SIGIR 2026

Adaptive Sparsity Optimization with Learnable Soft Top-K and Per-Term Thresholding for Efficient Retrieval

Wentai Xie, Parker Carlson, Shanxiu He, Tao Yang

Department of Computer Science
University of California, Santa Barbara

Motivation: Context and Problem

LLM-based sparse representations achieve strong relevance (Lion-SP) but come with overly long vectors

System	Model	Vocab. size	Query len	Doc len
SPLADEv3	BERT	30K	23	170
Lion-SP 1B / 8B	LLaMA 3	128K	293 / 476	1250 / 1457

- –Partially caused by a 4.2× larger vocabulary
- –Imposes serious challenges to retrieval latency and space efficiency

Goal of AdaSparse: optimize model sparsity through a synergy of adaptive pruning strategies

- –Complements the existing training pipeline

Impact of Vocabulary

Characteristics of the LLaMA 3 vocabulary

- –4.2× bigger than BERT's
- –More variants of the same English word
- –Much smaller term weights on average (Lion-SP)

Example query: *"how many years did william bradford serve as governor of plymouth colony?"*

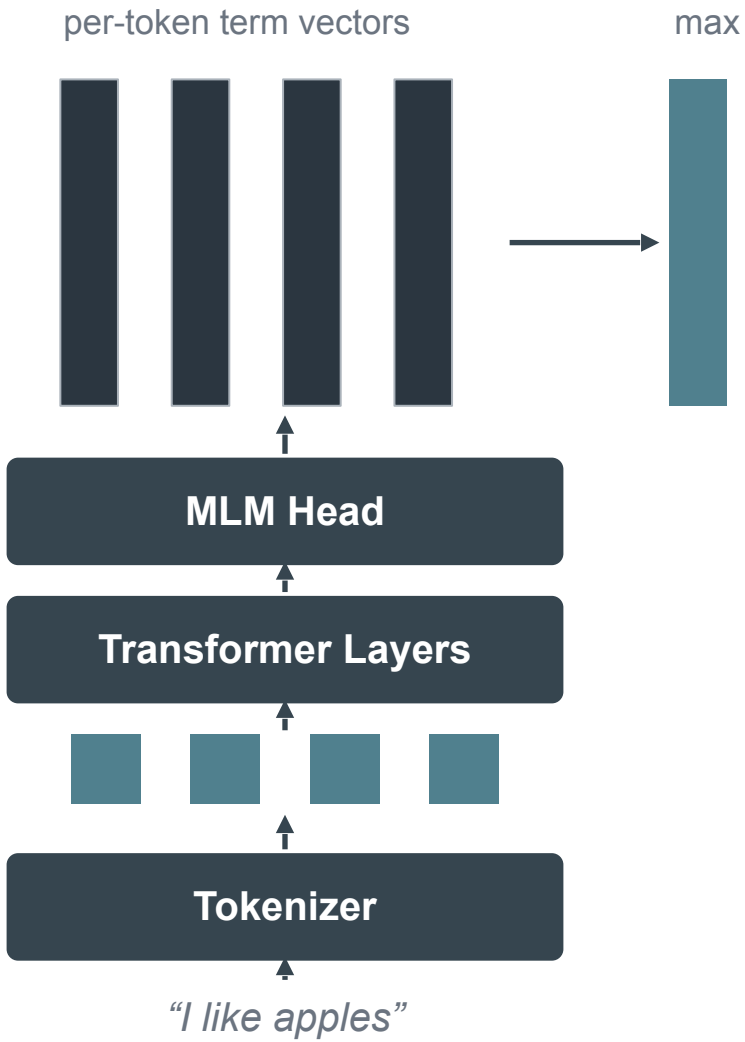
Redundancy and weight range of a generated sparse vector:

Model	Length	#years	#governor	Min-Max	Avg
SPLADE	20	1	2	0.03 - 1.15	0.40
Lion-SP-1B	272	11	8	0.01 - 1.28	0.20
AdaSparse-1B	64	4	3	0.07 - 1.27	0.41

Model Architecture View

SPLADE [SIGIR'21]
- –Uses the MLM head for term expansion and weighting
- –Trained with contrastive loss or knowledge distillation
- –Sparsity regularization with FLOPs

Lion-SP [SIGIR'25]
- –Changes the decoder-only architecture to use bidirectional attention
- –Sparse representation learning similar to SPLADE



AdaSparse: Proposed Sparsification Scheme

Fine-grained, multi-objective pruning through a synergy of adaptive methods:

- **Learnable soft top-K** (soft per-vector limit) + **PTT** (drops low-weight terms)
- Complemented by FLOPs regularization

Method	Strength	Weakness
FLOPs	Rapidly shrinks large, noisy weights	Leaves long tails of small weights
Top-K	Constant, predictable sparsity	K is not learnable; over-prunes complex vectors
Mass ratio pruning	Mass-adaptive sparsity	Mass ratio is not learnable
Uniform thresholding	Filters terms by relative importance	Not adaptive to diverse distributions; empty-vector risk
Per-term thresholding	Individualized and value-adaptive	Struggles with flat distributions and high sparsity
Learnable soft top-K	Adaptive without explicit pruning	Sensitive to noisy term rankings

Learnable Soft Top-K (STop)

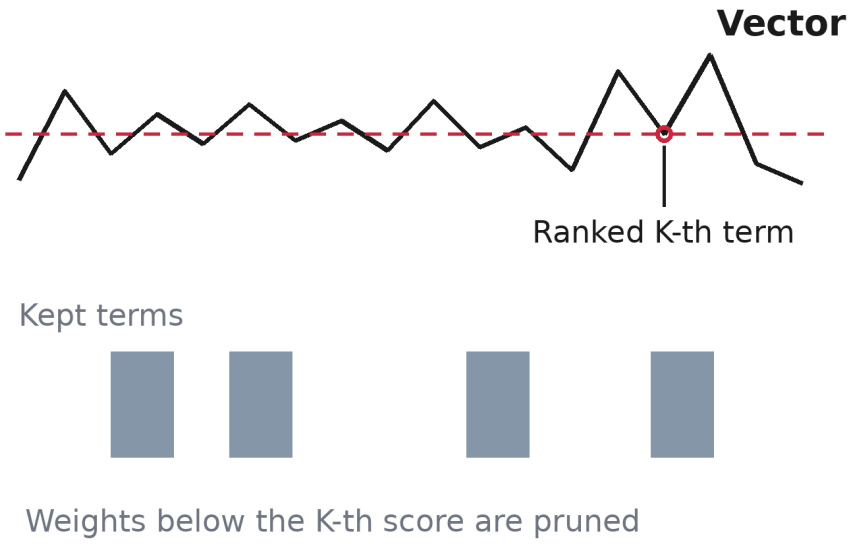
Soft expansion limit: does not explicitly prune K terms during training

Introduces a differentiable signal that penalizes terms beyond K in the loss function

Method:

1. Set the soft limit K = b × original input length
2. Define a gate function for each term
   – Use the K-th ranked term as the pivot to form the mask:

$$G(w_j, v) = \sigma\left(\frac{\widehat{w}_K - w_j - \beta \cdot \mathbb{I}(t_j \in v_{\text{orig}})}{\gamma}\right)$$



Learnable Soft Top-K (STop)

Soft expansion limit: does not explicitly prune K terms during training

Introduces a differentiable signal that penalizes terms beyond K in the loss function

Method:

3. Use a loss function to measure how well the model carries out the pruning task:

$$L_{top} = \sum_{t_j \in V} \left(\frac{\lambda_{qtop}}{|B|} \sum_{q \in B} (w_j^q)^2 G(w_j, q) + \frac{\lambda_{dtop}}{N} \sum_{d=1}^{N} (w_j^d)^2 G(w_j, d) \right)$$

Per-Term Thresholding (PTT)

An individualized, value-adaptive pruning threshold for each term

During document indexing and online query vector generation

–Prune a term weight when it falls below the threshold:

$$\widetilde{w}_j = \begin{cases} w_j, & w_j > \tau_j \\ 0, & w_j \leq \tau_j \end{cases}$$

During training

–Set:

$$\widetilde{w}_j = w_j \cdot \sigma\left(\frac{w_j - \tau_j}{\pi}\right)$$

–Loss function with a learnable threshold per query/document term:

$$L_T = \frac{1}{|V|} \sum_{j \in V} [\log(1 + e^{-\tau_{D,j}}) + \log(1 + e^{-\tau_{Q,j}})]$$

Putting the Techniques Together

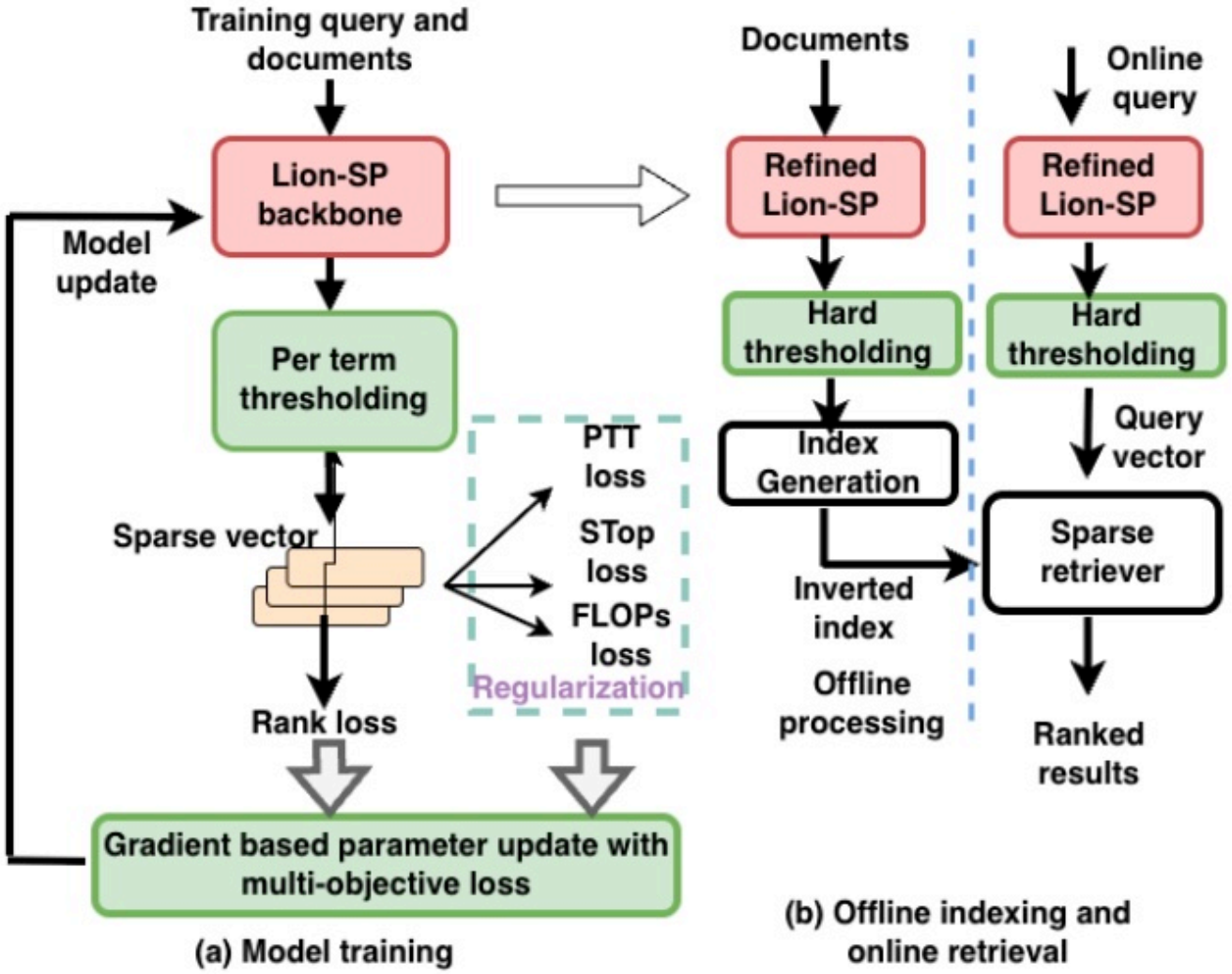
Multi-objective training loss in AdaSparse

$$\mathcal{L} = \left(\frac{1}{|B|}\sum_{q \in B} L_R\right) + \lambda_Q L_Q + \lambda_D L_D + \lambda_T L_T + L_{top}$$

Each loss term targets a different weight-distribution scenario:

Scenarios	Main techniques
Relevant and irrelevant weights	Rank loss
Large unimportant weights Noisy term weight rankings	FLOPs
Small unimportant weights Diverse term weight distribution	Per-term threshold.
Small weight with flat distribution Excessively long expansion	Soft top K

Offline/Online Processing Flow



Evaluation Setup: MS MARCO and BEIR

Training on MS MARCO passages
In-domain: MS MARCO Dev, DL19, DL20
Out-of-domain zero-shot: 13 BEIR datasets
Baselines

- –Lion-SP, HT, L1-FLOPs [SIGIR'22–'25]; Top-K: VDR [ICLR'24], mass-ratio pruning

Parameter setting for AdaSparse

- **–One tuned parameter:** expansion-limit factor b, swept on training
  - –Default b = 10× (queries), 8× (documents); actual BEIR range 1.3–13.5× / 1.4–6.1×
- **–All other parameters** follow prior work or a derived constraint

Comparison with Baselines

AdaSparse-1B vs. Lion-SP

- –4.2× less space
- –3.8–4× faster, with a 0.73% relevance difference

Best zero-shot relevance among sparsification baselines

- –2.3% better BEIR NDCG than MRP at similar sparsity
- –1.4–2.1× faster than HT, with 1% higher BEIR NDCG

	MS M. MRR	DL19 NDCG	DL20 NDCG	BEIR NDCG	Query len	Doc len
Lion-SP-1B [38]	**0.410**	0.747	**0.753**	0.535	293	1250
Lion-SP-1B-repro	0.408	0.753	0.744	**0.545**	208	1052
Lion-SP-1B-FLOPs'	0.382	0.732	0.717	0.508	100	**187**
Lion-L1-FLOPs	0.404	0.754	0.718	0.531	98	297
Top-305 1B	0.372	0.709	0.690	0.231	85	308
VDR (Top 256)	0.398	0.741	0.717	0.493	103	272
MRP(0.75) 1B	0.406	0.748	0.742	0.528	78	268
Lion-SP-1B-HT	0.404	0.754	0.744	0.536	189	391
AdaSparse-1B	0.408	**0.755**	0.729	0.541	**65**	280
Lion-SP-8B [38]	**0.418**	0.760	0.766	**0.552**	476	1457
AdaSparse-8B	0.417	**0.763**	0.752	0.543	**85**	297

Model	Latency (ms)		MRR@10 (Dev)			Storage
	$k = 10$	$k = 1000$	$k = 10$	$k = 1000$	Safe	(GB)
Lion-SP-1B	1.179 (4x)	9.048 (3.8x)	0.407	0.408	0.410	138 (4.2x)
Lion-SP-8B	1.341 (4.6x)	9.812 (4.1x)	0.415	0.416	0.418	154 (4.7x)
HT-1B	0.613 (2.1x)	3.223 (1.4x)	0.402	0.403	0.404	58 (1.8x)
AdaSparse-1B	0.292 (1x)	2.390 (1x)	0.401	0.406	0.408	33 (1x)
AdaSparse-8B	0.317 (1.1x)	2.863 (1.2x)	0.413	0.415	0.417	34 (1x)

Seismic retriever [SIGIR24]

Comparison at Two Sparsity Levels

AdaSparse-small: 6× soft limit for queries, 5× for documents
Two configurations of the others: matched to AdaSparse's average document length
AdaSparse-small: ~1.5× shorter than the others' defaults, yet higher relevance
AdaSparse-default: similar sparsity to the others' defaults, up to 2.3% higher BEIR NDCG

Method	Smaller Config.		Default Config.	
	Q/D Len.	MS M./BEIR	Q/D Len.	MS M./BEIR
AdaSparse	54 / 192	0.408 / 0.534	65 / 280	0.408 / 0.541
FLOPs	100 / 187	0.382 / 0.508	130 / 315	0.400 / 0.530
L1-FLOPs	104 / 221	0.397 / 0.512	99 / 296	0.404 / 0.531
MRP	60 / 200	0.398 / 0.526	78 / 268	0.406 / 0.529

Ablation: Method Combinations

Adding a soft top-K limit drastically reduces vector lengths with only a small relevance degradation

Per-term thresholding retains zero-shot relevance with more balanced performance than uniform thresholding

Performance with 1B	MS MARCO Dev			BEIR
	MRR@10	Q Len	D Len	NDCG@10
FLOPs	0.408	208	1052	0.545
FLOPs + STop	0.409	83	364	0.540
FLOPs + STop + PTT	0.408	65	280	0.541
FLOPs + STop + UTT	0.407	46	225	0.530

Impact of Hyperparameter Changes

Varying the soft expansion factor b gives more direct, explicit control of vector lengths

Increasing the per-term-thresholding weight also changes sparsity, but with a relatively larger zero-shot relevance drop

Different b for query					Different λ_T				
Setting	**MS M.**	**Q Len**	**D Len**	**BEIR**	**Setting**	**MS M.**	**Q Len**	**D Len**	**BEIR**
$b = 10$	0.408	65	280	0.541	$\lambda_T = 1$	0.408	65	280	0.541
$b = 8$	0.408	59	273	0.539	$\lambda_T = 3$	0.408	58	257	0.539
$b = 5$	0.407	48	262	0.536	$\lambda_T = 6$	0.408	54	245	0.534

Concluding Remarks

AdaSparse eases the tension between inference efficiency and the high-dimensional expansion of a massive LLM vocabulary

Multi-objective sparsification complementing FLOPs synergistically

– Learnable soft top-K regularization and per-term thresholding

AdaSparse vs. Lion-SP

– 1B: 4.2× less space, 3.8–4× faster, 0% Dev MRR & 0.73% BEIR NDCG gap

– 8B: 4.5× less space, 3.4–4.2× faster, 0.2% Dev MRR & 1.66% BEIR NDCG gap

Outperforms other sparsification baselines

– 2.3% better BEIR NDCG than MRP at similar sparsity; 1.4–2.1× faster than HT

Focused on LLaMA 3.2; future work targets other LLMs

This work has been supported by a SIGIR Student Travel Grant

Thank you!