KASH: 1-BIT KEY-VALUE CACHE QUANTIZATION VIA ASYMMETRIC HASHING

Yifan Zhang, Qinghao Hu, Zhihui Wei, Jian Cheng

Nanjing University of Science and Technology · Institute of Automation, Chinese Academy of Sciences

Presenter: Yifan Zhang

ICPR 2026 · Lyon, France

南京理工大学 NANJING UNIVERSITY OF SCIENCE & TECHNOLOGY

计算机科学与工程学院 School of Computer Science and Engineering

中国科学院自动化研究所 Institute of Automation, Chinese Academy of Sciences

ICPR 2026 International Conference on Pattern Recognition LYON

Outline

1. Background
2. Our Method
3. Experimental Results
4. Conclusion

Outline

1. Background

2. Our Method

3. Experimental Results

4. Conclusion

1. Background

KV Cache Bottleneck

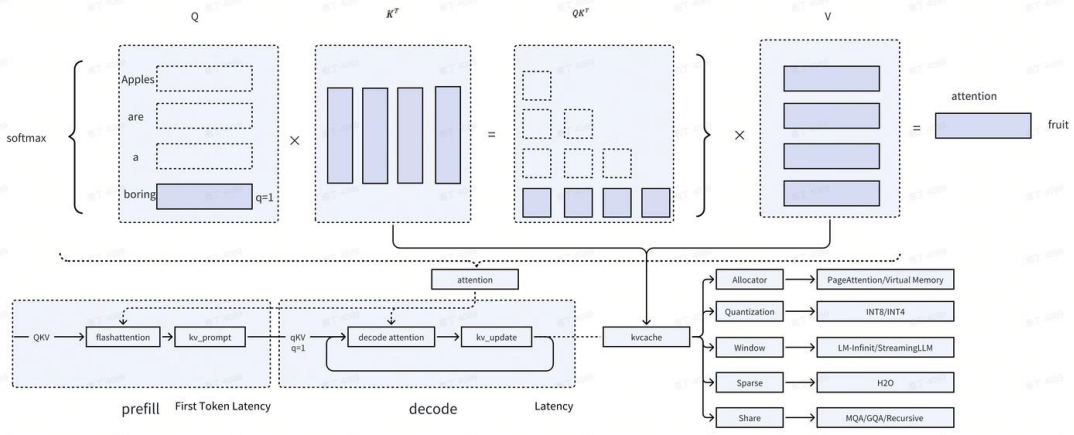
- Stores a key and value for every token
- Scales with batch size, context length and layers
- **Long-context decoding becomes memory-bound**

Quantization

- Less cache storage and memory bandwidth
- Grouped N-bit stores extra scales and zero-points
- KIVI-2 therefore averages about 3 bits per element

1-bit Barrier

- Keys are consumed through $q^T k$ retrieval scores
- Sign-only keys perturb $q^T k$ logits
- Softmax amplifies small score errors



hidden cost

quality collapse

We need to push average storage below the ≈3-bit floor while preserving the attention scores $q^T k$.

1. Motivation

From MIPS Retrieval to Attention Compression

MIPS(Maximum Inner Product Search) originates in database and information retrieval; attention shares the same inner-product retrieval structure.

1 MIPS in retrieval

Database vectors $x_1, \ldots, x_n$

Query vector y

$$i^* = \arg\max_i \; y^T x_i$$

Hard top-1 retrieval

2 Structural mapping

Retrieval query y

↕

Attention query q

Database vector x_i

↕

Cached key k_i

3 KASH insight

Attention uses soft retrieval

$$p_i = \text{softmax}_i(q^T k_i / \sqrt{d})$$

Transfer asymmetric MIPS hashing to binary Keys and continuous Queries, preserving attention scores under 1-bit storage.

KASH transfers asymmetric MIPS hashing from information retrieval to 1-bit KV-cache compression

1. Background

Why 1-bit Key quantization becomes an inner-product problem?

Attention consumes each key through $q^T k$; coordinate-wise reconstruction is only an indirect proxy.

Conventional objective: reconstruct K

$$\min||K - \widehat{K}||_F^2$$

At 1 bit, $\widehat{K}$ keeps mainly the sign pattern and loses magnitude.

A small reconstruction error does not guarantee $Q^T\widehat{K} \approx Q^T K$ for future queries.

Optimize what attention uses

MIPS-inspired objective: preserve scores

$$\min||B^T Z - K^T Q||_F^2$$

Binary Keys: $B = sign({H_k}^T K) \in \{-1, +1\}$

Continuous Queries: $Z = {H_q}^T Q \in \mathbb{R}$

Learn B and Z so that binary-key inner products directly approximate the original attention logits.

KASH directly optimizes attention scores under 1-bit Key storage; Hk and Hq are learned separately.

Outline

1. Background

2. Our Method

3. Experimental Results

4. Conclusion

2. Method


01
Asymmetric, data-dependent inner-product hashing
02
Outlier protection + per-head calibration
03
Hybrid 2-bit / 1-bit layer-adaptive cache

2. Method

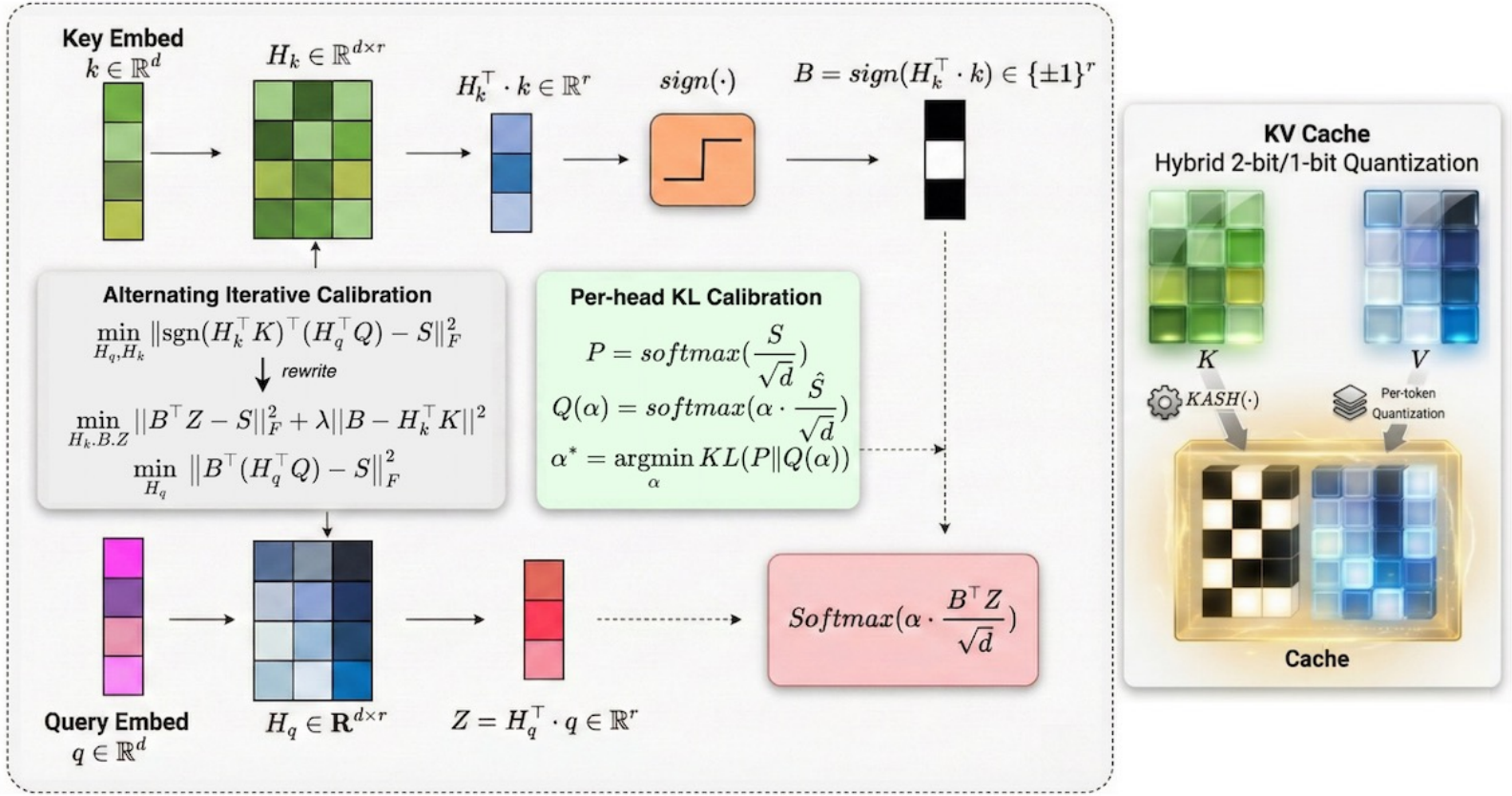
■ Proposed KASH pipeline

➢ **Offline calibration:**

- Learn Hk and Hq per layer-head
- Detect outlier channels

➢ **Online decoding:**

- Store bit-packed binary keys
- Keep queries continuous
- Reuse fixed maps at every step



■ Problem Formulation

Asymmetric inner-product fitting keeps Keys discrete and Queries continuous.

1 Define reference scores

Collect calibration Keys K and Queries Q for each layer-head, then form S.

2 Build asymmetric embeddings

Binary Key code B uses sign(·); projected Query Z remains real-valued.

3 Fit the inner products

Optimize Hk and Hq so $B^{\top}Z$ approximates the original score matrix S.

① Reference inner-product matrix

$$S \triangleq K^{\top}Q \quad (\text{or } S \triangleq \frac{1}{\sqrt{d}}K^{\top}Q).$$

② Asymmetric Key–Query mapping

$$B \triangleq \operatorname{sgn}(H_k^{\top}K) \in \{-1,+1\}^{r\times M}, \qquad Z \triangleq H_q^{\top}Q \in \mathbb{R}^{r\times N}.$$

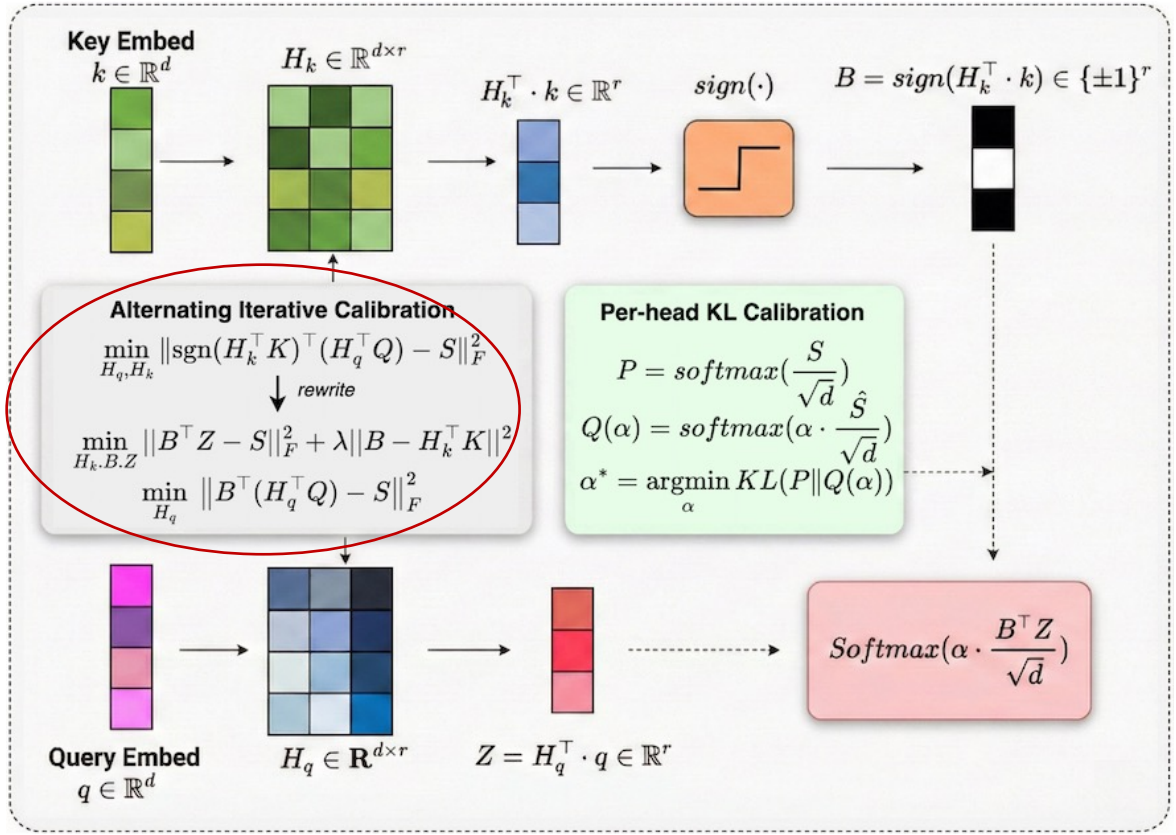
③ Inner-product fitting objective

$$\min_{H_k,H_q} \left\|\operatorname{sgn}(H_k^{\top}K)^{\top}(H_q^{\top}Q) - S\right\|_F^2 \equiv \min_{B,Z} \left\|B^{\top}Z - S\right\|_F^2.$$

1-bit discrete Keys + real-valued projected Queries → preserve $q^{\top}k$

2. Method

■ Alternating optimization



① **Fix Hq → Z = HqᵀQ is fixed**

$$\min_{H_k} \| \mathrm{sgn}(H_k^\top K)^\top Z - S \|_F^2. \Rightarrow \min_{H_k,B} \|B^\top Z - S\|_F^2 + \lambda\|B - H_k^\top K\|_F^2,$$

② **Alternate Hk and B**

B fixed → update Hk $H_k^\star = \arg\min_{H_k} \|B - H_k^\top K\|_F^2 \Rightarrow H_k^\star = (KK^\top)^{-1}KB^\top,$

Hk fixed → update B $b_t = \mathrm{sgn}(d_t - z_t \widetilde{Z}^\top \widetilde{B}),$

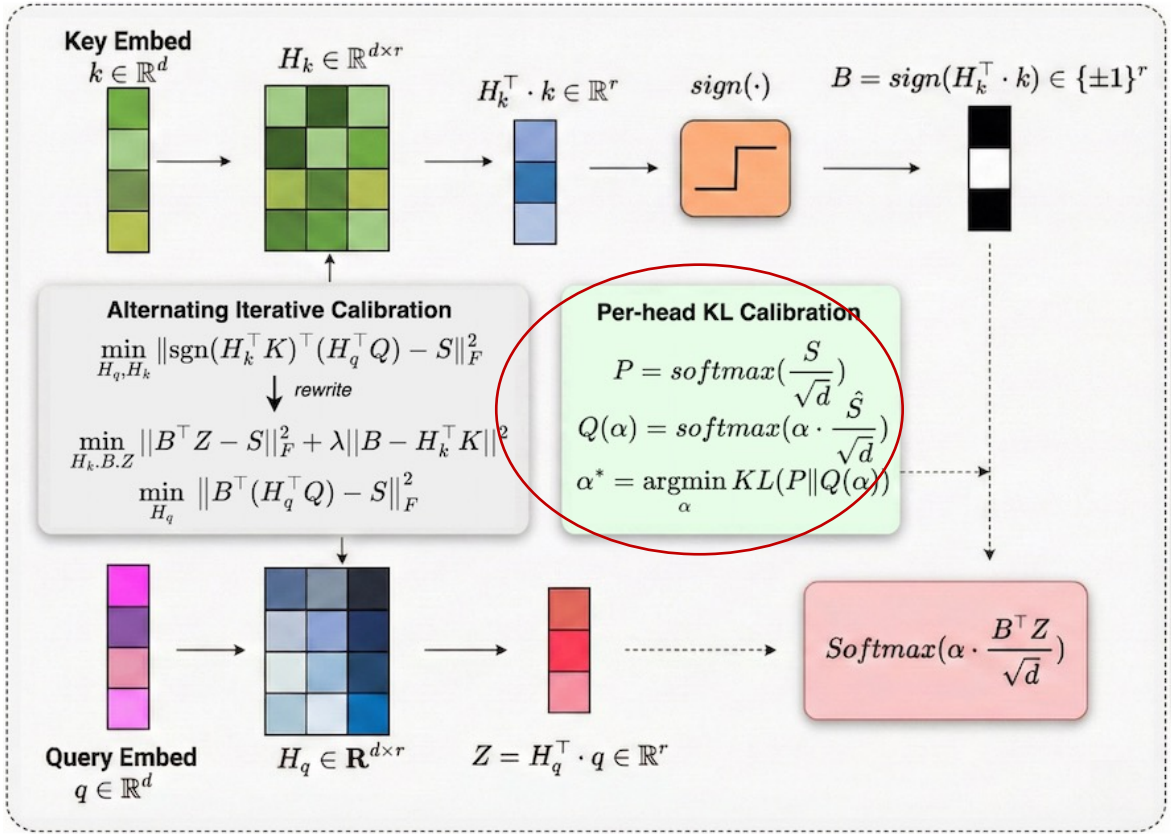
③ **Fix B → update Hq in closed form**

$$\min_{H_q} \|B^\top (H_q^\top Q) - S\|_F^2. \Rightarrow H_q^\star = (QQ^\top)^{-1} Q S^\top B^\top (BB^\top)^{-1},$$

④ **Repeat until converged**

2. Method

■ Per-head KL calibration



learn one α for each layer-head

① **Reference** **P = softmax(S / √d)** Full attention

② **Calibrated** **Q(α) = softmax(α·Ŝ / √d)** Scale the approximate logits

③ **Optimize** **α* = arg min KL(P ‖ Q(α))** One-dimensional offline search

2. Method

Algorithm 1. KASH Key Cache Quantization Framework

PHASE I: CALIBRATION (OFFLINE)

Require: Calibration data $\{(Q, K)\}$, bitwidth r, outlier ratio ρ.
Ensure: Transforms $\{H_q, H_k\}$, scalars $\{\alpha\}$, masks $\{\mathcal{O}\}$.

for layer $l \in \mathcal{L}_{\text{late}}$ **and** head h **do**
  Detect outliers $\mathcal{O}$ (top-ρ); Split $K \to [K_{\text{in}}, K_{\text{out}}]$; Init H_q, H_k via PCA.
  repeat
    Update $B \leftarrow \text{sgn}(H_k^\top K_{\text{in}})$ **and** H_k to align directions.
    Update $H_q \leftarrow \arg\min \|B^\top (H_q^\top Q) - S\|_F^2$ **and** calibrate α.
  until convergence
end for

PHASE II: HYBRID DECODING (ONLINE)

for each decoding step n **and** layer l **do**
  if $l \notin \mathcal{L}_{\text{late}}$ **then** ▷ *Early layers: 2-bit*
    KIVI.Write(k_n, v_n)
  else ▷ *Late layers: 1-bit asymmetric*
    Split $k_n \to [k_{\text{in}}, k_{\text{out}}]$ via $\mathcal{O}$.
    Quantize $b_n \leftarrow \text{sgn}(H_k^\top k_{\text{in}})$; Store bit-packed b_n and FP16 k_{out}.
    Attention Score: $\widehat{s} \leftarrow \text{softmax}\left(\frac{1}{\sqrt{d}} (\alpha \cdot \underbrace{(B_{\text{cache}}^\top H_q^\top q_n)}_{\text{1-bit Kernel}} + \underbrace{K_{\text{out}}^\top q_{\text{out}}}_{\text{FP16}}) \right)$.
  end if
end for

■ Offline calibration

- Detect outliers
- Initialize the two transforms
- Alternate updates of B, Hk, and Hq
- Calibrate alpha

■ Online decoding

- Early layers: 2-bit KIVI
- Late layers: 1-bit KASH
- Values: 2-bit per-token

Outline

1. Background
2. Our Method
3. Experimental Results
4. Conclusion

3. Experimental Results

■ Setup

➢ Models & Calibration

- LLaMA-2 7B and 13B
- WikiText-2: 128 × 2048 tokens

➢ Benchmarks

- LongBench: 7 long-context tasks
- LM-Eval: TruthfulQA, CoQA, GSM8K

■ LongBench Results on LLaMA-2-7B

➢ RepoB-P: RepoBench-P · MNews: Multi-News

Method	Avg.bit	TriviaQA	RepoB-P	Qasper	TREC	QMSum	LCC	MNews	Avg.
FP16	16	87.72	59.82	9.52	66	21.28	66.66	3.51	44.93
KIVI-2	3	87.42	60.23	9.31	66	20.5	66.88	1.14	44.50
KVQuant-3	3.33	**87.55**	**59.5**	**13.67**	**66**	16.53	61.52	1.65	43.77
QJL	3.08	87.47	58.52	9.56	**66**	20.83	64.27	1.3	43.99
KIVI-2 / KIVI-1	2.5	2.67	16.62	2.59	18	1.49	20.8	0.47	8.95
KASH (Ours)	**2.22**	87.51	57.76	9.61	**66**	**20.84**	**64.94**	**2.5**	**44.17**

3. Experimental Results

■ LongBench and LM-Eval Results on LLaMA-2-13B

➢ RepoB-P: RepoBench-P · MNews: Multi-News

Method	Avg.bit	TriviaQA	RepoB-P	Qasper	TREC	QMSum	LCC	MNews	Avg.
FP16	16	87.87	56.42	9.32	70	21.38	66.61	3.71	45.04
KIVI-2	3	87.31	53.66	8.27	69.5	20.69	65.08	3.19	43.96
QJL	3.08	87.38	**53.16**	9.18	69.5	20.05	**63.34**	**2.84**	43.64
KIVI-2/KIVI-1	2.5	0	13.46	0	19	0	18.61	0.01	7.30
KASH (Ours)	**2.22**	**87.47**	52.43	**9.83**	**70**	**20.69**	63.05	**2.84**	**43.76**

➢ Standard-context evaluation across TruthfulQA, CoQA, and GSM8K

Method	Avg.Bit	TruthfulQA	CoQA	GSM8K	Avg.
FP16	16	29.53	66.37	22.67	39.52
KIVI-2	3	29.84	66.23	**20.77**	38.95
KIVI-2/KIVI-1	2.5	29.43	65.57	16.3	37.1
KASH (Ours)	**2.2**	**31.43**	**66.57**	20.22	**39.41**

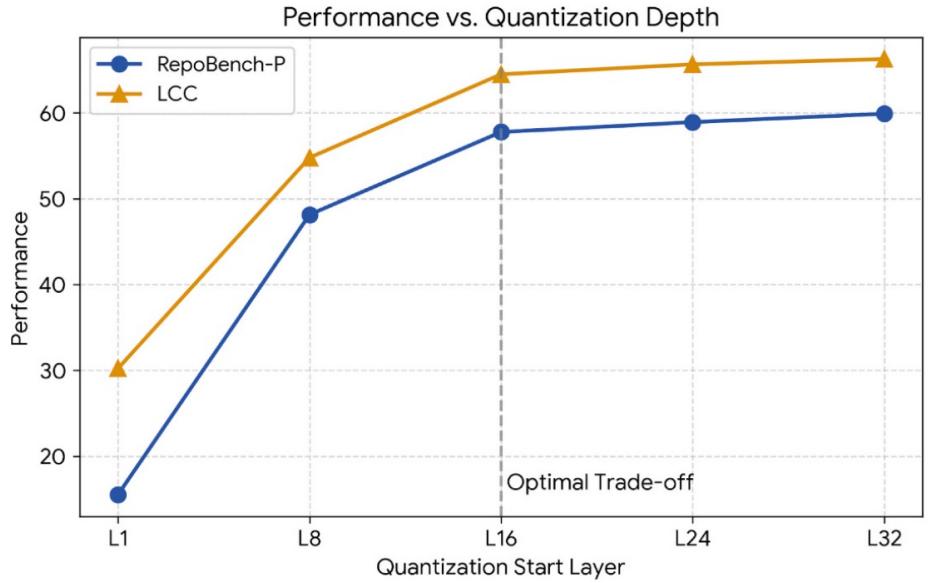
3. Experimental Results

■ Ablation Results

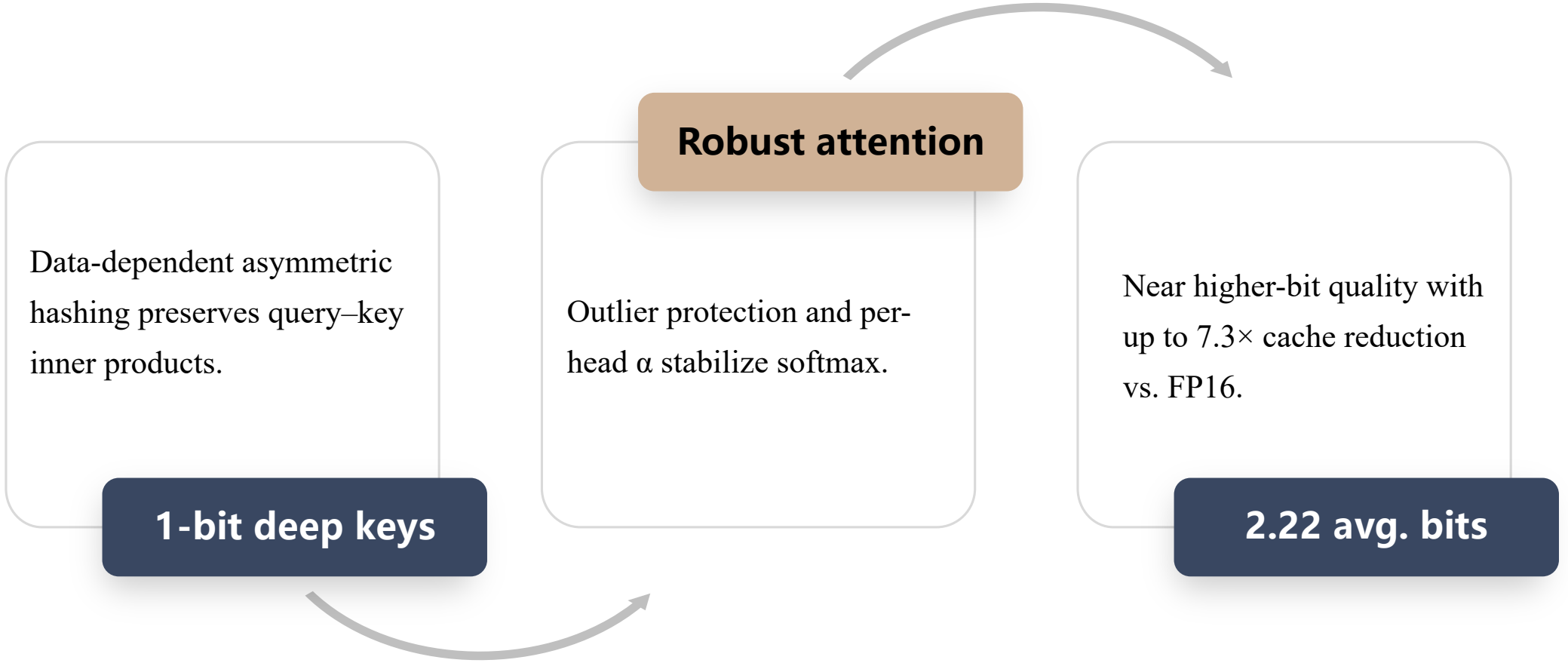
➢ Hq and α

Method	TriviaQA	RepoBench-P	TREC	Avg.
FP16 (no quant.)	87.72	59.82	66	71.18
Ours (Full): Hk ≠ Hq, w/ α	87.51	57.76	66	70.42
w/o Hq (set Hq = I), w/ α	84.06	50.59	64	66.22
w/o α (fix α = 1), w/ Hq	87.31	56.69	66	70
w/o Hq and w/o α	83.86	47.59	63	64.82

➢ Quantization Depth



4. Conclusion



ultra-long contexts / distribution shifts / value-cache learning / combinations with sparse attention...

THANKS!

Reporter : Yifan Zhang yifan_zhang@njust.edu.cn

南京理工大学 NANJING UNIVERSITY OF SCIENCE & TECHNOLOGY

CASIA 中国科学院自动化研究所 Institute of Automation, Chinese Academy of Sciences

ICPR 2026 International Conference on Pattern Recognition LYON