HHR: HIERARCHICAL HASH RETRIEVAL FOR EFFICIENT LLM GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Efficient long-context inference is essential for large language models (LLMs), yet it poses a severe computational bottleneck. Hash-based retrieval offers an efficient alternative by encoding queries and keys into binary codes and using Hamming distance for key selection. However, this leads to a critical mismatch between Hamming distance and attention relevance. Query-Key logits depend jointly on directional similarity and feature magnitudes, whereas hash binarization discards magnitude information, causing both false-positive retrieval of low-logit keys and false-negative omission of high-logit keys. To address these failures, we propose **Hierarchical Hash Retrieval (HHR)**, a coarse-to-fine framework that progressively improves retrieval accuracy through **Geometry-Aware Key Routing (GKR)** and **Learned Hash Projection (LHP)**. GKR learns a head-wise orthogonal transformation to redistribute feature magnitudes and derive more discriminative page-level logit bounds, enabling effective pruning of low-logit keys while preserving important candidates. LHP then learns a head-wise projection space that aligns Hamming distance with the true Query-Key relevance ranking for fine-grained retrieval. By combining GKR and LHP, HHR suppresses false positives and recovers false negatives, substantially improving the fidelity of hash-based sparse attention. Extensive experiments across diverse LLMs and benchmarks demonstrate that HHR achieves superior performance over existing methods. For example, on LongBench, HHR improves the average score by 1.10 points and, at a context length of 128K, achieves up to a 3.30× decoding speedup and a 2.83× end-to-end speedup for Llama-3.1-8B-Instruct.

1 INTRODUCTION

The ability to process long contexts has become a critical requirement for large language models (LLMs) (Xu et al., 2025; Yi et al., 2025) in a wide range of real-world applications, including document understanding, retrieval-augmented generation, and multi-turn reasoning over extended dialogues (Wang et al., 2024; Sun et al., 2026b; Zhang et al., 2026). However, the complexity of the self-attention mechanism with respect to sequence length imposes a severe computational bottleneck, making efficient long-context inference a pressing challenge (Vaswani et al., 2017; Sun et al., 2026a; Wang et al., 2024). To mitigate this, sparse attention methods have been extensively explored (Wang et al., 2026; Zhu et al., 2026; Gao et al., 2026; Gong et al., 2025). These methods reduce the number of attended keys by selecting a small subset of important positions, thereby circumventing the need for full attention over the entire context (Child et al., 2019).

Among the various sparse attention strategies (Liu et al., 2026; Choi et al., 2026), hash-based retrieval has attracted considerable interest due to its adaptability and efficiency (Desai et al., 2025; Chen et al., 2025). By projecting queries and keys into a compact space and encoding them into binary codes, hash-based methods first adaptively extract the important keys based on Hamming distance, which can be executed using efficient XOR and POPCNT operations, and then perform attention computation on these selected keys. For instance, MagicPIG (Chen et al., 2025) employs locality-sensitive hashing to select a small subset of candidate keys from the full KV cache. HATA (Gong et al., 2025) learns binary hash representations for queries and keys to approximate Query-Key similarity via Hamming distance.

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.

Despite these gains, we uncover a critical mismatch in hash-based retrieval, where the magnitude information discarded for efficient hashing is essential for determining attention relevance. Specifically, Query-Key logits depend jointly on directional similarity and magnitudes, whereas hash binarization preserves only the signs of projected features and discards their magnitudes. As a result, Hamming distance deviates from the true Query-Key relevance, leading to inaccurate retrieval. As shown in Fig. 2, this mismatch gives rise to two complementary types of retrieval failure. First, directionally similar keys yield small Hamming distances to the query yet low logits due to their low magnitudes, becoming false positives that waste budget in Region III. Second, keys with large magnitudes yield high logits to the query but diverge directionally, producing large Hamming distances and becoming false negatives that miss important tokens in Region II.

To address both failure modes, as shown in Fig. 1, we propose **Hierarchical Hash Retrieval (HHR)**, a novel coarse-to-fine framework that tackles false positives and false negatives. HHR comprises two complementary modules, including **Geometry-Aware Key Routing (GKR)** for coarse-grained pruning, and **Learned Hash Projection (LHP)** for fine-grained hash retrieval. In the first stage, GKR partitions keys into non-overlapping pages and estimates the maximum logit of each page using per-dimension extrema. Pages with bounds below a threshold are discarded to eliminate low-relevance keys early. To obtain more discriminative bounds while preserving exact logits, GKR learns a head-wise orthogonal transformation that redistributes feature magnitudes across dimensions. The resulting extrema tighten the bounds for irrelevant pages, effectively pruning Region III false positives without sacrificing important keys. In the second stage, LHP performs fine-grained hashing over the surviving candidates. Unlike conventional random projections, LHP learns a head-wise projection that aligns Hamming distance with the true relevance ranking. Differentiable sign relaxation enables gradient-based training, while orthogonality, bit-wise decorrelation, and balance regularization improve the quality of the binary space. Consequently, LHP substantially reduces Region II false negatives by assigning a more favourable hash space.

Extensive experiments across various LLMs and benchmarks demonstrate the effectiveness of the proposed HHR. For example, on LongBench, HHR achieves average scores of 46.96, 44.46, and 45.66 on Llama-3.1-8B-Instruct, Mistral-7B-Instruct, and Qwen3-4B, respectively, outperforming SOTA by 1.10, 2.82, and 2.69. At the same time, HHR achieves up to a 3.30$\times$ decode speedup and a 2.83$\times$ end-to-end speedup at a 128K context length on Llama-3.1-8B-Instruct.

2 RELATED WORK

Structured sparse attention. Sparse attention mitigates the inference cost of long-context modeling by attending to only a subset of Query–Key pairs. Early approaches rely on structured sparsity patterns. For instance, LongGen combines local windows, attention sinks, and block-sparse attention for efficient long-context generation (Ge et al., 2025). DuoAttention assigns full or restricted KV cache depending on each head's retrieval capability (Xiao et al., 2025b). Star Attention employs distributed block-sparse attention to scale to extremely long sequences (Acharya et al., 2025). More recent methods move beyond fixed patterns by determining sparse regions in an input-dependent manner. HiP progressively prunes irrelevant blocks through hierarchical search (Lee et al., 2025).

Dynamic sparse attention. Dynamic sparse attention methods operate in an input-dependent manner (Jiang et al., 2024; Li et al., 2025b; Choi et al., 2026; Vasylenko et al., 2026; Desai et al., 2026; Zhu et al., 2026). FlexPrefill adaptively selects sparse patterns and budgets conditioned on queries and attention distributions (Lai et al., 2025). SeerAttention introduces a learnable gating network to predict salient blocks (Gao et al., 2025). SpargeAttention discards low-value computations via online block-level filtering (Zhang et al., 2025a). Twilight adopts a hierarchical Top-p mechanism to dynamically adjust the sparsity budget (Lin et al., 2025), suggesting that recent sparse attention is gradually shifting from predefined patterns toward query-dependent selection. NSA and MoBA reduce the attention scope through hierarchical token selection and block-level routing, respectively (Yuan et al., 2025; Lu et al., 2025). Another line of work leverages clustering or external memory to compress the search space, including Squeezed Attention, ClusterAttn, and Dynamic Block-Sparse Attention (Hooper et al., 2025; Zhang et al., 2025b; Xiao et al., 2025a).

Hash-based sparse attention. Hash-based sparse attention represents another efficient paradigm that leverages hashing to identify important keys relevant to each query. HyperAttention (Han et al., 2024) employs locality-sensitive hashing (LSH) to identify dominant entries and approxi-

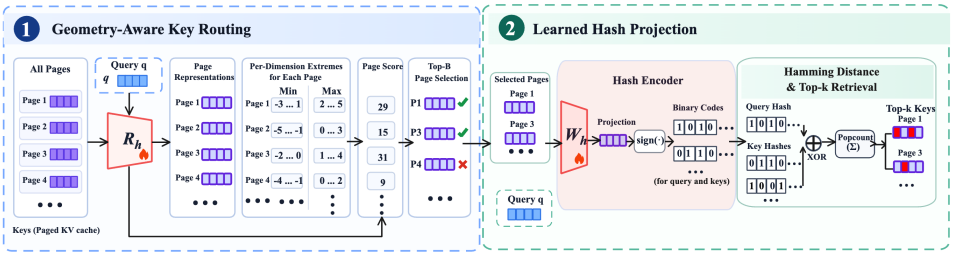


Figure 1: Overview of Hierarchical Hash Retrieva (HHR).

mates the remaining entries through sampling. MagicPIG (Chen et al., 2025) leverages LSH to enable sampling-based attention approximation, using importance sampling over key-value pairs to estimate the attention output. HAX (Zhan et al., 2025) combines LSH attention with Key Selection attention to capture content-dependent Query-Key interactions and shared attention to a small set of globally important keys, respectively. HashAttention (Desai et al., 2025) and HATA (Gong et al., 2025) both learn hash representations for queries and keys, using binary codes to approximate their similarity relationships for efficient pivotal-token selection. However, existing hash-based methods fail to consider the heterogeneous contributions of different feature dimensions to attention matching, leading to a misalignment between Hamming distance and the actual Query-Key relevance, making the retrieval accuracy unsatisfactory.

3 Method

3.1 Preliminary

Hash-based retrieval methods (Gong et al., 2025; Desai et al., 2025; Li et al., 2025a) project queries and keys into binary codes and use Hamming distance to estimate key importance. Given a hash function $f(\cdot)$, the binary representations are obtained as:

$$b_q = \text{sign}(f(q)), \quad b_k = \text{sign}(f(k)), \tag{1}$$

where $\text{sign}(\cdot)$ binarizes its input. The Hamming distance between two codes is then:

$$D_H(b_q, b_k) = \sum_{j=1}^{b} \mathbb{I}[b_{q,j} \neq b_{k,j}], \tag{2}$$

with b denoting the code length. This distance can be computed very efficiently via bitwise operations, offering substantial speedups over dense dot products.

3.2 Motivation

Hash-based sparse attention methods typically use random projections to map queries and keys into a lower-dimensional hashing space, convert the projected features into binary codes, and retrieve keys according to Hamming distance (Chen et al., 2025; Desai et al., 2025). However, Hamming distance can be poorly aligned with the actual Query-Key logits, leading to both false positive and false negative retrievals.

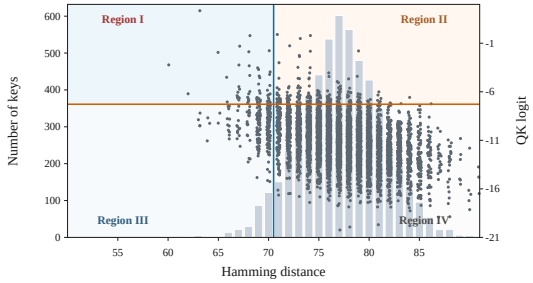


Figure 2: Distribution of Hamming distances obtained using a random orthogonal projection followed by $\text{sign}(\cdot)$, together with the corresponding Query-Key logits.

To characterize this misalignment, Fig. 2 compares the Hamming distance with the Query-Key logits. The orange dashed line denotes the Rop-5% boundary according to Query-Key logits, while the blue dashed line denotes the Top-5% boundary according to Hamming distance. Their intersection divides the key space into four regions:

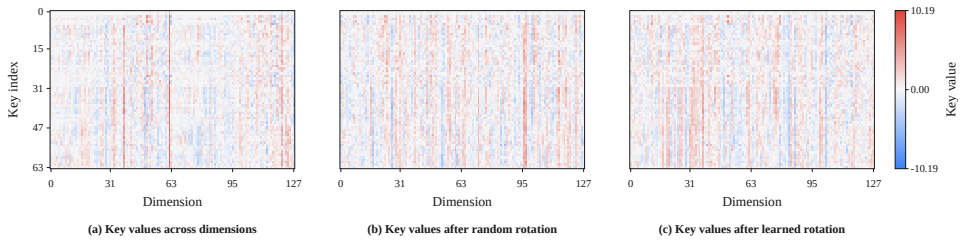


Figure 3: Distribution of keys before and after applying the learned transformation R_h. (a) Original key distribution across 128 dimensions for 64 consecutive keys. (b) The same keys after applying a random orthogonal rotation. (c) The same keys after applying the learned orthogonal matrix R_h.

- **Region I (upper-left):** Keys with small Hamming distances and high Query-Key logits, which are correctly identified as important by both hashing and original attention.
- **Region II (upper-right):** Keys with large Hamming distances but high Query-Key logits, which are important for attention but are missed by hash retrieval.
- **Region III (lower-left):** Keys with small Hamming distances but low Query-Key logits, which are incorrectly retained by hash retrieval.
- **Region IV (lower-right):** Keys with large Hamming distances and low Query-Key logits, which are correctly identified as unimportant by both mechanisms.

Regions II and III stem from the misalignment between Hamming similarity and Query-Key relevance. Specifically, the Query-Key logit between a query q and a key k is $q^\top k = \|q\|\|k\|\cos\theta$, where $\cos\theta$ captures directional similarity and $\|q\|, \|k\|$ determine the magnitude. Hamming distance, in contrast, depends only on binary signs and discards magnitudes, so it mainly reflects directional consistency in hash space but cannot preserve the magnitude information that determines key importance in attention (Shrivastava & Li, 2014).

This misalignment produces two retrieval failures. First, keys with similar directions but different magnitudes from the query yield small Hamming distances but low Query-Key logits, causing false positives in Region III, where low-logit keys consume the limited retrieval budget. Second, keys with large magnitudes but directions differing from the query yield high Query-Key logits but large Hamming distances, causing false negatives in Region II, where high-logit keys are discarded and attention misses important tokens.

3.3 Hierarchical Hash Attention

Motivated by the above analysis, we propose Hierarchical Hash Retrieval (HHR), which addresses the two types of retrieval failure via a coarse-to-fine strategy combining Geometry-Aware Key Routing (GKR) and Learned Hash Projection (LHP), as illustrated in Fig. 1.

3.3.1 Geometry-Aware Key Routing

To filter out low-logit keys early and save the retrieval budget for important keys, we introduce **Geometry-Aware Key Routing (GKR)** as a coarse-grained first stage. GKR partitions each head's keys into non-overlapping pages, estimates an upper bound on the maximum logit of each page (Tang et al., 2024), and discards pages whose bounds fall below a threshold. Thus, before fine-grained retrieval, pages dominated by low-logit keys (Region III) are discarded, while those potentially containing high-logit keys (Region II) are retained.

For the h-th head and page p, let q_h denote a query, and $l^j_{h,p}$ and $u^j_{h,p}$ denote the minimum and maximum of the keys in page p along dimension j. The maximum contribution of dimension j to the Query-Key logit is $\max(q^j_h l^j_{h,p},\ q^j_h u^j_{h,p})$, so the page-level estimated upper bound is:

$$U_{h,p} = \sum\nolimits_{j=1}^{d} \max\left(q^j_h l^j_{h,p},\ q^j_h u^j_{h,p}\right), \tag{3}$$

where d is the feature dimension. Since every key's value lies between the two extrema, $U_{h,p}$ bounds the largest logit attainable in the page. Given the threshold c_h, GKR retains a page p if $U_{h,p} > c_h$ and discards it otherwise, thereby removing pages unlikely to contain high-logit keys.

Importantly, bound quality depends on how keys distribute across dimensions. The same logits yield different dimension-wise extrema under different coordinate systems (Liu et al., 2024; Wang et al., 2026; Hooper et al., 2024). An orthogonal transformation thus offers a degree of freedom: it changes the extrema while preserving the logits exactly. We therefore learn a head-wise orthogonal matrix R_h applied to both query and key before computing the bound:

$$\tilde{q}_h = q_h R_h, \qquad \tilde{k}_{h,p,i} = k_{h,p,i} R_h, \tag{4}$$

where $k_{h,p,i}$ is the key at position i in page p of head h. Since $R_h R_h^\top = I$, the transformation preserves the exact Query-Key logits: $\tilde{q}_h(\tilde{k}_{h,p,i})^\top = q_h R(k_{h,p,i} R_h)_\top = q_h R_h R_h^\top (k_{h,p,i})^\top = q_h (k_{h,p,i})^\top$. Thus, a proper R_h can leave attention unchanged but redistributes feature values, allowing dimension-wise extrema to yield more discriminative page-level bounds. GKR then evaluates Eq. 3 on the transformed keys, learning R_h so that pages with important keys get large bounds, while irrelevant pages get tighter bounds.

To this end, GKR optimizes R_h with three complementary objectives: (1) $\mathcal{L}_{\text{recall}}$ keeps pages containing important keys, (2) $\mathcal{L}_{\text{precision}}$ suppresses unnecessary page selection, and (3) $\mathcal{L}_{\text{align}}$ aligns the retention score distribution with the relevance ranking of full attention. In particular, during training, the exact logits provide an oracle. For head h, the Top-K keys with the largest logits define important pages, *i.e.*, pages containing at least one such key. Let M_h be the important pages currently filtered out ($U_{h,p} \le c_h$). For each missed page $p \in M_h$, its bound should exceed the threshold by a margin m:

$$\mathcal{L}_{\text{recall}} = \frac{1}{|M_h|} \sum\nolimits_{p \in M_h} \max\left(0,\ c_h + m - U_{h,p}\right), \tag{5}$$

with $\mathcal{L}_{\text{recall}} = 0$ when M_h is empty. This objective encourages missed pages to exceed the threshold by a safety margin, thereby preventing high-logit keys from being discarded. High recall alone over-selects pages, so we add a precision objective to suppress unnecessary page selection. Let F_h be the selected pages containing no oracle Top-K key (false positives), and let $U^*_{h,p}$ be the oracle upper bound of page p, *i.e.*, its largest exact Query-Key logit. The gap $U_{h,p} - U^*_{h,p}$ measures the overestimation of the estimated upper bound, which is quantified by: $s_{h,p} = \min(\frac{1}{\sigma_h} \max(0, U_{h,p} - U^*_{h,p}),\ s_{\max})$, where σ_h is the standard deviation of $\{U^*_{h,p}\}$ over the pages of head h and $s_{\max}$ is the maximum allowed overestimation. The precision loss is then defined as:

$$\mathcal{L}_{\text{precision}} = \frac{\sum_{p \in F_h} (1 + s_{h,p}) \log(1 + s_{h,p})}{\sum_{p \in F_h} (1 + s_{h,p})}, \tag{6}$$

with $\mathcal{L}_{\text{precision}} = 0$ when F_h is empty. This objective penalizes false-positive pages with overestimated bounds, encouraging the learned transformation to tighten those bounds and reduce unnecessary selections. Then, to retain overall ranking, we add a distribution-level objective that couples the estimated upper bound with the hash-based score. Let $D_{h,i}$ be the pre-binarization distance (Li et al., 2021) between the query and key i under head h. Aggregating token-level distances with log-sum-exp gives the page-level Hash score:

$$H_{h,p} = \log \sum\nolimits_{i \in p} \exp\left(-D_{h,i} / \tau_h\right), \tag{7}$$

where τ_h is the temperature. The negative sign turns distance into similarity, so a larger $H_{h,p}$ indicates a more relevant page. After normalization of $\text{zscore}(x) = (x - \mu_x)/\sigma_x$, we combine the estimated upper bound and the page-level Hash score into a single retention score $C_{h,p} = \text{zscore}(U_{h,p}) + \text{zscore}(H_{h,p})$, and align it with the page relevance distribution of full attention:

$$P^*_{h,p} = \text{softmax}_p\left(\frac{\text{zscore}(U^*_{h,p})}{\tau_{\text{page}}}\right), \qquad \mathcal{L}_{\text{align}} = \text{KL}\left(P^*_h \,\|\, \text{softmax}_p(C_h)\right), \tag{8}$$

where τ_{page} is the temperature. Unlike recall and precision, which affect only the estimated upper bound, $\mathcal{L}$align encourages the combined retention score to both match the page relevance distribution induced by the exact Query-Key logits, thereby coupling the estimated upper bound with the Hash-based score to preserve the relevance ranking of full attention. Finally, the three objectives are combined as:

$$\mathcal{L}_{R_h} = \mathcal{L}_{\text{recall}} + \lambda_1 \mathcal{L}_{\text{precision}} + \lambda_2 \mathcal{L}_{\text{align}}, \tag{9}$$

where λ_1 and λ_2 control the contributions of the precision and distribution alignment objectives, respectively. To keep R_h orthogonal during optimization, we parameterize it with the Cayley transform (Helfrich et al., 2018): $R_h = (I + A)^{-1}(I - A)$, where A is skew-symmetric, $A^\top = -A$. Since $I + A$ is invertible, R_h is orthogonal by construction, so optimizing A adapts the feature distribution to Eq. 9 while preserving the exact Query-Key logits.

Fig. 3 shows that original keys are dominated by outlier dimensions (Joo et al., 2025; Hooper et al., 2024). Random rotation partially mitigates this, but the learned transformation yields a more favorable, less dimension-dominated distribution with tighter, more discriminative bounds. Thus, the page-level upper bound separates relevant from irrelevant pages effectively. For example, at page size 32, retaining 10% of candidates and treating Top-5% keys as important, GKR improves Top-5% recall from 43.20% to 47.48%. Overall, GKR learns an orthogonal transformation that preserves Query-Key logits while reorganizing the feature space, yielding discriminative bounds that prune Region III pages while achieving a better recall–budget trade-off and substantially improving the retention of high logit keys for Region II keys.

3.3.2 Learned Hash Projection

After the first stage, a much smaller candidate set remains for fine-grained retrieval. We therefore introduce **Learned Hash Projection (LHP)** as the second fine-grained stage, which learns a compact projection space aligning hash distance with Query-Key relevance, so binary codes better preserve directional similarity and support efficient retrieval.

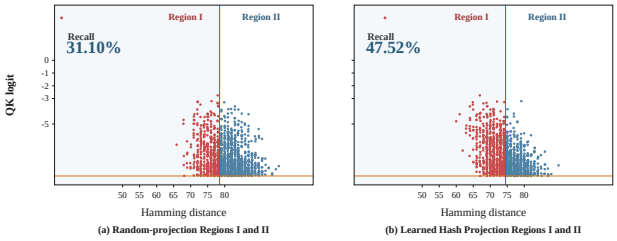


Figure 4: Top-10% key recall from layer 25, KV head 4 of Llama-3.1-8B-Instruct.

Specifically, for head h, we learn a projection matrix $\mathbf{W}_h \in \mathbb{R}^{d\times b}$, where b is the hash code length. $\mathbf{W}_h$ is trained offline on keys retained from GKR-selected pages. Since $\mathrm{sign}(\cdot)$ is non-differentiable, we use a differentiable relaxation during training. For input $\mathbf{x}$, its projected features and relaxed hash code are defined as:

$$\mathbf{z} = \mathbf{x}\mathbf{W}_h, \qquad \widehat{\mathbf{c}} = \tanh\left(0.1\,\mathbf{z}\right), \tag{10}$$

where the scaling factor 0.1 controls the saturation behavior of the $\tanh$ nonlinearity. During training, we compute the soft hash distance D_i between the query and the i-th candidate key $\mathbf{k}_{h,i}$ using these relaxed hash codes from Eq. 10. At inference time, the relaxed codes are replaced by their binary counterparts $\mathbf{c} = \mathrm{sign}(\mathbf{z})$, allowing for efficient Hamming-distance-based retrieval.

To train $\mathbf{W}_h$, we first define a soft selection probability $m_i = \sigma\left(\frac{D_{N+1}-D_i}{\tau_{\mathrm{sel}}}\right)$ for each candidate key $\mathbf{k}_{h,i}$, where $\sigma(\cdot)$ is the sigmoid function, D_{N+1} is the hash distance of the $(N+1)$-th closest candidate, and τ_{sel} is a temperature parameter controlling the sharpness of the soft selection. A smaller D_i thus yields a larger selection probability m_i. Let $\mathcal{K}_h^\star$ denote the set of Top-N relevant keys within the candidate set retained by GKR, determined according to their exact attention logits. We formulate the retrieval objective as:

$$\mathcal{L}_{\mathrm{retrieval}} = -\log\Big(\frac{1}{N}\sum\nolimits_{i\in\mathcal{K}_h^\star} m_i\Big). \tag{11}$$

Minimizing this objective encourages highly relevant keys to obtain smaller hash distances, thereby increasing their probability of being retrieved in the fine-grained hashing stage. In addition to the retrieval objective, we introduce three regularization terms. First, we enforce an orthogonality constraint on the projection matrix:

$$\mathcal{L}_{\mathrm{orth}} = \left\|\mathbf{W}_h^\top\mathbf{W}_h - \mathbf{I}\right\|_F^2, \tag{12}$$

which encourages different projection dimensions to capture complementary directions in the input space. Second, given a training batch of M relaxed hash codes $\widehat{\mathbf{c}}$, we stack them into $\widehat{\mathbf{C}} \in \mathbb{R}^{M\times b}$, center them along the sample dimension as $\boldsymbol{\mu} = \frac{1}{M}\sum_{i=1}^{M}\widehat{\mathbf{c}}_i$, $\widetilde{\mathbf{C}} = \widehat{\mathbf{C}} - \mathbf{1}\boldsymbol{\mu}^\top$, and compute the

bit-wise covariance matrix by $\mathbf{\Sigma} = \frac{1}{M}\widetilde{\mathbf{C}}^\top\widetilde{\mathbf{C}}$. We then define the decorrelation loss as:

$$\mathcal{L}_{\text{decor}} = \|\mathbf{\Sigma} - \text{Diag}(\mathbf{\Sigma})\|_F^2, \tag{13}$$

which suppresses correlations among different hash bits and thus reduces redundancy in the binary representations. Third, to prevent individual hash bits from collapsing to predominantly positive or negative values, we introduce a bit-balance loss:

$$\mathcal{L}_{\text{bal}} = \Big\|\frac{1}{M}\sum\nolimits_{i=1}^{M}\widehat{\mathbf{c}}_i\Big\|_2^2, \tag{14}$$

which encourages each bit to maintain balanced positive and negative activations across training samples, thereby avoiding degenerate codes and improving the effective utilization of the binary code space. The overall LHP training objective is obtained by combining the above terms:

$$\mathcal{L}_{\text{LHP}} = \mathcal{L}_{\text{retrieval}} + \lambda_3\mathcal{L}_{\text{orth}} + \lambda_4\mathcal{L}_{\text{decor}} + \lambda_5\mathcal{L}_{\text{bal}}, \tag{15}$$

where λ_3, λ_4, and λ_5 balance the objectives. Unlike HATA (Gong et al., 2025), LHP only identifies the Top-N relevant keys and ignores the rest, avoiding noisy low-relevance keys. As shown in Fig. 4, for Top-10% key selection, the learned projection raises key recall from 31.10% to 47.52%, reducing misalignment between Hamming-distance retrieval and Query-Key logit ranking. Overall, LHP learns a compact projection space where hash distance better aligns with Query-Key relevance, enabling more accurate binary retrieval that transfers candidates from Region II to Region I.

4 Experiments

4.1 Experimental Setup

Baselines and configurations. We compare HHR against seven baselines. Full Attention serves as the dense reference. Under the same token budget, Oracle selects the exact Top-K keys based on real Query-Key attention scores. The sparse retrieval baselines include HATA (Gong et al., 2025), MagicPIG (Chen et al., 2025), and Loki (Singhania et al., 2024). For KV cache compression, StreamingLLM (Xiao et al., 2024) retains attention sinks and recent tokens, whereas SnapKV (Li et al., 2024) selects informative cached positions from an observation window. Each baseline follows its recommended configuration under a comparable token budget. All methods use the Top-K ratio of 1.5%, the same prompt templates, and retain full attention in the first two layers. Additional experimental details and the hyperparameter settings are provided in Appendix A. All experiments are conducted on a single NVIDIA A800 GPU with PyTorch (Ansel et al., 2024).

Models, datasets, and metrics. HHR is evaluated on Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Mistral-7B-Instruct-v0.3 (Jiang et al., 2023), and Qwen3-4B (Yang et al., 2025) by using LongBench (Bai et al., 2024) and RULER (Hsieh et al., 2024) datasets. LongBench contains 16 English datasets covering question answering, summarization, few-shot learning, synthetic retrieval, and code completion, on which we report our results. Evaluation follows the official metric for each dataset, including F1, ROUGE-L, accuracy, and code similarity. RULER covers 11 tasks at a context length of 32K. These tasks examine retrieval, aggregation, and question answering. Both task-level and average accuracy are reported.

4.2 Main Results

Results on LongBench. As shown in Table 1, on LongBench, HHR demonstrates strong performance across all three model families. Compared with other sparse-attention and KV cache compression methods, HHR achieves average scores of 46.96, 44.46, and 45.66 on Llama-3.1-8B-Instruct, Mistral-7B-Instruct, and Qwen3-4B, respectively, obtaining the best results among these methods for each model. HHR also maintains consistent advantages on single-document QA, summarization, and few-shot tasks. These results indicate that HHR can effectively preserve task-relevant information and achieve strong cross-task generalization.

Results on RULER. As shown in Table 2, on RULER, HHR also consistently outperforms other methods for all three model families. HHR maintains accuracy close to Full Attention. In contrast, SnapKV, MagicPIG, and Loki suffer substantial performance degradation. For example, HHR achieves a mean performance of 76.23 and 82.71 on Mistral-7B-Instruct and Qwen3-4B, improving the SOTA by 6.43 and 4.02, respectively.

Table 1: Evaluation results on LongBench.

Model	Method	Single-Doc QA	Multi-Doc QA	Summarization	Few-shot	Synthetic	Code	Mean
Llama-3.1-8B-Instruct	Full Attention	40.56	44.37	29.18	69.28	54.19	50.80	47.51
	TopK (Oracle)	40.93	44.33	28.98	69.37	53.53	50.59	47.44
	MagicPIG	38.04	43.55	27.46	68.09	53.37	46.94	45.75
	SnapKV	31.14	42.49	20.23	58.94	**54.17**	48.30	41.46
	StreamingLLM	14.95	10.37	13.61	32.96	1.25	48.98	19.76
	Loki	31.84	34.19	27.26	62.83	42.62	**52.11**	41.11
	HATA	40.52	43.74	26.64	68.83	53.56	43.75	45.86
	HHR (Ours)	**40.63**	**44.29**	**28.39**	**69.32**	54.16	47.55	**46.96**
Mistral-7B-Instruct	Full Attention	36.76	38.14	28.90	70.75	51.00	52.85	45.71
	TopK (Oracle)	35.98	37.63	28.62	70.96	50.75	52.48	45.38
	MagicPIG	30.20	32.08	26.18	66.12	41.04	50.85	40.47
	SnapKV	27.22	32.00	20.07	58.83	**49.50**	50.42	38.39
	StreamingLLM	12.59	12.78	15.28	34.66	3.50	45.46	20.24
	Loki	30.08	28.02	25.50	64.54	30.06	**51.51**	37.97
	HATA	32.21	34.45	25.14	68.82	46.27	45.94	41.64
	HHR (Ours)	**34.85**	**37.00**	**28.06**	**70.39**	**49.50**	50.72	**44.46**
Qwen3-4B	Full Attention	40.04	43.43	27.42	68.46	48.62	51.16	46.10
	TopK (Oracle)	39.57	43.91	27.46	68.44	48.38	50.36	45.98
	MagicPIG	35.15	40.01	23.81	64.89	44.25	49.27	42.41
	SnapKV	30.84	40.17	17.70	58.56	**48.50**	**50.40**	39.98
	StreamingLLM	11.65	16.49	14.98	25.81	1.75	47.66	19.10
	Loki	35.49	40.34	26.65	63.92	47.75	46.14	42.94
	HATA	37.19	42.84	25.15	66.53	48.34	37.84	42.97
	HHR (Ours)	**39.97**	**43.66**	**28.66**	**68.59**	48.38	45.58	**45.66**

Table 2: Evaluation results on RULER with a context length of 32K. S1–S3 denote NIAH Single 1–3; MK1–MK2 denote NIAH Multi-key 1–2; MV and MQ denote NIAH Multi-value and NIAH Multi-query; CWE and FWE denote Common Words Extraction and Frequent Words Extraction; QA1 and QA2 denote question answering on SQuAD and HotpotQA, respectively.

Model	Method	S1	S2	S3	MK1	MK2	MV	MQ	CWE	FWE	QA1	QA2	Mean
Llama-3.1-8B-Instruct	Full Attention	93.60	100.00	99.80	99.60	86.80	99.30	100.00	37.88	86.00	80.40	61.40	85.89
	TopK (Oracle)	98.40	100.00	98.60	99.40	96.00	97.40	99.75	28.26	73.40	76.80	60.80	84.44
	StreamingLLM	0.80	1.20	2.00	2.00	1.20	2.50	1.75	0.72	49.07	22.20	25.40	9.89
	SnapKV	99.40	95.60	0.00	97.40	18.20	43.95	83.45	10.44	48.27	78.20	61.40	57.85
	MagicPIG	79.60	85.60	69.20	88.00	72.40	66.25	68.10	18.52	**84.60**	70.80	61.00	69.46
	Loki	93.80	98.00	**99.40**	95.00	71.20	97.90	98.15	**55.06**	58.60	72.40	61.20	81.88
	HATA	**100.00**	**100.00**	99.20	99.40	96.80	**98.60**	99.80	18.68	66.93	**79.00**	59.80	83.47
	HHR (Ours)	99.20	99.60	**99.40**	**99.60**	**97.40**	**98.60**	**99.90**	31.22	70.40	78.60	**61.60**	**85.05**
Mistral-7B-Instruct	Full Attention	100.00	99.80	96.40	92.00	84.60	96.40	89.70	73.18	82.73	60.40	40.60	83.26
	TopK (Oracle)	100.00	99.00	96.80	91.00	76.80	97.90	95.10	76.98	76.93	59.20	40.60	82.76
	StreamingLLM	1.40	1.60	1.80	1.80	2.20	2.30	1.90	1.82	**83.20**	19.40	22.20	12.69
	SnapKV	98.60	29.00	0.00	13.60	1.40	1.10	3.00	7.78	47.47	58.60	37.60	27.10
	MagicPIG	15.40	2.60	0.00	2.00	0.80	1.55	1.25	55.26	82.27	45.60	33.40	21.83
	Loki	**99.80**	83.00	81.00	58.80	40.20	**95.95**	**91.40**	**64.94**	72.27	44.00	36.40	69.80
	HATA	**99.80**	93.80	81.80	82.40	52.00	81.45	68.75	29.72	73.13	53.60	39.20	68.70
	HHR (Ours)	99.60	**95.80**	**92.20**	**86.60**	**64.80**	87.30	83.45	53.56	75.07	**60.00**	**40.20**	**76.23**
Qwen3-4B	Full Attention	100.00	100.00	100.00	97.00	97.40	89.65	96.75	64.82	81.40	58.40	46.00	84.67
	TopK (Oracle)	100.00	99.80	99.20	96.20	95.00	77.95	79.55	60.20	61.13	48.60	43.40	78.28
	StreamingLLM	0.80	1.75	0.25	2.50	2.25	1.65	0.56	0.90	1.58	18.75	20.25	4.66
	SnapKV	**100.00**	75.60	0.00	86.40	5.20	16.40	70.20	14.28	45.27	56.80	40.60	46.43
	MagicPIG	90.00	90.60	74.80	84.80	90.60	74.35	85.00	58.30	**77.07**	51.00	43.40	74.54
	Loki	99.80	93.40	96.60	93.20	73.00	14.70	13.80	52.64	68.07	**64.60**	**58.20**	66.18
	HATA	**100.00**	98.80	98.60	95.80	94.60	77.20	78.15	60.44	63.40	55.40	43.20	78.69
	HHR (Ours)	**100.00**	**99.60**	**100.00**	**97.60**	**96.20**	**94.25**	**90.40**	**63.44**	64.47	58.80	45.00	**82.71**

4.3 Ablation Study

GKR and LHP. To evaluate the roles of R and $\mathbf{W}$ respectively learned in GKR and LHP, we replace them with randomly initialized versions. As shown in Fig. 5, the datasets include 2WikiMQA (Ho et al., 2020), TREC (Li & Roth, 2002), and MultiFieldQA-en (Bai et al., 2024). HHR achieves the best performance across all datasets and backbones. After randomizing R, the performance drops by up to 2.52 and 2.23 points on Llama-3.1-8B-Instruct and Qwen3-4B, respectively. After randomizing $\mathbf{W}$, the performance drops by 1.50 and 2.50 points, respectively, indicating that both components play important roles in improving performance. In addition, jointly training R and $\mathbf{W}$ takes **only 18.93 and 28.45 minutes**, indicating that the proposed method incurs only minimal additional training overhead.

Generalization Performance across Different Model Scales. To evaluate the generalization ability of HHR across different model scales, we conduct experiments on Qwen3-4B, Qwen3-8B, and Qwen3-14B, as shown in Fig. 6. Under an extremely sparse Top-K budget of only 2%, our HHR

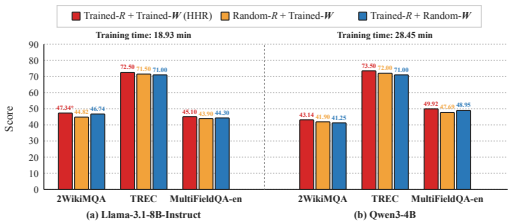


Figure 5: Comparison of the learned R and $\mathbf{W}$. The reported training time denotes the total time required to jointly train R and $\mathbf{W}$.

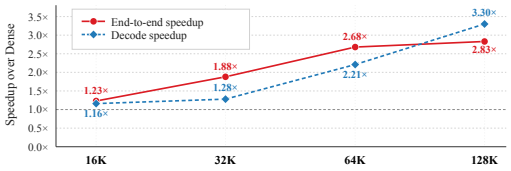


Figure 7: Speedup of HHR over full attention at different context lengths on Llama-3.1-8B-Instruct.

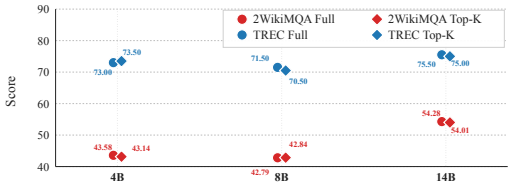


Figure 6: Comparison of Full and HHR Top-K performance across different Qwen3 model sizes.

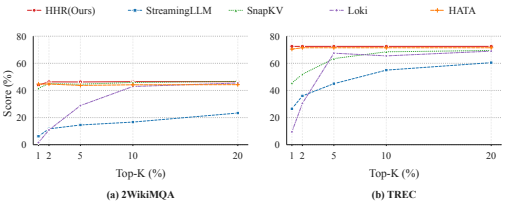


Figure 8: Comparison under different Top-K ratios on Llama-3.1-8B-Instruct.

consistently maintains performance close to Full Attention across all three model scales, validating the stable generalization ability of HHR.

Context Length. We evaluate inference efficiency at context lengths of 16K, 32K, 64K, and 128K. As shown in Fig. 7, decode speedup over dense attention increases from 1.16× at 16K to 3.30× at 128K, indicating a greater benefit during decoding as the context grows. End-to-end speedup reaches 2.68× at 64K and 2.83× at 128K. These results show that HHR improves efficiency across all evaluated context lengths, with particularly strong gains in long-context decoding.

Top-K Ratio. We conduct a Top-K ratio ablation on Llama-3.1-8B-Instruct and compare our HHR with other methods. As shown in Fig. 8, HHR stabilizes at around 46.5% from 2% onward on 2WikiMQA and consistently achieves 72.50% on TREC across all Top-K ratios, outperforming the other methods in most settings. More ablation studies are presented in Appendix C.

5 Limitations and Future Work

Despite achieving the best performance in certain cases, HHR still exhibits a performance gap relative to oracle Top-K selection. Therefore, developing more effective strategies remains an important focus for future work. For example, a stronger learning objective could potentially yield a better transformation R and projection $\mathbf{W}$. Moreover, HHR is orthogonal to other compression techniques such as KV quantization. The effect of combining HHR with them remains an interesting question but is beyond the scope of this paper, and we leave it for future work.

6 Conclusion

In this paper, we identify a fundamental mismatch in hash-based sparse attention, where binarization discards feature magnitudes essential for Query–Key logits, causing false positives from directionally similar but low-logit keys and false negatives from high-logit keys with divergent directions. To address this, we propose Hierarchical Hash Retrieval (HHR), a coarse-to-fine framework combining Geometry-Aware Key Routing (GKR) and Learned Hash Projection (LHP). GKR learns orthogonal transformations to redistribute magnitudes and derive tighter page-level logit bounds for early pruning of low-logit keys, while LHP learns a head-wise projection aligning Hamming distance with true relevance ranking to reduce false negatives. Together, they substantially improve the fidelity of hash-based sparse attention. Experiments across diverse LLMs and benchmarks confirm HHR's effectiveness and efficiency.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language polishing, grammatical revision, as well as partial code development and debugging. Generative AI tools were not used to replace experimental execution, generate experimental data, or serve as a source of unverified research conclusions or literature evidence. All AI-assisted text, code, and research suggestions were manually reviewed and verified by the authors. For textual content, we checked technical accuracy, consistency of presentation, and correspondence with the relevant literature. AI-assisted code was verified through actual execution and experimental testing. Research ideas and technical analyses suggested with the assistance of generative AI were independently assessed based on theoretical derivations, experimental results, and relevant prior work. The authors take full responsibility for the final content of this work, including all text, code, technical claims, and other research materials produced or revised with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

We support the reproducibility of our results by providing complete descriptions of the proposed method, implementation details, and experimental settings. The main algorithms and training/inference procedures are presented in the main text, while the models, datasets, hyperparameters, evaluation metrics, and detailed experimental configurations are described in the experimental section and appendix. Additional ablation studies and supplementary results are also provided in the appendix. The source code and necessary experimental configurations have been included in the supplementary material to facilitate reproduction of the results reported in this paper.

REFERENCES

Shantanu Acharya, Fei Jia, and Boris Ginsburg. Star attention: Efficient LLM inference over long sequences. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM international conference on architectural support for programming languages and operating systems, volume 2*, 2024.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. Longbench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Zhuoming Chen, Ranajoy Sadhukhan, Zihao Ye, Yang Zhou, Jianyu Zhang, Niklas Nolte, Yuandong Tian, Matthijs Douze, Leon Bottou, Zhihao Jia, and Beidi Chen. MagicPIG: LSH sampling for efficient LLM generation. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2025.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers, 2019.

Seonghwan Choi, Beomseok Kang, Dongwon Jo, and Jae-Joon Kim. Retrospective sparse attention for efficient long-context generation. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2026.

Aditya Desai, Shuo Yang, Alejandro Cuadron, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. HashAttention: Semantic sparsity for faster inference. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025.

Aditya Desai, Kumar Agrawal, Shuo Yang, Alejandro Cuadron Lafuente, Luis Gaspar Schroeder, Matei Zaharia, Joseph E Gonzalez, and Ion Stoica. vattention: Verified sparse attention via sampling. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2026.

Yizhao Gao, Zhichen Zeng, Dayou Du, Shijie Cao, Peiyuan Zhou, Jiaxing Qi, Junjie Lai, Hayden Kwok-Hay So, Ting Cao, Fan Yang, and Mao Yang. Seerattention: Learning intrinsic sparse attention in your llms, 2025.

Yizhao Gao, Shuming Guo, Shijie Cao, Yuqing Xia, Yu Cheng, Lei Wang, Lingxiao Ma, Yutao Sun, Tianzhu Ye, Li Dong, Hayden So, Yu Hua, Ting Cao, Fan Yang, and Mao Yang. Sparse attention adaptation for long reasoning. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2026.

Suyu Ge, Xihui Lin, Yunan Zhang, Jiawei Han, and Hao Peng. A little goes a long way: Efficient long context training and inference with partial contexts. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2025.

Ping Gong, Jiawei Yi, Shengnan Wang, Juncheng Zhang, Zewen Jin, Ouxiang Zhou, Ruibo Liu, Guanbin Xu, Youhui Bai, Bowen Ye, Kun Yuan, Tong Yang, Gong Zhang, Renhai Chen, Feng Wu, and Cheng Li. HATA: Trainable and hardware-efficient hash-aware top-k attention for scalable large model inference. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Insu Han, Rajesh Jayaram, Amin Karbasi, Vahab Mirrokni, David Woodruff, and Amir Zandieh. Hyperattention: Long-context attention in near-linear time. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2024.

Kyle Helfrich, Devin Willmott, and Qiang Ye. Orthogonal recurrent neural networks with scaled Cayley transform. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, 2020. doi: 10.18653/v1/2020.coling-main.580.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length llm inference with kv cache quantization. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Coleman Richard Charles Hooper, Sehoon Kim, Hiva Mohammadzadeh, Monishwaran Maheswaran, Sebastian Zhao, June Paik, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. Squeezed attention: Accelerating long context length LLM inference. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *CoRR*, 2023. doi: 10.48550/ARXIV.2310.06825.

Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Minference 1.0: Accelerating pre-filling for long-context llms via dynamic sparse attention. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Donghyeon Joo, Helya Hosseini, Ramyad Hadidi, and Bahar Asgari. Mustafar: Promoting unstructured sparsity for kv cache pruning in llm inference. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Xunhao Lai, Jianqiao Lu, Yao Luo, Yiyuan Ma, and Xun Zhou. Flexprefill: A context-aware sparse attention mechanism for efficient long-sequence inference. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2025.

Heejun Lee, Geon Park, Youngwan Lee, Jaduk Suh, Jina Kim, Wonyong Jeong, Bumsik Kim, Hyemin Lee, Myeongjae Jeon, and Sung Ju Hwang. A training-free sub-quadratic cost transformer model serving framework with hierarchically pruned attention. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2025.

Jinxing Li, Bob Zhang, Guangming Lu, Yong Xu, Feng Wu, and David Zhang. Harmonization shared autoencoder gaussian process latent variable model with relaxed hamming distance. *IEEE Transactions on Neural Networks and Learning Systems*, 2021. doi: 10.1109/TNNLS.2020.3026876.

Wenhao Li, Yuxin Zhang, Gen Luo, Haiyuan Wan, Ziyang Gong, Fei Chao, and Rongrong Ji. Spotlight attention: Towards efficient llm generation via non-linear hashing-based kv cache retrieval. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2025a.

Xin Li and Dan Roth. Learning Question Classifiers. In *Proc. of the International Conference on Computational Linguistics (COLING)*, 2002.

Yucheng Li, Huiqiang Jiang, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Amir H. Abdi, Dongsheng Li, Jianfeng Gao, Yuqing Yang, and Lili Qiu. MMInference: Accelerating prefilling for long-context visual language models via modality-aware permutation sparse attention. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025b.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Chaofan Lin, Jiaming Tang, Shuo Yang, Hanshuo Wang, Tian Tang, Boyu Tian, Ion Stoica, Song Han, and Mingyu Gao. Twilight: Adaptive attention sparsity with hierarchical top-p pruning. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Zhigeng Liu, Zhiyuan Ning, Ruixiao Li, Xiaoran Liu, Yuerong Song, Min Zhang, Ziwei He, and Xipeng Qiu. Faster than flash: Exploiting attention sparsity for efficient long-context decoding. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen (Henry) Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: a tuning-free asymmetric 2bit quantization for kv cache. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024.

Enzhe Lu, Zhejun Jiang, Jingyuan Liu, Yulun Du, Tao Jiang, Chao Hong, Shaowei Liu, Weiran He, Enming Yuan, Yuzhi Wang, Zhiqi Huang, Huan Yuan, Suting Xu, Xinran Xu, Guokun Lai, Yanru Chen, Huabin Zheng, Junjie Yan, Jianlin Su, Yuxin Wu, Yutao Zhang, Zhilin Yang, Xinyu Zhou, Mingxing Zhang, and Jiezhong Qiu. Moba: Mixture of block attention for long-context llms. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Anshumali Shrivastava and Ping Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

Prajwal Singhania, Siddharth Singh, Shwai He, Soheil Feizi, and Abhinav Bhatele. Loki: Low-rank keys for efficient sparse attention. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yutao Sun, Zhenyu Li, Yike Zhang, Tengyu Pan, Bowen Dong, Yuyi Guo, and Jianyong Wang. Efficient attention mechanisms for large language models. *Patterns*, 2026a. doi: https://doi.org/10.1016/j.patter.2026.101594.

Zechen Sun, Yuyang Sun, Zhaochen Su, Zecheng Tang, Juntao Li, Ao Zhou, Wenliang Chen, and Min Zhang. Data foundations of long-context language models: A survey. *Transactions of the Association for Computational Linguistics*, 14:1803–1825, 2026b. doi: 10.1162/tacl.a.775.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. QUEST: Query-aware sparsity for efficient long-context LLM inference. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Pavlo Vasylenko, Hugo Pitorro, Andre Martins, and Marcos V Treviso. Long-context generalization with sparse attention. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2026.

Xindi Wang, Mahsa Salmani, Parsa Omidi, Xiangyu Ren, Mehdi Rezagholizadeh, and Armaghan Eshaghi. Beyond the limits: A survey of techniques to extend the context length in large language models. In *Proceedings of the International Joint Conference on Artificial Intelligence(IJCAI)*, 2024.

Yifei Wang, Yueqi Wang, Zhenrui Yue, Huimin Zeng, Yong Wang, Ismini Lourentzou, Zhengzhong Tu, Xiangxiang Chu, and Julian McAuley. FASA: FREQUENCY-AWARE SPARSE ATTENTION. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2026.

Emily Xiao, Chin-Jou Li, Yilin Zhang, Graham Neubig, and Amanda Bertsch. Efficient many-shot in-context learning with dynamic block-sparse attention. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025a.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2024.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, junxian guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. Duoattention: Efficient long-context LLM inference with retrieval and streaming heads. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2025b.

Mengwei Xu, Dongqi Cai, Wangsong Yin, Shangguang Wang, Xin Jin, and Xuanzhe Liu. Resource-efficient algorithms and systems of foundation models: A survey. *ACM Comput. Surv.*, 2025. doi: 10.1145/3706418.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zihao Yi, Jiarui Ouyang, Zhe Xu, Yuwen Liu, Tianhao Liao, Haohao Luo, and Ying Shen. A survey on recent advances in llm-based multi-turn dialogue systems. *ACM Comput. Surv.*, 2025. doi: 10.1145/3771090.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

Zhihao Zhan, Jianan Zhao, Zhaocheng Zhu, and Jian Tang. Overcoming long context limitations of state space models via context dependent sparse attention. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Duzhen Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhijiang Guo, Le Song, and Cheng-Lin Liu. From system 1 to system 2: A survey of reasoning large language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026. doi: 10.1109/TPAMI.2025.3637037.

Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. SpargeAttention: Accurate and training-free sparse attention accelerating any model inference. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025a.

Minwei Zhang, Haifeng Sun, Jingyu Wang, Shaolong Li, Wanyi Ning, Qi Qi, Zirui Zhuang, and Jianxin Liao. ClusterAttn: KV cache compression under intrinsic attention clustering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025b.

Kan Zhu, Tian Tang, Qinyu Xu, Zhan Jin, Yile Gu, Zhichen Zeng, Rohan Kadekodi, Liangyu Zhao, Ang Li, Arvind Krishnamurthy, and Baris Kasikci. Tactic: Adaptive sparse attention with clustering and distribution fitting for long-context llms. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2026.

A CONFIGURATIONS DETAILS

Table 3 summarizes the configurations of the models used in our experiments. For inputs exceeding the maximum context length, we truncate from both the beginning and the end. Table 4 summarizes the configurations of the baseline methods used in our experiments. Table 5 summarizes the hyper-parameters used in our training objective. For Geometry-Aware Key Routing, we set the candidate budget to 30%, the final Top-K ratio to 1.5%, and the hash length to 128 bits.

Table 3: Configurations of the models used for evaluation.

Model	Configs	Values
Llama-3.1-8B-Instruct	#Layer	32
	#Attention Heads	32
	#KV Heads	8
	Hidden Size	4096
	Max Context Length	131072
Mistral-7B-Instruct	#Layer	32
	#Attention Heads	32
	#KV Heads	8
	Hidden Size	4096
	Max Context Length	32768
Qwen3-4B	#Layer	36
	#Attention Heads	32
	#KV Heads	8
	Hidden Size	2560
	Max Context Length	40960

Table 4: Configurations of the evaluated baseline methods.

Method	Settings
StreamingLLM	Streaming KV cache compression with 4 attention sink tokens and a retained cache ratio of 1.5%.
MagicPIG	Hash-based sparse attention with $K = 8$, $L = 40$, 4 sink tokens, and a local window size of 64, resulting in an effective candidate ratio of approximately 1.5%, comparable to TopK-1.5%.
Loki	PCA-based Top-k attention with 32 selected channels and a Top-k ratio of 1.5%.
HATA	Hash-based Top-k attention with trained hash weights and a hash length of 128 bits.
HHR (Ours)	Hierarchical hash-based retrieval with geometry-aware key routing and learned hash projection, using 128-bit hash codes.

Table 5: Loss coefficients used in our method.

Coefficient	Corresponding loss	Value
λ_1	$\mathcal{L}_{\text{precision}}$	0.05
λ_2	$\mathcal{L}_{\text{align}}$	0.25
λ_3	$\mathcal{L}_{\text{orth}}$	0.02
λ_4	$\mathcal{L}_{\text{decor}}$	0.02
λ_5	$\mathcal{L}_{\text{bal}}$	0.02

B LongBench-v2 Results

We further evaluate HHR on LongBench-v2 in Table 6. HHR achieves competitive performance across all three models. On Llama-3.1-8B-Instruct, HHR obtains the highest Overall score of 30.39, outperforming Full Attention, Oracle, and HATA. On Mistral-7B-Instruct-v0.3, HHR achieves an Overall score of 29.07, substantially outperforming HATA while remaining competitive with Full Attention and Oracle. For Qwen3-4B, HHR achieves the highest Hard Avg. score of 15.96 and improves the Overall score over HATA from 14.20 to 14.89, demonstrating its effectiveness on challenging long-context tasks

Table 6: Evaluation results on LongBench-v2.

Model	Method	Easy Avg.	Hard Avg.	Overall
Llama-3.1-8B-Instruct	Full Attention	31.74	28.67	29.84
	TopK (Oracle)	29.47	28.89	29.11
	HATA	31.90	29.17	30.21
	HHR (Ours)	**32.25**	**29.24**	**30.39**
Mistral-7B-Instruct-v0.3	Full Attention	30.36	28.97	29.50
	TopK (Oracle)	32.61	27.42	29.40
	HATA	28.14	**27.69**	27.86
	HHR (Ours)	**32.24**	27.12	**29.07**
Qwen3-4B	Full Attention	14.51	15.52	15.13
	TopK (Oracle)	14.51	15.52	15.13
	HATA	**13.36**	14.72	14.20
	HHR (Ours)	13.15	**15.96**	**14.89**

C More Ablation Study

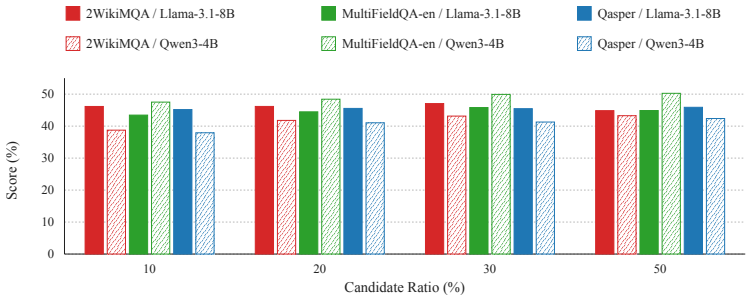


Figure 9: Comparison of model performance under different candidate ratios.

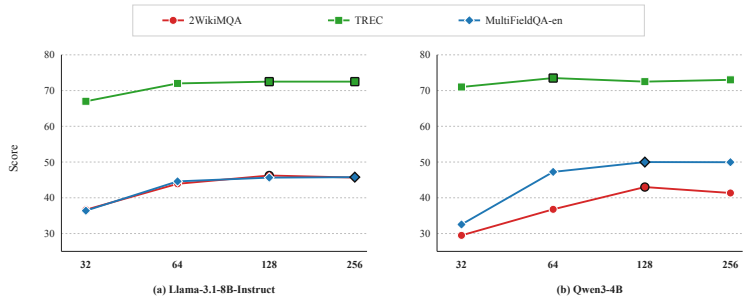


Figure 10: Sensitivity analysis of the number of hash bits.

Table 7: Ablation study of different loss terms in HHR.

Setting	GovReport	HotpotQA	LCC	Avg.
HHR (Ours)	**32.92**	**51.66**	**48.49**	**44.36**
w/o $\mathcal{L}_{\text{precision}}$	32.06	50.51	47.48	43.35
w/o $\mathcal{L}_{\text{align}}$	31.71	49.87	46.91	42.83
w/o $\mathcal{L}_{\text{orth}}$	32.23	50.82	47.71	43.59
w/o $\mathcal{L}_{\text{decor}}$	32.26	50.78	47.67	43.57
w/o $\mathcal{L}_{\text{bal}}$	32.39	51.07	47.91	43.79

Table 8: Stage-wise ablation study of HHR.

Method	QMSum	RepoBench-P	SAMSum
QUEST	22.52	45.84	41.59
GKR	23.17	46.41	42.22
LHP	23.68	46.99	42.73
HHR	**24.34**	**47.62**	**43.38**

Budget on Geometry-Aware Key Routing. As shown in Fig. 9, different models exhibit varying sensitivity to the budget. For Llama-3.1-8B, budgets of 20% and 30% generally yield better overall performance, with the 30% setting achieving the best results on 2WikiMQA and MultiFieldQA-en. In comparison, Qwen3-4B performs more consistently and achieves stronger overall performance with budgets of 30% and 50%.

Hash Bit Length. We evaluate our HHR's sensitivity to the hash bit length using 32, 64, 128, and 256 bits in Fig. 10. Increasing the bit length from 32 to 128 significantly improves overall performance. However, further increasing it to 256 bits yields limited gains and even slightly degrades performance on some tasks. The 128-bit setting is the most stable across different models and datasets, and we therefore adopt it as the default setting.

Effect of Loss Terms. We conduct an ablation study on the loss terms in HHR, as shown in Table 7. The full HHR model achieves the best performance on all datasets, with an average score of 44.36. Removing any single loss term leads to a consistent performance drop. Specifically, removing $\mathcal{L}_{\text{align}}$ causes the largest degradation, reducing the average score to 42.83 (a drop of 1.53), which indicates that aligning the hash representations is crucial for effective retrieval. The absence of $\mathcal{L}_{\text{precision}}$ also results in a notable decline to 43.35. In contrast, removing $\mathcal{L}_{\text{orth}}$, $\mathcal{L}_{\text{decor}}$, or $\mathcal{L}_{\text{bal}}$ leads to smaller drops, with average scores of 43.59, 43.57, and 43.79, respectively. These results demonstrate that all loss terms contribute to the overall performance, while the alignment loss plays the most important role.

Stage-wise Ablation Study. We further investigate the individual contributions of Geometry-Aware Key Routing (GKR) and Learned Hash Projection (LHP), with QUEST included as a retrieval baseline. As shown in Table 8, using GKR alone improves over QUEST by 0.65, 0.57, and 0.63 points on QMSum, RepoBench-P, and SAMSum, respectively. LHP alone yields larger gains of 1.16, 1.15, and 1.14 points on the three datasets. When GKR and LHP are combined, HHR achieves 24.34, 47.62, and 43.38, outperforming QUEST by 1.82, 1.78, and 1.79 points, respectively. The full model consistently surpasses either individual module, indicating that GKR and LHP provide complementary benefits for improving retrieval quality.