results include normalized mean squared error (NMSE) and preservation of retrieval performance on MSMARCO Dev and TREC DL 2019/2020. Spearman's correlation is averaged across TREC DL 2019 and 2020, with variance also reported ($p < 0.05$).

	SAE k=32	SAE k=48	SAE k=64	SAE k=128
Accuracy	0.859	0.829	0.808	0.764

Table 2: Results of latent intrusion test using MSMARCO passages. We report accuracy over different numbers of activated latents k.

erated by encoder-based models (Ye et al., 2024; Kang et al., 2025), these efforts have largely focused on the interpretation of the dense embeddings themselves, leaving the interpretability of retrieval results and their implications in IR tasks underexplored.

3 SAE Training and Evaluation for DPR Model Interpretation

Interpreting a DPR model through SAE latent concepts requires both an effective training of the SAE and a verification that its reconstructed embeddings faithfully preserve the information from the original embeddings. Additionally, it is essential to evaluate whether each latent concept extracted by the SAE represents clear semantic concept. In this section, we outline the training and evaluation procedures for achieving these objectives.

3.1 Training Sparse Autoencoder

A SAE is optimized to reconstruct the input vector $h \in \mathbb{R}^d$ by learning a sparse latent representation $z \in \mathbb{R}^m (m >> d)$ with sparsity constraint $L_0(z) = k (k << d)$. Formally, a SAE consists of

$$\text{Encoder:} \quad z(h) = \sigma(W_{\text{enc}} h + b_{\text{enc}}) \tag{1}$$

$$\text{Decoder:} \quad \hat{h}(z) = z W_{\text{dec}} + b_{\text{dec}} \tag{2}$$

where $W_{enc} \in \mathbb{R}^{m*d}$, $b_{enc} \in \mathbb{R}^m$, $W_{dec} \in \mathbb{R}^{d*m}$, and $b_{dec} \in \mathbb{R}^d$ are the parameters of encoder and decoder respectively and $\sigma(\cdot)$ is the activation function. Although there are various SAE variants depending on the activation and sparsity mechanisms, we employ widely used BatchTopK SAE (Bussmann et al., 2024), where the model is trained to minimize the following loss:

$$L(h) = \left\| h - \hat{h}(z(h)) \right\|_2^2 + \lambda L_{\text{aux}}, \tag{3}$$

$$z(h) = \text{BatchTopK}(W_{\text{enc}} h + b_{\text{enc}}). \tag{4}$$

The BatchTopK activation function masks latents whose activation values are not in the top n * k to 0 across a batch of n samples, allowing flexible allocation of the number of latents for each sample in a single batch and L_{aux} is an auxiliary loss, used to prevent dead latents. At inference time, latents with activation larger than mean of top k-th activation over the whole datapoints are considered "activated" and all other activations are zeroed out resulting in a sparse m dimensional vector $z(h)$. Training the SAE requires dense embeddings generated by a pretrained DPR model. In this work, we adopt SimLM (Wang et al., 2023a) as our target model to interpret and use it to embed approximately 8.8 million passages and 0.5 million train queries from the MSMARCO passage retrieval dataset (Bajaj et al., 2018) into dense vectors. The SAE is then trained to reconstruct these dense embeddings[3]. We set the hyperparameter m = 32 * d and experiments with k of 32, 48, 64, 128. We detail the training setup and dataset in Appendix A.

3.2 Evaluating Sparse Autoencoder

Now we propose to quantitatively evaluate the SAE based on following criteria.

1. **Vector-level Reconstruction Fidelity**: We measure the normalized mean squared error (NMSE) between the original DPR embeddings and the reconstructed embeddings. Specifically, NMSE is calculated by dividing the raw MSE by the baseline reconstruction error of always predicting the mean activation.

[3]We show generalizability of our framework across target DPR models and to unseen datasets in Appendix C.

2. **Preservation of IR Performance**: Previous works on SAEs have measured language modeling performance change as a measure of SAE reconstruction fidelity (Rajamanoharan et al., 2024; Gao et al., 2025). We evaluate how much of the downstream dense retrieval performance is maintained by conducting dense retrieval with the reconstructed embeddings instead.

3. **Ranking Result Reconstruction Fidelity**: Beyond retrieval performance, we examine how faithfully the reconstructed embeddings retain the detailed ranking order of retrieved documents. We report Spearman's correlation between the two ranked lists obtained from the target model and reconstructed embeddings.

For evaluation, we used the MSMARCO Dev and TREC Deep Learning (DL) Track 2019/2020 datasets (Craswell et al., 2020, 2021). Table 1 shows a trade-off between the reconstruction quality and the degree of sparsity of SAEs. This trend can be observed with all three criteria we adapted to measure reconstruction fidelity of SAEs in IR settings and is consistent with prior studies on sparse autoencoders on decoder models confirming soundness of our measures.

Additionally, since our goal is to interpret the DPR model via individual latent concepts, we conduct experiments to verify whether each latent indeed represents an interpretable semantic concept. Inspired by the "word intrusion test" (Chang et al., 2009), we perform a latent intrusion test, where we collect 9 passages from MSMARCO corpus that most strongly activate a given latent plus 1 randomly chosen "intruder" passage that does not activate the latent. We then use a powerful LLM (GPT-4.1 mini) to identify the outlier. Table 2 shows that, though individual latent's quality degrades as sparsity decreases, SAEs are still effective at disentangling semantic structures within dense embeddings of DPR models into latent concepts that retain distinct, meaningful information.

4 Interpreting DPR Models Through Latent Descriptions

Building on the latent concepts from the trained SAE, in the following sections, we first generate natural language description for each latent concept (§4.1) and then we demonstrate the utility of these descriptions in two downstream human interpretability tasks[4]: understanding the semantic structure of DPR's dense embeddings (§4.2) and simulating the model's ranking process between query and documents (§4.3).

Title: 1960s Important News and Events, Key Technology Fashion and Popular Culture
Context: In 1960 the average income per year was $5,315.00 and by 1969 was $8,540.00. … A few more prices from the 60's and how much things cost. … If you have $100 converted from 1960 to 2005 it would be equivalent to $679.09 today. In 1960 a new house cost $12,700.00 and by 1969 was $15,500.00.

Latent index	Description	Activation value
8983	Historical financial and social benchmarks 1960	5.80
21047	Contexts of income-related financial and economic	1.62
20216	1960s-70s cultural and fashion evolution	1.58
1119	Economic inflation definition and causes	1.44
24152	Salary trend reports with comparative yearly analysis	1.35

Figure 2: Top-5 most activated latent concepts extracted from the dense embedding of a passage discussing economic benchmarks and cultural context in the 1960s. Reported activation value is weighted by IDF.

4.1 Generate automatic descriptions for each latents

We generate natural language descriptions enabling humans to intuitively understand the semantic meaning of each latent concept. Specifically, for each latent concept in the SAE k=32 trained in Section 3 (which showed the best performance on latent intrusion test), we collect MSMARCO passages that most activate the corresponding latent concept and instruct a LLM to summarize the common themes, concepts, or characteristics shared across these passages. Figure 2 shows how a decomposition of a document into generated latent descriptions looks like.

4.2 Interpreting Dense Embeddings via Latent Descriptions

In this section, we evaluate whether latent descriptions can be used to interpret the semantic information embedded within DPR model's dense embeddings. To this end, we decompose a passage's dense embedding into latent components and assess whether the corresponding latent descriptions enable accurate identification of the original passage. Specifically, each human annotator is pre-

[4]We lay out detailed experimental settings in Appendix B.

Task	Accuracy
Embedding Interpretability	
Passage identification	0.943
Ranking Interpretability	
R@P vs R@P	0.903
R@P vs NR@P	0.938
R@N vs NR@P	0.921

Table 3: Human annotator accuracy on interpretability tasks: Embedding Interpretability (§ 4.2) and Ranking Interpretability (§ 4.3). (Note: R@P = Retrieved Positive; NR@P = Not Retrieved Positive; R@N = Retrieved Negative.)

sented with one target passage, nine randomly sampled distractor passages, and a set of latent concepts extracted from the target with their activation strengths. Annotators are then asked to identify the target passage based on the provided descriptions and activation strength.

Since the latent concepts used in this study are obtained via unsupervised training of SAE reconstructing DPR model embeddings, their activation frequency across passages varies significantly (Figure 5). In particular, high frequency latents tend to be abstract and less interpretable. To address this skewed distribution and to better capture the semantic importance of each latent, we adjust each latent's activation strength with its Inverse Document Frequency (IDF), as defined in Eq. 6. We provide illustrative examples comparing latent descriptions with and without IDF weighting in Appendix D.1.

We randomly sampled 600 passages from the MSMARCO corpus and evaluated the embedding interpretability task and report the accuracy in Table 3. The high accuracy (0.943) confirms that latent descriptions effectively reveal the semantic content encoded in the DPR model's dense representations.

4.3 Interpreting Ranking result via Latent Descriptions

We evaluate the interpretability of the DPR model's similarity scoring process through latent descriptions. Specifically, we assess whether humans can simulate the ranking behavior of the DPR model. We provide human annotators two candidate passages for a given query, along with the latent descriptions and IDF-adjusted activation values extracted from each passage and query. Then they are tasked to choose from which the two documents the DPR model would have ranked higher.

We evaluate the model simulatability across three specific settings of interest:

1. Retrieved Positive **vs** Retrieved Positive.
2. Retrieved Positive **vs** Not Retrieved Positive.
3. Retrieved Negative **vs** Not Retrieved Positive.

The experiments were conducted using queries from the TREC-DL 2019 and 2020 datasets. We define documents ranked within the top 1000 as "retrieved", and those ranked lower as "not retrieved". For each query, the official gold document is considered "positive", while all others are considered "negative"

As shown in Table 3, human annotators achieve high accuracy in each settings of interest showing accuracy higher than 0.9, confirming that explanations from our framework indeed help human annotators to simulate model predictions more accurately.

5 Concept-Level Sparse Retrieval

Now we extend the sparse autoencoder framework to the sparse retrieval domain. We propose Concept-Level Sparse Retrieval (CL-SR), a novel approach that treats each latent concept as a fundamental unit of retrieval. By replacing lexical terms with these semantically coherent and generalized latent concepts, CL-SR enables inverted index retrieval from any dense retrieval model. CL-SR offers **Semantic generalization** and **Computational efficiency**. We further analyse these qualities in Section 5.1, 5.2.

In CL-SR, the scoring between a query and a passage is computed using the following formulation (Eq. (5)).

$$s(q,d) = \sum_{i \in q \cap d} f_d(q,i) \cdot f_d(d,i) \cdot \mathrm{idf}(i) \quad (5)$$

where

$$\mathrm{idf}(i) = \log \frac{|D|}{1 + \left|\left\{d \in D \mid z(h)_{d,i} > 0\right\}\right|} \quad (6)$$

$$f_q(q,i) = \frac{z(h)_{q,i}(1+k_2)}{z(h)_{q,i} + k_2},$$

$$f_d(d,i) = \frac{z(h)_{d,i}(1+k_1)}{z(h)_{d,i} + k_1\left(1 - b + b\frac{\|z(h)_d\|_1}{\sum_{d \in D}\|z(h)_d\|_1/|D|}\right)}.$$

Model	MSMARCO dev		TREC DL 2019	TREC DL 2020	Efficiency			
	MRR@10	Recall@1k	NDCG@10	NDCG@10	FLOPs	Avg. D Len	Storage (GB)	Vocab Size
(Unsupervised) Sparse Retrieval								
BM25	0.183	0.853	0.506	0.478	0.13	39.41	0.67	2,660,824
RM3	0.165	0.870	0.522	0.489	–	39.41	0.67	2,660,824
docT5query	0.277	0.947	0.626	0.607	0.94	756.62	0.98	2,660,824
(Neural) Sparse Retrieval								
query2doc	0.214	0.918	0.635	0.582	0.87	39.41	0.67	2,660,824
DeepImpact	0.327	0.947	0.657	0.603	–	71.61	1.40	3,514,102
uniCOIL	0.315	0.924	0.643	0.652	1.03	67.96	1.30	30,522
SPLADE-max	0.340	0.965	0.683	**0.671**	1.35	91.50	2.60	30,522
CL-SR (Efficient)	0.343	0.954	0.643	0.593	**0.11**	**22.95**	**0.57**	**11,709**
CL-SR (Max)	**0.368**	**0.969**	**0.686**	0.634	0.74	64.73	1.27	18,679

Table 4: Comparison of retrieval performance and efficiency across unsupervised, neural, and our CL-SR models on MSMARCO dev, TREC DL 2019, and TREC DL 2020. FLOPs measured by following the settings of (Formal et al., 2021b). query2doc performance copied from (Wang et al., 2023b). All other metrics measured using Pyserini (Lin et al., 2021).

This equation replaces the traditional BM25 term frequency with latent activation values and redefines document length normalization based on the total activation in latent space. Here, $z(h)_{q,i}$ and $z(h)_{d,i}$ denote the activation values of latent concept i in the query and passage, respectively, and D denotes the entire set of documents.

As discussed in Section 4.2, high-frequency latents tend to contribute less meaningfully to information retrieval. To mitigate their influence, we apply IDF weighting[5].

Implementation and evaluation setup

We conducted retrieval experiments using SAEs trained in section 3.1. Among them, we utilze two configurations for comparison: k=32 (Efficient) and k=128 (Max). The Efficient model is optimized for computational efficiency whereas the Max model prioritizes retrieval accuracy by allowing a greater number of latent concepts to be used as document identifiers.

Similar to traditional sparse retrieval methods, CL-SR allows for the construction of an inverted index in advance, based on indices of latent concepts activated from the each document in the collection. To reduce storage and retrieval overhead, we index only a fixed number of highly activated latent concepts per passage by setting maximum number of allowed latents for each passage rather than indexing all the activated latents. Detailed hyperparameter settings and results for other SAE variants are provided in the Appendix E.

At retrieval time, only the query embedding is projected into the latent space, and its sparse latent activations are directly used to compute document rankings using the Eq.(5). To assess retrieval efficiency, we measure: (1) FLOPs, defined as the expected number of floating-point operations per query–document pair; (2) the average number of activated latents per passage (analogous to number of tokens in term-based retrieval); and (3) index storage size. Specifically, FLOPs are computed as $\mathbb{E}_{q,d}\left[\sum_{j\in V} p_j^{(q)} \cdot p_j^{(d)}\right]$, where V denotes the vocabulary, and p_j is the activation probabilities for token j in document d and query q respectively. Following prior work (Formal et al., 2021b) this metric is computed over a set of approximately 100k queries, on the MSMARCO collection. For evaluation, we use the same datasets as in Section 3.2

Baselines

We categorize the baseline models into two major groups: Unsupervised Sparse Retrieval and Neural Sparse Retrieval. The Unsupervised group includes classical term frequency–based methods such as BM25, as well as its extensions via query expansion (RM3) (Lavrenko and Croft, 2001) and passage expansion (docT5query) (Nogueira and Lin, 2019). These methods do not involve any neural network at inference time. In contrast, the Neural Sparse Retrieval group employs neural networks—typically PLMs—at inference time to dynamically reweight or expand query and document terms. Specifically, we include query2doc (Wang et al., 2023b), DeepImpact (Mallia et al., 2021), uniCOIL (no expansion) (Lin and Ma, 2021), and SPLADE v2 (max) (Formal

[5]The impact of IDF weighting on CL-SR is provided in the Appendix D.2

et al., 2021a) as representative neural approaches.

While more recent variants such as Distil-SPLADE and SPLADE v3 (Lassance et al., 2024) achieve stronger performance through heavy distillation and advanced training techniques (e.g., ensemble teacher re-rankers, model-specific hard negative sampling), these enhancements are orthogonal to the focus of our work. To ensure a fair comparison, we report non-distilled SPLADE v2-max as a baseline.

5.1 Retrieval Effectiveness and Efficiency

As shown in Table 4, the CL-SR framework demonstrates retrieval accuracy on par with other neural sparse retrieval baselines while achieving superior computational efficiency. On MSMARCO Dev, SAE-Max achieves the highest performance with MRR@10 of 0.368 and Recall@1k of 0.969. SAE-Efficient matches the performance of SPLADE v2 with an MRR@10 of 0.343, but requires only 0.11 FLOPs per query and 0.57 GB of index storage. We believe this efficiency arises from representing documents with a compact set of semantically abstracted latent concepts, which substantially reduces the cost of query-document matching.

On TREC-DL 2019/2020, models based on exact lexical matching are more favorable since most queries are long-tailed and entity-centric (Wang et al., 2023b). Nevertheless, SAE-Max still achieves competitive performance with nDCG@10 scores of 0.686 (2019) and 0.634 (2020).

Figure 3 illustrates the trade-off between retrieval effectiveness (MRR@10) and computational cost (FLOPs) on the MS MARCO Dev set. The performance of CL-SR shows diminishing returns as the maximum number of allowed latent concepts per passage increases. To balance efficiency and effectiveness, we configure CL-SR Efficient and CL-SR Max to index up to 24 and 65 latent concepts per passage, respectively, resulting in average document lengths (i.e., number of active latents) of 22.95 and 64.73.

5.2 Robustness Analysis

CL-SR computes query–document similarity not at the lexical term level, but in a semantically generalized latent concept space. Each latent concept is a learned representation of baseline model that can cluster lexically diverse yet semantically related expressions into a single discrete unit. This design preserves the semantic expressiveness, allowing CL-SR to remain robust to vocabulary mismatch and semantic mismatch.

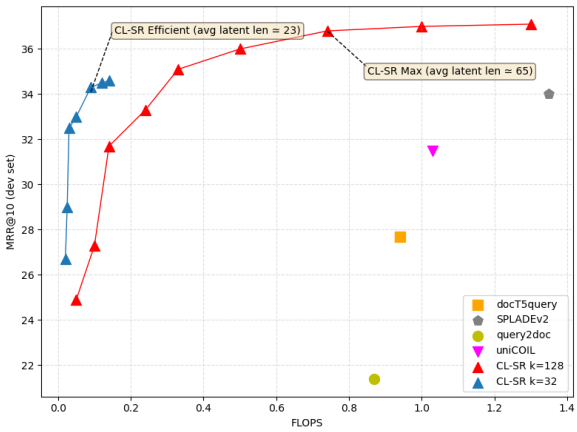

Figure 3: Performance vs FLOPs for CL-SR with varying number of latent concepts used for each document when indexing.

Models	MRR@10	Retrieval Type
BM25	0.0 (-100%)	Unsupervised
docT5query	0.052 (-80.8%)	
DeepImpact	0.094 (-71.2%)	Neural Sparse
SPLADEv2	0.106 (-68.9%)	
CL-SR (Efficient)	0.124 (-63.7%)	
CL-SR (Max)	**0.143 (-61.1%)**	
SimLM (baseline)	0.185 (-55.0%)	Dense

Table 5: MRR@10 performance on MSMARCO Dev queries where BM25 fails to retrieve the gold passage within the top-1000 results. Relative performance drops are shown in parentheses.

5.2.1 Robustness on vocabulary/semantic mismatch

To empirically verify this robustness, we construct a *Mismatch Set* from the MS MARCO dataset by selecting only the queries for which BM25 fails to retrieve the gold passage within top-1000 results among 8.84 million candidate passages. This subset comprises 988 queries out of the total 6,980 Dev queries, representing failure cases where traditional term-based sparse retrieval is likely to fail due to lexical or semantic discrepancies between queries and relevant documents[6].

Experimental results (Table 5) show that while traditional sparse models exhibit a significant performance drop on the Mismatch Set, CL-SR effectiveness remains relatively stable. Notably, SAE-Efficient demonstrates stronger robustness compared to SPLADEv2, and SAE-Max achieves the highest robustness among sparse models and is on

[6]Additional evaluations based on failure cases defined at top-10 and top-100 are reported in Appendix F.

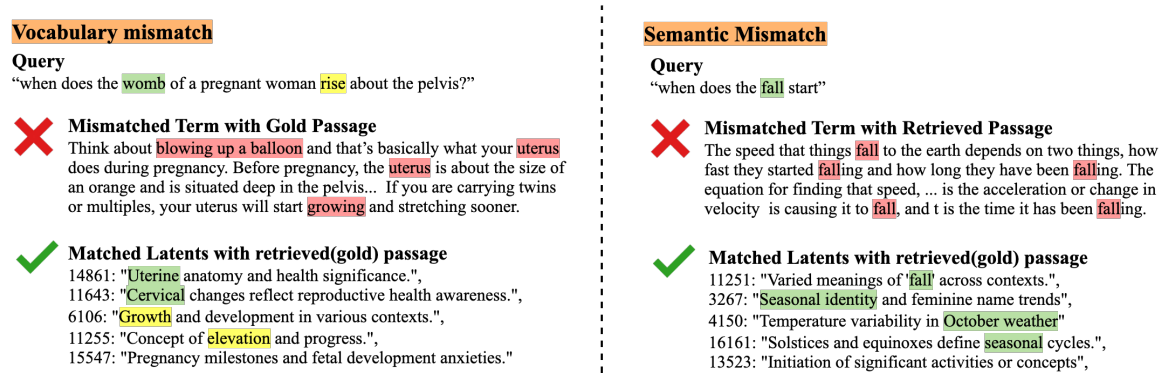

Figure 4: Qualitative case study illustrating CL-SR's ability to handle vocabulary and semantic mismatch.

par with its target model (SimLM). These findings suggest that latent concepts operate as a semantically rich and generalizable retrieval unit, capable of bridging vocabulary and semantic gaps without relying on query or document expansion mechanisms.

5.2.2 Case study

To further investigate how CL-SR addresses mismatch scenarios in practice, we conduct a qualitative analysis on the Mismatch Set. Figure 4 provides case-studies that illustrate how CL-SR effectively handles both vocabulary mismatch and semantic mismatch conditions.

Vocabulary Mismatch. In Figure 4 left, the query contains the term "womb", while the corresponding gold passage uses "uterus". Moreover, the term "rise" in the query is expressed metaphorically in the passage as "blowing up a balloon". These lexical variations caused BM25 to fail due to the absence of exact term overlap. In contrast, CL-SR captures mutually activating concepts such as latent 14861 (Uterine anatomy and health significance), 11643 (Cervical changes), and 6106 (Growth and development). These shared activations enable CL-SR to retrieve the correct passage by aligning abstract concepts, rather than relying on lexical overlap or external term expansion.

Semantic Mismatch. In Figure 4 right, the query involves the term "fall", which can be pragmatically inferred to mean "autumn" from the query itself. However, BM25 assigns high relevance score to an unrelated document where "fall" appears multiple times but in the physical sense (i.e., to drop). CL-SR overcomes this by distributing semantic meaning across multiple latent concepts, such as latent 11251 (Varied meanings of "fall") and 3267 (Seasonal identity), thereby capturing the intended sense of the query and retrieving the correct document.

These case studies highlight CL-SR's ability to generalize beyond exact lexical term matching by leveraging contextual and conceptual representations, overcoming structural limitations inherent in term-based retrieval frameworks.

6 Conclusion

This study proposes a novel interpretability framework for DPR models by leveraging SAEs to decompose dense embeddings—previously considered uninterpretable—into semantically distint latent concepts. Through extensive experiments, we demonstrate that the latent concepts effectively preserve the information contained in the original embeddings while functioning as interpretable semantic units. Building on this, we empirically validate the explainability of both the DPR models embeddings and the similarity scoring process. Additionally, we introduced CL-SR, a novel retrieval paradigm that integrates the semantic expressiveness of dense retrieval with the efficiency and interpretability of sparse retrieval. The proposed CL-SR maintains retrieval accuracy comparable to traditional term-based sparse methods while achieving superior efficiency in terms of both computational cost and storage requirements. Notably, it exhibits strong robustness in challenging scenarios involving vocabulary and semantic mismatches. We believe our framework provides a new method to gain insights on dense passage retrieval process that can be used to improve dense retrieval or be leveraged to enhance neural sparse retrievals.

Limitations

Known issues of SAE Though SAEs are the most popular unsupervised decomposition methods in

interpretability, they pose substantial practical and conceptual limitations(Sharkey et al., 2025). Notably, features found by SAEs are known to be incomplete(Leask et al., 2025; Templeton et al., 2024; Robert_AIZI, 2024) and dataset dependent(Kissane et al., 2024) and we believe these problems to be present in our SAE as well. Although recent studies try to address these issues by modifying SAE architectures, we stick to the widely used BatchTopK SAE and leave application of newer variants to future work.

Evaluation of XAI Due to the lack of established baselines for explaining ranking models, we primarily rely on LLMs for the evaluation of explanations by instructing them to perform proxy tasks. Though utilizing LLMs for evaluation is widely utilized in SAE literatures, their sensitivity to prompts and variability in reasoning limits the reliability of the evaluation. To compensate, we conduct a human study designed to simulate the ranking model's predictions. This evaluation method requires human annotators and is therefore labor-intensive and difficult to scale, which limited us to evaluating a sampled subset of the data.

Acknowledgments

This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00350379, 30), Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00369, RS-2022-II220369, 30), Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2019-II190421, Artificial Intelligence Graduate School Program(Sungkyunkwan University),20), and ICT Creative Consilience Program through the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2020-II201821, 20).

License

This work utilizes the *Pyserini* toolkit (Lin et al., 2021), an open-source Python framework for reproducible information retrieval research with sparse and dense representations, released under the Apache License 2.0.

We also make use of the *MS MARCO* datasets, provided by Microsoft for non-commercial research purposes only. The dataset is distributed "as is" and subject to Microsoft's Terms and Conditions.[7]

References

Avishek Anand, Lijun Lyu, Maximilian Idahl, Yumeng Wang, Jonas Wallat, and Zijian Zhang. 2022. Explainable information retrieval: A survey. *arXiv preprint arXiv:2211.02405*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*. Version 3.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*.

Bart Bussmann, Patrick Leask, and Neel Nanda. 2024. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*. ArXiv:2412.06410 [cs.LG].

Jonathan Chang, Jordan Boyd-Graber, Sean Gerrish, Chong Wang, and David M. Blei. 2009. Reading tea leaves: how humans interpret topic models. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems*, NIPS'09, page 288–296, Red Hook, NY, USA. Curran Associates Inc.

Nick Craswell, W. Bruce Croft, Maarten de Rijke, Jiafeng Guo, and Bhaskar Mitra. 2018. Neural information retrieval: Introduction to the special issue. *Information Retrieval Journal*, 21(2):107–110.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2021. Overview of the trec 2020 deep learning track. *arXiv preprint arXiv:2102.07662*. Version 1.

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*. Version 2.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain,

[7] https://microsoft.github.io/msmarco/

Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Transformer Circuits Thread*.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021a. Splade v2: Sparse lexical and expansion model for information retrieval. *Preprint*, arXiv:2109.10086. ArXiv:2109.10086 [cs.IR].

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021b. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2288–2292. ACM.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. 2025. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. Coil: Revisit exact lexical match in information retrieval with contextualized inverted list. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*, pages 113–122, New York, NY, USA. Association for Computing Machinery.

Chao-Wei Huang and Yun-Nung Chen. 2024. Pairdistill: Pairwise relevance distillation for dense retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18225–18237, Miami, Florida, USA. Association for Computational Linguistics.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. 2024. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*.

Hao Kang, Tevin Wang, and Chenyan Xiong. 2025. Interpret and control dense retrieval with sparse latent features. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 700–709, Albuquerque, New Mexico. Association for Computational Linguistics.

Connor Kissane, robertzk, Neel Nanda, and Arthur Conmy. 2024. Saes are highly dataset dependent: a case study on the refusal direction.

Carlos Lassance, Hervé Déjean, Thibault Formal, and Stéphane Clinchant. 2024. Splade-v3: New baselines for splade. *arXiv preprint arXiv:2403.06789*.

Victor Lavrenko and W Bruce Croft. 2001. Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 120–127. ACM.

Patrick Leask, Bart Bussmann, Michael T Pearce, Joseph Isaac Bloom, Curt Tigges, Noura Al Moubayed, Lee Sharkey, and Neel Nanda. 2025. Sparse autoencoders do not find canonical units of analysis. In *The Thirteenth International Conference on Learning Representations*.

Minghan Li, Sheng-Chieh Lin, Barlas Oguz, Asish Ghoshal, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2022. Citadel: Conditional token interaction via dynamic lexical routing for efficient and effective multi-vector retrieval. *arXiv preprint arXiv:2211.10411*.

Jimmy Lin and Xueguang Ma. 2021. A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques. *Preprint*, arXiv:2106.14807. ArXiv:2106.14807 [cs.IR].

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*, pages 2356–2362, Online. Association for Computing Machinery.

Michael Llordes, Debasis Ganguly, Sumit Bhatia, and Chirag Agarwal. 2023. Explain like i am bm25: Interpreting a dense model's ranked-list with a sparse approximation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1976–1980. ACM.

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

Antonio Mallia, Omar Khattab, Torsten Suel, and Nicola Tonellotto. 2021. Learning passage impacts for inverted indexes. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*, pages 1723–1727, New York, NY, USA. Association for Computing Machinery.

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2025. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations*.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9844–9855, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to doctttttquery. *arXiv preprint arXiv:1904.08375*.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*. Https://distill.pub/2020/circuits/zoom-in.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. 2024. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *Preprint*, arXiv:2407.14435.

Robert_AIZI. 2024. Research report: Sparse autoencoders find only 9/180 board state features in othellogpt.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, Stephen Casper, Max Tegmark, William Saunders, David Bau, Eric Todd, Atticus Geiger, Mor Geva, Jesse Hoogland, Daniel Murfet, and Tom McGrath. 2025. Open problems in mechanistic interpretability. *Preprint*, arXiv:2501.16496.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*.

Michael Völske, Alexander Bondarenko, Maik Fröbe, Benno Stein, Jaspreet Singh, Matthias Hagen, and Avishek Anand. 2021. Towards axiomatic explanations for neural ranking models. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 13–22. ACM.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2023a. Simlm: Pre-training with representation bottleneck for dense passage retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2244–2258, Toronto, Canada. Association for Computational Linguistics.

Liang Wang, Nan Yang, and Furu Wei. 2023b. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423, Singapore. Association for Computational Linguistics.

Ran Xu, Wenqi Shi, Yue Yu, Yuchen Zhuang, Yanqiao Zhu, May Dongmei Wang, Joyce C. Ho, Chao Zhang, and Carl Yang. 2024. Bmretriever: Tuning large language models as better biomedical text retrievers. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22234–22254, Miami, Florida, USA. Association for Computational Linguistics.

Christine Ye, Charles O'Neill, John F Wu, and Kartheik G. Iyer. 2024. Steering semantic search with interpretable features from sparse autoencoders. In *MINT: Foundation Model Interventions*.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, M. Zhang, and Shaoping Ma. 2021. Interpreting dense retrieval as mixture of topics. *ArXiv*, abs/2111.13957.

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.

A SAE training Details

We trained our SAE on the MSMARCO corpus and its training queries, which were preprocessed into input sequences of maximum 144 and 32 tokens each for input into our baseline model. Training was performed with a learning rate of 5×10^{-5}, a batch size of 4096, and for 100 epochs. Training a single SAE model in this setup took about 350 minutes on a single RTX 3090 GPU.

All models were trained using the AdamW (Loshchilov and Hutter, 2019) optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 6 \times 10^{-10}$. At each training step, the decoder weights are normalized to 1 following Bricken et al. (2023).

We follow Gao et al. (2025)'s method for the auxiliary loss. The total loss is defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L} + \lambda \mathcal{L}_{\text{aux}}, \quad \text{where } \lambda = 0.0625.$$

We consider neurons inactive for 20 training steps as dead and use top 2*k dead neurons for the dead reconstruction.

B Experimental Details

In this appendix, we provide details about human evaluation, dataset, sampling, LLM prompts of our experiments.

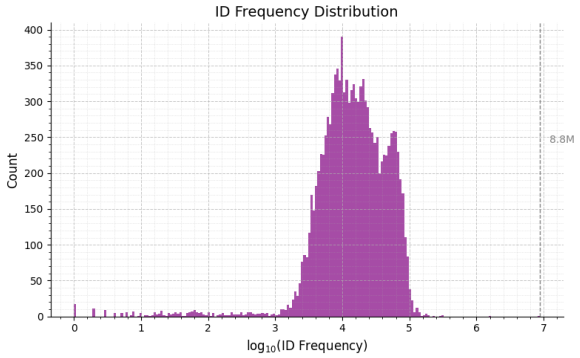

Figure 5: Frequency distribution of SAE latents across documents.

Dataset	R@P vs R@P	R@P vs NR@P	NR@P vs R@N
TREC DL 2019	104,529	196,110	1,004,471
TREC DL 2020	63,035	84,767	1,099,965

Table 6: Document pair statistics for ranking interpretation on TREC DL 2019 and 2020.

B.1 Description generation details

For BatchTopK SAE, at inference time, latents with activation value larger than mean of top k-th activation over the whole dataset is considered "activated". For each latent that has been activated at least once over the whole MSMARCO corpus, we provide top 30 most activating MSMARCO passage and instruct GPT4.1-mini to generate description for the latent. We use the prompt of Figure 6 (b) for the instruction.

B.2 Dense Embedding Interpretation Details

For each 600 randomly sampled documents of MSMARCO corpus, we make a set of feature descriptions. Then, for each document, we additionally sample 9 random documents to choose our target document from. We instruct 6 graduate students in machine learning (not authors of this paper) to identify the target passage and report accuracy over 600 samples. We provide a concrete example of the task in Figure 8

B.3 Ranking Interpretation Details

We run SimLM (target model of our SAEs) on TREC 2019, 2020 yielding total combination of document pairs as (Table 6). From each set, we randomly sample 100 pairs, resulting in a total of 600 pairs. We then distribute 600 total prompts to 6 graduate students in machine learning (not authors of this paper) to predict which of the two given documents would have been ranked higher by the target model given latent concepts extracted. We provide a concrete example of the task in Figure 9

C Generalizability

To evaluate the generalizability of the Sparse Autoencoder (SAE) beyond the training distribution, we conduct two sets of transfer experiments, summarized in Table 7 and Table 8.

SAE Generalization Across Baseline DPR Models. Table 7 evaluates the robustness of the SAE when applied to different dense retrievers. Keeping the training settings and MSMARCO data fixed, we replace the SimLM encoder with two off-the-shelf DPR variants—TAS-B(Hofstätter et al., 2021) and GTR-T5(Ni et al., 2022)—and retrain the SAE. This allows us to test whether the proposed framework is model-agnostic.

SAE Transfer to Unseen Datasets. In Table 8, We assess whether our SAE trained on MSMARCO passage embeddings from the SimLM model retains effectiveness on unseen datasets. Since computing Spearman's correlation over the full ranking is costly, we restrict the evaluation to datasets of manageable size: TREC-COVID (50 queries, 171k passages) and NFCorpus (323 queries, 3.6k passages).

D Impact of IDF Weighting

Figure 5 shows that the SAE(k=32) latents follow a heavy-tail distribution across documents: highly frequent latent concepts are more abstract (and less informative), whereas low-frequency latent concepts capture specific semantics.

D.1 Interpretability Case Study

Figure 7 illustrates a case study on an MSMARCO passage describing cost-of-living and inflation benchmarks in the 1960s, showing a subset of latent concepts extracted from the passage's dense embedding. When no IDF weighting is applied (W/o IDF), high-frequency, semantically abstract latents—e.g. latent 8033 ("Definitions and concept distinctions in text")—dominate the activation ranking and obscure more discriminative features. After applying IDF weighting (W/ IDF), these common latents are weakened and truly informative concepts emerge: latent 8983 ("Historical financial and social benchmarks circa 1960") rises to the top, and latents 21047 ("Income-related financial and economic concepts") and 20216 ("1960s–70s cultural and fashion evolution") receive substantially

Model	NMSE	TREC DL 2019			TREC DL 2020		
		NDCG@10	Recall@1k	Spearman	NDCG@10	Recall@1k	Spearman
Baseline (TAS-B)	–	0.721	0.783	–	0.685	0.800	–
Reconstructed (k=32)	0.2106	0.573 (×0.80)	0.685 (×0.88)	0.865 $\pm 3.4E-2$	0.517 (×0.76)	0.676 (×0.85)	0.861 $\pm 3.6E-2$
Reconstructed (k=48)	0.2005	0.575 (×0.80)	0.690 (×0.88)	0.894 $\pm 2.1E-2$	0.523 (×0.76)	0.700 (×0.88)	0.891 $\pm 2.1E-2$
Reconstructed (k=64)	0.1610	0.593 (×0.82)	0.695 (×0.89)	0.928 $\pm 1.8E-2$	0.565 (×0.83)	0.716 (×0.90)	0.925 $\pm 1.7E-2$
Reconstructed (k=128)	0.1188	**0.626** (×0.87)	**0.735** (×0.94)	**0.934** $\pm 1.4E-2$	**0.604** (×0.88)	**0.753** (×0.94)	**0.932** $\pm 1.4E-2$
Baseline (GTR-T5)	–	0.687	0.737	–	0.664	0.721	–
Reconstructed (k=32)	0.1976	0.527 (×0.77)	0.617 (×0.84)	0.907 $\pm 1.4E-2$	0.469 (×0.71)	0.630 (×0.87)	0.908 $\pm 1.2E-2$
Reconstructed (k=48)	0.1675	0.565 (×0.82)	0.631 (×0.86)	0.926 $\pm 1.1E-2$	0.547 (×0.82)	0.650 (×0.90)	0.926 $\pm 9.3E-3$
Reconstructed (k=64)	0.1474	0.592 (×0.86)	0.647 (×0.88)	0.937 $\pm 1.0E-2$	0.568 (×0.85)	0.666 (×0.89)	0.937 $\pm 8.3E-3$
Reconstructed (k=128)	0.1364	**0.590** (×0.86)	**0.687** (×0.93)	**0.959** $\pm 7.3E-3$	**0.614** (×0.93)	**0.685** (×0.95)	**0.958** $\pm 6.7E-3$

Table 7: Quantitative evaluation of the SAE trained on TAS-B and GTR-T5 embeddings with the same settings (k=32,48,64,128) and training data as used for SimLM, demonstrating its generalization performance across different DPR models. Stat. sig. difference w/ paired t-test ($p < 0.05$).

Model	TREC-COVID			NFCorpus		
	MRR@10	Recall@1k	Spearman	MRR@10	Recall@1k	Spearman
Baseline (SimLM)	0.828	0.237	–	0.498	0.592	–
Reconstructed (k=32)	0.692 (×0.83)	0.188 (×0.79)	0.887 $\pm 3.3E-2$	0.444 (×0.89)	0.567 (×0.96)	0.806 $\pm 2.7E-2$
Reconstructed (k=48)	0.704 (×0.85)	0.190 (×0.80)	0.891 $\pm 3.5E-2$	0.446 (×0.89)	0.569 (×0.96)	0.809 $\pm 2.9E-2$
Reconstructed (k=64)	0.728 (×0.88)	0.193 (×0.82)	0.914 $\pm 3.2E-2$	0.457 (×0.92)	0.575 (×0.97)	0.851 $\pm 6.8E-2$
Reconstructed (k=128)	0.733 (×0.89)	0.216 (×0.92)	0.930 $\pm 2.8E-2$	0.467 (×0.94)	0.571 (×0.96)	0.896 $\pm 2.0E-2$

Table 8: Zero-shot evaluation of the SAE trained on MSMARCO SimLM embeddings, showing its generalization performance on unseen datasets (TREC-COVID and NFCorpus). Stat. sig. difference w/ paired t-test ($p < 0.05$).

higher activation ranks compared to the unweighted case, thereby enhancing the interpretability of the embedding.

D.2 CL-SR Performance

To quantify the effect of applying Inverse Document Frequency (IDF) to latent activations, we compare two scoring variants:

- **w/ IDF**: activations are scaled by their IDF (Eq. 5), and
- **w/o IDF (Dot-product)**: raw activation values are used without any weighting.

Table 9 reports the relative performance drop of the dot-product variant (shown in gray) compared to CL-SR. The results show that applying IDF weighting consistently improves retrieval performance by reducing the influence of overly frequent latents.

	w/ IDF	w/o IDF(Dot-product)
TREC-DL 2019 & 2020 NDCG@10		
CL-SR$_{\text{efficient}}$	0.643 / 0.593	0.530(-15.61%) / 0.494(-17.94%)
CL-SR$_{\text{max}}$	0.686 / 0.634	0.587(-13.29%) / 0.562(-10.94%)
MSMARCO Dev MRR@10 & Recall@1k		
CL-SR$_{\text{efficient}}$	0.343 / 0.954	0.274(-19.65%) / 0.908(-4.52%)
CL-SR$_{\text{max}}$	0.368 / 0.969	0.316(-14.13%) / 0.934(-3.61%)

Table 9: Retrieval performance comparison between IDF-weighted activations and without IDF (Dot-product) activations. Relative performance drops are shown in parentheses.

E Impact of Latent Count on CL-SR Performance

We study how the number of latent concepts k affects both effectiveness and hyperparameter settings of Concept-Level Sparse Retrieval (CL-SR). We vary k over {32, 48, 64, 128}, tuning the BM25-style parameters (k_1, b, k_2) on the MSMARCO Dev set for each configuration.

Table 10 reports MRR@10 and Recall@1k on MSMARCO Dev, as well as NDCG@10 on TREC-DL 2019/2020. As k increases from 32 to 128, we observe a steady improvement in all metrics: MRR@10 rises from 0.343 to 0.368, Recall@1k from 0.954 to 0.969, and NDCG@10 on TREC-DL from 0.643/0.593 to 0.686/0.634. These gains confirm that the number of latents used for reconstruction increases, the original embedding information can be recovered more faithfully.

F Robustness Test on Mismatch Set

To complement the main analysis based on top-1000 failures, we additionally evaluate retrieval

# of Latents	MSMARCO Dev MRR@10	TREC DL 2019 NDCG@10	TREC DL 2020 NDCG@10	Hyperparameters (k_1, b, k_2)
32	0.343	0.643	0.594	0.6, 1.75, 2.5
48	0.353	0.646	0.621	0.6, 1.25, 2.0
64	0.359	0.662	0.603	0.4, 0.75, 2.5
128	0.368	0.686	0.634	0.2, 3.0, 0.5

Table 10: Performance of CL-SR with varying numbers of latent concepts on MSMARCO Dev and TREC-DL (Recall@1k column omitted).

Models	Failures@10	Failures@100
BM25	0.0 (-100%)	0.0 (-100%)
docT5query	0.129 (-52.4%)	0.08 (-70.5%)
DeepImpact	0.192 (-41.2%)	0.123 (-62.4%)
SPLADE-Max	0.218 (-35.8%)	0.149 (-56.1%)
CL-SR (Efficient)	0.226 (-33.9%)	0.163 (-52.3%)
CL-SR (Max)	**0.248** (-32.6%)	**0.185** (-49.7%)

Table 11: MRR@10 performance on MSMARCO Dev queries where BM25 fails to retrieve the gold passage within the top-K candidates. Relative performance drops are shown in parentheses.

performance on subsets where BM25 fails to retrieve the gold passage at top-10 (4.3k queries) and top-100 (2.3k queries) on MSMARCO DEV queries. Table 11 reports the results. We observe that CL-SR (Efficient) and CL-SR (Max) consistently exhibit substantially smaller performance drops.

G Comparison with Neuron-based Interpretability

Previous works like Zhan et al. (2021) has tried to chunk and interpret vector representations in standard basis. We believe this could be problematic since these ranking models are trained with cosine similarity which is rotation invariant. Our method on the other hand does not assume that dimensions have a spatial meaning and instead learns an interpretable basis. Previous works like Olah et al. (2020) also shows that individual neurons (or spatial dimension) are often polysemantic and we believe this holds in our setting as well, motivating our usage of sparse autoencoders. To empirically test this, we performed the latent intrusion test (as described in 4) to the 768-dimensional hidden representation from SimLM. Under the same setting, the accuracy was 0.46, compared to 0.86 with our SAE based method, highlighting a substantial gap in interpretability.

Model	Index (GB)	Factor	Encode Latency (GPU, ms/query)	Search Latency (CPU, ms/query)	Search Latency (GPU, ms/query)
SimLM	26.0	×8.52	**3.42**	924.42	**5.58**
SPLADE	2.6	×0.85	3.63	163.71	—
CL-SR (Max)	1.27	×0.42	3.89	154.25	—
CL-SR (Efficient)	**0.57**	×0.19	3.86	**117.96**	—

Table 12: Efficiency comparison on MSMARCO Dev. Lower is better. Bold indicates the best in each column.

H Efficiency Comparison with Dense Retrieval

To provide a more comprehensive comparison, we additionally report an efficiency analysis against the target dense model, SimLM, in Table 12. We provide our experimental results on the MS-MARCO Dev set following the implementation of in Li et al. (2022) with PyTorch (GPU, Single Nvidia RTX 3090) and Numpy (CPU, Single Intel(R) Core(TM) i7-14700KF CPU @ 3.40 GHz). We set the batch size to 1 and ran all 6,980 MS MARCO Dev queries three times—reporting the lowest observed latency.

I Interpretability Case Study

Table 13 illustrates SAE-based decomposition of SimLM's dense embeddings for the query "which pathogen depends on living cells," one of the top-ranked incorrect passage, and the non retrieved gold passage. All three embeddings activate latent 8327 ("pathogens causing disease in humans via blood"), capturing the generic pathogen concept. However, the retrieved (incorrect) passage uniquely shares immune-system latent 8442 ("immune system influenced by stress and lifestyle") with the query, boosting its matching score to be top-ranked. In contrast, the gold passage activates latent 10944 ("viruses as pseudo-living entities requiring host cells"), which encode the virus-specific host-dependency semantics needed to answer the query; because these latents do not co-occur with the query embedding, SimLM underestimates the passage's relevance. This case study demonstrates that our SAE-based framework can provide a transparent, concept-level diagnosis of retrieval failures.

Source	Text	Top Activated Latents (ID: description (activation))
Query	which pathogen depends on living cells	8327: Pathogens causing disease in humans via blood (5.1931) 8442: Immune system influenced by stress and lifestyle (3.3578) 11271: Living as active state or ongoing condition (3.2910) 4166: Definitions of concepts involving multiple components or functions (2.0142) 8190: Response as reaction, answer, or measured outcome (1.9798) 3580: Multifaceted encyclopedic contexts for “Blood” (1.7898) 1322: Defining unicellular organisms by single-cell simplicity (1.6423)
Retrieved Incorrect Passage	The ability of a multicellular organism to defend itself against invasion by pathogens (bacteria, fungi, viruses, etc.) depends on its ability to mount immune responses.	8306: Definitions of concepts involving multiple components or functions (3.8576) 8442: Immune system influenced by stress and lifestyle (2.9287) 1322: Defining unicellular organisms by single-cell simplicity (2.9263) 8327: Pathogens causing disease in humans via blood (2.8417) 10630: Multifaceted protective actions across legal, psychological, and social domains (1.9795) 6445: Polysemous term "host" across diverse domains (1.9598) 10951: Ability as actualized skill versus potential capacity (1.8653)
Not Retrieved Gold Passage	Yes, viruses need a living host to replicate. A virion needs to be inside a living cell in order to hijack that cell so that more virus particles can be made by the cell. Since a virus is not a living thing, it doesn’t reproduce in the same way.	10944: Viruses as pseudo-living entities with classification challenges (5.6177) 11271: Living as active state or ongoing condition (4.7194) 8327: Pathogens causing disease in humans via blood (3.6580) 6445: Polysemous term "host" across diverse domains (2.7850) 256: DNA semiconservative replication mechanism (2.3768) 15511: Single-parent reproduction, genetic uniformity, rapid population growth (2.3407) 8296: Immune cells engulfing and digesting foreign material (1.6999)

Table 13: Latent concept activations for a failed query from the MS MARCO Dev set (i.e. R@1000 failure), top-ranked (incorrect) passages, and the gold passage. Each latent concept is listed with its idf-scaled activation value in parentheses.

Prompt Template

```
You are an expert linguist analyzing pieces of documents. Below, you will see a set of documents that has some common features, but one of them is an intruder (it does not have that common feature in it).

Your task is to identify the intruder document and explain why it does not fit.

The last line of your response must be the formatted response, using "[intruder]:Document#"

<\nDocument{i} : {passage}>

Which document is the intruder, and why?
```

(a) Prompt template for latent intrusion test.

Prompt Template

```
You are a meticulous AI researcher conducting an important investigation into patterns found in language. Your task is to analyze text and provide an interpretation that thoroughly encapsulates possible patterns found in it.

Guidelines:
You will be given a list of text examples on which a certain common pattern might be present. How important each text is for the pattern is listed after each text.

- Try to produce a concise final description. Simply describe the text latents that are common in the examples, and what patterns you found.
- If the examples are uninformative, you don't need to mention them. Don't focus on giving examples of important tokens, but try to summarize the patterns found in the examples.
- Based on the found patterns, summarize your interpretation in 1-8 words.
- Do not make lists of possible interpretations. Keep your interpretations short and concise.
- The last line of your response must be the formatted interpretation, using [interpretation]:

<Example{i}: {passage} Activation: {act}>
```

(b) Prompt template for generating natural-language descriptions of each latent concept.

Figure 6: LLM prompt templates for our two tasks: (a) latent intrusion test and (b) description generation.

Title: 1960s Important News and Events, Key Technology Fashion and Popular Culture
Context: In 1960 the average income per year was $5,315.00 and by 1969 was $8,540.00. In 1960 a gallon of gas was 25 cents and by 1969 was 35 cents. In 1960 the average cost of new car was $2,600.00 and by 1969 was $3,270.00. A few more prices from the 60's and how much things cost.o provide an estimate of inflation we have given a guide to the value of $100 US Dollars for the first year in the decade to the equivalent in today's money. If you have $100 Converted from 1960 to 2005 it would be equivalent to $679.09 today. In 1960 a new house cost $12,700.00 and by 1969 was $15,500.00.

Extracted latent features	W/ IDF	W/o IDF
8983: "Historical financial and social benchmarks 1960"	**5.801**	0.903
21047: "Contexts of income-related financial and economic"	1.623	0.256
20216: "1960s-70s cultural and fashion evolution"	1.583	0.296
3486: "Fact-based encyclopedic descriptive prose"	0.981	0.555
8033: "Definitions and concept distinctions in text"	0.00	**4.943**

Figure 7: Impact of IDF reweighting on an MSMARCO passage—suppressing abstract latents (8033, 3486) and elevating content-specific latents (8983, 21047, 20216).

features	documents
Anatomical, medical, and symbolic meanings of forehead, Score : 3.15 Anatomical and statistical discourse on penis size, Score : 2.61 Instructional tech and troubleshooting query patterns, Score : 2.04 Standardized medical treatment listing format, Score : 1.79 Physical enlargement due to pathological or physiological causes, Score : 1.75 Ambiguous term referencing band, contracts, and medical contexts, Score : 1.72 Dual focus on bearded dragons and human beards., Score : 1.59 Geographic and demographic data about Reston, Score : 1.54 Place-oriented demographic and geographic descriptions., Score : 1.52 Pregnancy-related acid reflux heartburn patterns, Score : 1.51 Descriptive definitions of -oid terms and legal/medical contexts, Score : 1.44 Factual biological and ecological information about llamas, Score : 1.44 Legal and financial process of property foreclosure, Score : 1.43 Duration variability determined by influencing factors, Score : 1.35 Structured informative content with clear factual descriptions, Score : 1.18 Acne causes, types, and targeted treatments patterns, Score : 1.15 Universal concept of extending limits and adaptability, Score : 1.09 Numbered listicle advice articles with practical guidance, Score : 1.09 Healthcare system identity with employment salary details, Score : 0.99 Informative and descriptive prose across topics, Score : 0.79 Definitions and concept distinctions in explanatory text, Score : 0.00	1 On Earth, volcanism occurs in several distinct geologic settings. Most of these are associated with the boundaries of the enormous rigid plates that make up the lithosphere—the crust and upper mantle.
	Sometimes these side effects are caused by the drug's numbing effect on the area of the brain responsible for pain perception. Lipitor, a commonly prescribed cholesterol medication, is linked to unexpected muscle pain and loss of muscle coordination.
	Become a Certified Safety Professional in 5 Steps. Research what it takes to become a certified safety professional. Learn about education, training and certification requirements to find out if this is the career for you.
	In order to accurately figure out the calories, you need a recipe, or THE recipe, from the actual pizza you are referring to. It depends on the amounts of flour, cheese, tomatoes, and olive oil that are being used. With the information you've provided the experts can only give you a range of calories.
	People also viewed. 1 How to cure bad sunburn on your forehead. 2 How to cure a swollen penis. 3 How long will it take to cure restylane. How to cure swollen 1 foreskin. How long it takes to cure typhoid.
	Report Abuse. They dont. They use it to see if you need to pay a deposit. If you have good credit they dont need a deposit. If bad, then $200. The reason being, they bring equipment that totals around a $1000. They want to make sure you pay for this in their contract.Also, if you pay on time for the first 3 months, they credit your account with the $200.Hope this helps.hey use it to see if you need to pay a deposit. If you have good credit they dont need a deposit. If bad, then $200. The reason being, they bring equipment that totals around a $1000.
	Jake Balloon. Jake inflates himself into the form of a huge ball, KNOCKING BACK and damaging (65 + 0.6 Power Damage) all nearby foes. Giant Jake. Jake grows HUGE and stomps around for 5 seconds, damaging (60 + 0.5 Power Damage/sec.) nearby foes. While in Giant Jake form, Jake cannot make Basic Attacks or use other Powers, but he is immune to controling effects (except KNOCKBACK) and will remove any existing ROOTS or SLOWS when the power is activated.
	San Jose (/ˌsæn hoʊˈzeɪ/; Spanish for Saint Joseph), originally Pueblo de San José de Guadalupe and officially the City of San José, is the third-largest city by population in California, the tenth-largest by population in the United States, and the county seat of Santa Clara County.

Figure 8: Example of human evaluation for dense embeddding interpretability. The left column lists the top-activated latent concepts extracted from a single document, along with their descriptions and activation scores. The right column shows the candidate documents. List of documents are truncated in this figure.

Query1133167 : how is the weather in jamaica	**Document5943911 : Averages for Montego Bay in April. April. Very similar to March's temperature averages, April is warm, with a little more rain possible, as we head towards the start of the wet season. You can expect a pleasant 26°C average during the day, with a high of around 29°C.**	**Document7218021 : Historically, Jamaica's October offers peak rain fall during Jamaica's second raining season. Also Historically, late September into early October is the Caribbean's peak of the Caribbean hurricane season. Early September also shows a storm peak. It will be hot and humid.**
Feature16018 : year-round tropical warm humid climate, Score : 13.24	Feature11644 : April average weather statistics and descriptions, Score : 8.61	Feature16018 : year-round tropical warm humid climate, Score : 9.29
Feature11946 : Monthly seasonal weather summaries with temperature and rainfall trends, Score : 5.06	Feature10567 : Location-focused demographic and geographic summaries, Score : 5.36	Feature15746 : October temperature decline and weather overview, Score : 6.18
Feature17443 : Quantitative facts and contact information presentation, Score : 2.62	Feature13666 : Geographic location demographic and statistical summaries, Score : 5.27	Feature5623 : Structured climatological summaries of September weather, Score : 3.65
Feature18822 : Seasonal regional climate descriptions, Score : 2.10	Feature10256 : Structured factual data on locations and services, Score : 4.41	Feature22466 : Seasonal weather patterns and variability descriptions, Score : 3.23
Feature16188 : Structured daily weather forecast summaries, Score : 1.69	Feature18047 : Detailed March weather summaries with temperature and sunshine data, Score : 3.73	Feature12447 : Informative data-centric descriptions of Caribbean region, Score : 2.94
Feature22466 : Seasonal weather patterns and variability descriptions, Score : 1.61	Feature16018 : year-round tropical warm humid climate, Score : 3.65	Feature19795 : Dual meaning: city name and tropical cyclone, Score : 2.62
Feature22272 : Named entities and titles as questions/exclamations, Score : 1.45	Feature10594 : Standardized travel weather forecast templates, Score : 2.94	Feature15323 : Usage of "peak" to indicate maximum or critical points, Score : 2.43
Feature23006 : Step-by-step functional mechanism explanations, Score : 1.40	Feature13747 : Named entity descriptions and formal definitions, Score : 1.87	Feature4583 : Definitions of postponement and deferral due to rain, Score : 1.76
Feature2806 : Geographic and meteorological summaries of places named Kingston, Score : 1.38	Feature803 : Numerical temperature ranges with contextual details, Score : 1.81	Feature2806 : Geographic and meteorological summaries of places named Kingston, Score : 1.49
Feature18390 : Weather-related media and software entities, Score : 1.37	Feature12967 : Structured climate summaries with temperature statistics, Score : 1.32	Feature17284 : Defining and contextualizing natural and activity seasons, Score : 1.34
Feature6645 : Concise factual explanations with informative clarity, Score : 1.32	Feature21866 : Encyclopedic place and term definitions, Score : 1.20	Feature4710 : August weather summaries by location, Score : 1.31

Figure 9: Example of human evaluation for ranking interpretability. The first column shows the query and its extracted latent features, while the second and third columns list latent features extracted from two candidate documents. Features activated by both the query and each document are highlighted. List of features are truncated in this figure.