RICH: PRESERVING INTRA-CLASS SEMANTIC ORDER IN BINARY CODES LEARNED FROM COARSE LABELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep hashing compresses images into short binary codes for large-scale retrieval. Its dominant paradigm, hash-center methods, trains every sample of a class toward one binary center, so the code encodes the class label but not the ordering among same-class items: kayaks and sailboats both labeled `watercraft` collapse onto the same center. That ordering decides what a user sees, since with thousands of same-class items the top-100 of a correct-class query is fixed by it, and class-level mean average precision (mAP) does not measure it. We study this ordering in the coarse-label setting, where training uses only superclass labels and subclass labels are withheld for evaluation, formalize it as a retrieval-quality dimension of its own, and propose two metrics for it: Intra-class Spearman ρ and Semantic Recall@k. Keeping the ordering in a binary code is hard because the two objectives compete for a fixed bit budget and encoding both in one bit space causes interference that more bits do not remove. Our proposed Residual-Informed Center Hashing (RICH) resolves the conflict by splitting the N bits between a center head trained on the superclass label and a residual head trained on sub-clusters found by K-means in the pretrained embedding, with the number of sub-clusters set automatically per dataset from the per-class sample count and the codebook capacity. In two-stage retrieval, the center bits rank by superclass and the residual bits refine the order within it. On CIFAR-100, ImageNet-S20, and ImageNet-S50, RICH outperforms the second-best baseline on all intra-class metrics at every bit width, reaching $1.7{\sim}4.5\times$ its subclass mAP@100, scored against held-out subclass labels, and $1.2{\sim}2.2\times$ its intra-class ρ, scored against the pretrained partition. It also leads on all nine inter-class settings when comparing its center bits with the baselines' full codes at the same width, because the sub-cluster loss sharpens the shared backbone that the center head reads. Code is available at `https://anonymous.4open.science/r/RICH-286C`.

1 INTRODUCTION

Deep hashing (Wang et al., 2018b) learns binary codes for large-scale image retrieval that are compact to store and fast to compare by Hamming distance. It is supervised by class labels, which in practice are coarse, because fine-grained labels require expert annotators to distinguish subtle differences between classes (Touvron et al., 2021; Grčić et al., 2024). Learning finer-grained representations from coarse labels alone has been studied as coarse-to-fine representation learning (Wu et al., 2018; Touvron et al., 2021; Bukchin et al., 2021; Grčić et al., 2024), with significant progress in continuous embedding spaces, but it has not been carried into binary codes. Because codes are discrete, continuous-space objectives do not transfer directly, and binarizing a continuous embedding after the fact retains intra-class ordering but loses class separation (Appendix A).

Early deep hashing methods learned pairwise similarity via pairwise or triplet losses (Li et al., 2016; Cao et al., 2017), but $O(N^2)$ training cost and convergence instability limited scalability. Hash-center methods (Yuan et al., 2020; Hoe et al., 2021; Wang et al., 2023) overcome these issues by assigning a binary center per class and training samples to converge to it, establishing the dominant paradigm in supervised hashing. The recent codebook-centric variant (Yin et al., 2026) encodes inter-class structure more actively. However, these methods converge all samples of a class to a single target, losing intra-class semantic structure. Losing this structure matters because a retrieval system is judged by the top of its ranked list: in a database with thousands of items per class (e.g.,

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.

2,700 per class in our primary benchmarks), the hundred results shown for a correct-class query are chosen entirely by the intra-class order. Class-level mAP counts every same-class item as equally relevant, so it scores all these orderings alike, and the quality differences go unmeasured. When that order is arbitrary, a kayak query returns lifeboats and sailboats (Figure 2, Appendix B), and a code that encodes only the class label carries no more retrieval information than the label itself, regardless of code length.

Re-ranking the retrieved candidates with continuous features does not recover this order, because it only reorders the same hundred candidates, which were already drawn arbitrarily from 2,700 same-class items. The fraction of same-subclass items among them remains unchanged, and the kayaks the code omits stay omitted (Appendix C). Recovering them requires re-ranking the whole class, which means storing continuous features for the entire database (i.e., 2 KB for a 512-d float embedding against 8 B for a 64-bit code) and forfeits the compression that motivates hashing. The order must therefore be encoded in the code itself, and existing hash-center methods do not, reaching only $12{\sim}30\%$ subclass mAP@100 (Table 1).

Fine-grained hashing methods (Shen et al., 2022; Wei et al., 2021) address discrimination among fine-grained classes but require the corresponding class labels for training, and no existing method derives that supervision from coarse labels alone. We call the resulting problem *coarse-label hashing*: learn N-bit codes from superclass labels alone such that Hamming ranking both separates superclasses and orders same-class items by their fine-grained semantics, with subclass labels withheld for evaluation (Section 3.1).

This loss of intra-class semantics cannot be resolved by simply allocating more bits. Binary codes optimized for class discrimination converge all same-class samples to a single target, so additional bits are consumed in the same direction. Conversely, encoding intra-class structure in the same bit space causes Hamming distance to sum class-discrimination and intra-class signals indiscriminately, so the two objectives cannot coexist in a single bit space.

This paper proposes Residual-Informed Center Hashing (RICH) for coarse-to-fine representation learning in binary code space. RICH splits the N bits between a center head and a residual head, encoding coarse class discrimination and preserving intra-class semantic structure in independent bit subspaces. To the best of our knowledge, it is the first method to preserve intra-class ordering in binary codes learned from coarse labels alone. The contributions are summarized as follows:

1. **Problem formulation.** We formalize intra-class semantic ordering as an independent retrieval-quality dimension in coarse-label hashing and propose two metrics: Intra-class Spearman ρ, the rank agreement between intra-class code and pretrained-embedding distances, and Semantic Recall@k, the neighbor overlap between the model's continuous features and its codes.
2. **Dual-head binary-code design.** We propose a dual-head architecture in which a center head and a residual head learn coarse discrimination and fine-grained structure preservation in independent bit subspaces, together with a K_{opt} algorithm that sets the number of sub-clusters from the per-class sample count and codebook capacity, requiring no fine-grained annotation. On CIFAR-100, ImageNet-S20, ImageNet-S50, and three additional benchmarks, RICH improves intra-class Spearman ρ by $1.2{\sim}2.3\times$ over the second-best baseline at competitive inter-class mean average precision (mAP) under a matched inter-class bit budget, and on the three primary datasets it leads all nine inter-class settings. Splitting the code is therefore not a trade-off: weakening the sub-cluster loss lowers inter-class mAP as well.
3. **Hierarchical retrieval strategy.** We present a two-stage retrieval strategy: coarse search on the center bits, then intra-class refinement on the residual bits. Without fine-grained annotations, RICH achieves $1.7{\sim}4.5\times$ the second-best subclass mAP@100 on the three primary datasets at every bit width.

2 Related Work

Coarse-to-Fine Representation Learning. In continuous embeddings, early coarse-to-fine methods train a non-parametric classifier that generalizes to finer categories (SNCA, Wu et al., 2018) or combine coarse classification with self-supervision for sub-label retrieval (Grafit, Touvron

et al., 2021). Recent work turns to contrastive and hierarchical objectives for intra-class separation (Bukchin et al., 2021; Feng & Patras, 2023; An et al., 2022) and to discovering fine-grained classes from coarse labels (Grčić et al., 2024). Carrying this into binary code space adds constraints that continuous embeddings do not face: bit-budget competition and Hamming-distance interference.

Supervised Deep Hashing. Hash-center methods established the dominant paradigm in supervised hashing by assigning a binary center per class and training samples to converge to it. CSQ (Yuan et al., 2020) pre-designs centers, OrthoHash (Hoe et al., 2021) maximizes inter-center distance via orthogonal loss, MDS (Wang et al., 2023) enforces uniform inter-class Hamming distances, SHC (Chen et al., 2025) reflects a semantic hierarchy in the codebook, and CRH (Yin et al., 2026) reassigns class centers dynamically during training. A recent variant in the cross-modal setting, DSTH (Han et al., 2026), keeps the center paradigm but assigns several centers per class to absorb intra-class variance. All of these optimize Hamming structure at the class-label level. A single center per class collapses intra-class diversity; multiple centers only tighten clusters around each center, and intra-class structure cannot be preserved in the same bit space.

Fine-Grained and Hierarchical Hashing. Fine-grained hashing methods address discrimination among fine-grained classes through region localization (Zeng et al., 2019; Cui et al., 2020), attribute annotation (Wei et al., 2023), characteristics matching (Chen et al., 2024; 2026), semantic preservation-based code generation (Li et al., 2025), and query optimization (Wang et al., 2025), but require the corresponding fine-grained labels for training. Hierarchical hashing methods (Zhang et al., 2021; Wang et al., 2018a) use semantic hierarchies so that inter-class Hamming distances reflect hierarchical relationships. Hierarchical hashing addresses inter-class relationships, not intra-class ordering. Fine-grained hashing reaches the subclass level, but it moves the class boundary rather than ordering within a class because it targets one center per fine class; the ordering inside that class collapses in turn.

Unsupervised Binary Representation Learning. Unsupervised methods such as locality-sensitive hashing (LSH) (Indyk & Motwani, 1998), iterative quantization (ITQ) (Gong et al., 2013), product quantization (PQ) (Jégou et al., 2011), self-supervised contrastive quantization (Jang & Cho, 2021), and hierarchical hyperbolic quantization (Qiu et al., 2024) learn binary codes directly from the data distribution. PQ and its variants also split the vector space into independently quantized subspaces, but the split is unsupervised, over coordinate axes of one semantic level, and in a continuous space. Whether the structure present in a pretrained embedding survives binarization is a separate question from whether it exists. Sign and ITQ binarization of the same features retain intra-class ordering but lose class separation, and only the supervised dual-head design keeps both (Appendix A).

Research gap. No line of work learns an ordering within a class from coarse labels in a binary code. Coarse-to-fine learning preserves intra-class order in a continuous space; hash-center methods assign each class a single target; fine-grained and hierarchical hashing move the class boundary using the fine label and score it by binary relevance at that granularity; and unsupervised quantization splits coordinates without supervision. RICH splits a binary code into supervised inter-class and intra-class subspaces from coarse labels alone, never seeing the fine label during training and using it only as evaluation-time ground truth for graded intra-class relevance (Järvelin & Kekäläinen, 2002).

3 METHOD

3.1 PROBLEM FORMULATION

Given a training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$ with $\mathbf{x}_i \in \mathbb{R}^D$ and superclass label $y_i \in \{1, \dots, C\}$, supervised deep hashing learns $h : \mathbb{R}^D \to \mathbb{R}^N$ whose sign gives the N-bit code $\mathbf{b}_i = \operatorname{sign}(h(\mathbf{x}_i))$. Retrieval ranks the database $\mathcal{X}$ by Hamming distance to the query code. For length-n vectors, we write $\operatorname{dist}_H(\mathbf{u}, \mathbf{v}) = \frac{1}{2}(n - \mathbf{u}^\top \mathbf{v})$: the Hamming distance when both are binary, and its extension to the relaxed arguments below. In this setting, the coarse-to-fine objective is to learn codes that support both coarse classification and fine-grained intra-class retrieval, using only the coarse labels y_i. Hash-center methods (Yuan et al., 2020; Hoe et al., 2021; Wang et al., 2023) assign each superclass c a binary center $\mathbf{c}_c \in \{-1, +1\}^N$ and minimize

$$\mathcal{L}_{\text{center}} = \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}} \ell\big(h(\mathbf{x}_i),\ \mathbf{c}_{y_i}\big), \tag{1}$$

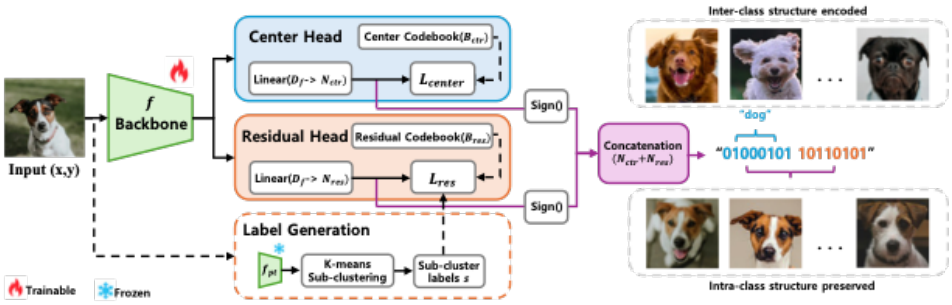


Figure 1: RICH architecture. The center head (N_{ctr} bits) encodes inter-class structure, and the residual head (N_{res} bits) encodes intra-class structure; center bits alone rank superclasses, and the residual bits refine the ranking within one.

where ℓ is a cosine-similarity or cross-entropy loss. The codebook-centric variant (Yin et al., 2026) instead learns the center assignment via alternating optimization in the style of expectation–maximization (EM). The E-step re-selects $\mathbf{c}_c$ from a predefined candidate codebook $\mathcal{B}$ based on the current class-mean hash code $\overline{\mathbf{b}}_c \in [-1,+1]^N$, setting $\mathbf{c}_c \leftarrow \arg\min_{\mathbf{b}\in\mathcal{B}} \text{dist}_H(\mathbf{b}, \overline{\mathbf{b}}_c)$, and the M-step updates the network toward the assigned center.

Both formulations pull all samples of a class toward a single center, thereby compressing intra-class semantic structure in the binary space. Proposition 1 (Appendix D) formalizes the limit case, showing that at the loss minimum $\mathbf{b}$ is constant on each class of the training set and thus encodes no intra-class ranking. In practice, the loss does not reach zero, but the same-class collapse pressure still compresses intra-class structure (as shown by the low intra-class ρ of all hash-center baselines in Table 1). Adding more bits does not close this gap: baseline intra-class ρ stays within 0.04–0.32 at every code length, against 0.26–0.63 for RICH.

3.2 Method Overview

To address this problem, RICH encodes inter-class and intra-class signals into independent bit subspaces (Figure 1). A shared backbone f maps an input image $(\mathbf{x}, y)$ to a D_f-dimensional feature $\mathbf{f}$, which feeds two independent hash heads. The center head produces an N_{ctr}-bit code trained by $\mathcal{L}_{\text{center}}$ with the coarse label y, and the residual head produces an N_{res}-bit code trained by $\mathcal{L}_{\text{res}}$ with sub-cluster labels obtained by K-means on features of the fixed pretrained backbone f_{pt}. The head is called residual because this clustering runs inside each class, so its supervision is the variation the superclass label leaves unexplained, discretized without additional annotation. Section 3.7 details the retrieval strategies.

3.3 Dual-Head Architecture

Concretely, the two heads are independent hash layers on the shared backbone, and the final code concatenates their binarized outputs:

$$\mathbf{b} = [\text{sign}(h_{\text{ctr}}(\mathbf{f})),\ \text{sign}(h_{\text{res}}(\mathbf{f}))] \in \{-1,+1\}^N, \tag{2}$$

where $\mathbf{f}$ is the backbone feature and $h_{\text{ctr}}, h_{\text{res}}$ denote the real-valued head outputs before Sign. The two heads share the backbone but have independent losses and codebooks, jointly optimizing their respective bit subspaces. We set $N_{\text{ctr}} = N_{\text{res}} = \lfloor N/2 \rfloor$, assigning equal importance to the two objectives (Appendix E empirically validates this choice).

Head Design. Each head is a single Linear($D_f \rightarrow N_h$) layer followed by Sign, with $N_h \in \{N_{\text{ctr}}, N_{\text{res}}\}$ for the center and residual head, respectively. Each head has a binary codebook initialized with the max-min Hamming scheme (Yuan et al., 2020; Wang et al., 2023; Yin et al., 2026), and its samples are trained toward their assigned centers. Following the multi-head structure of Yin et al. (2026), we split the N_h bits into sub-heads of size d_{sub}, the smallest power of two at least $\lceil \log_2 M \rceil$ for the head's codebook size M. Proposition 2 (Appendix D) guarantees that each sub-head can then express M distinct codes without collision. The two heads differ in (i) the supervision

signal, (ii) the codebook size, and (iii) how centers are assigned: one per class, one-to-one across classes, for the center head, and several per class from a pool the classes may share for the residual head. The center head follows the standard hash-center paradigm (Yuan et al., 2020; Hoe et al., 2021; Wang et al., 2023; Yin et al., 2026) with C class centers $\mathbf{C}_{\text{ctr}} \in \{-1,+1\}^{C\times N_{\text{ctr}}}$ assigned from a candidate codebook $\mathcal{B}_{\text{ctr}}$ of size $M{=}2C$, supervised by the superclass label y (the factor of two ensures that unused candidate centers remain available during each reassignment step). The residual head uses a shared pool $\mathcal{B}_{\text{res}}$ of $M{=}2K_{\text{opt}}^{\max}$ candidate centers (same rationale as the center head), where $K_{\text{opt}}^{\max}$ is the global sub-cluster cap of Section 3.4. Each class c independently selects $K_{\text{opt},c}$ centers $\mathbf{C}_{\text{res},c} \in \{-1,+1\}^{K_{\text{opt},c}\times N_{\text{res}}}$ from it via the E-step, supervised by sub-cluster labels. We index these per class rather than collecting them into one matrix, because $K_{\text{opt},c}$ varies by class and different classes may select the same center. Proposition 3 (Appendix D) shows that the full-code Hamming distance between any two classes is bounded below by the center head's inter-class distance, so sharing a residual pool does not collapse inter-class separation. Section 3.5 details the center-assignment update.

3.4 Sub-cluster Determination and Label Generation

To construct the supervisory signal for the residual head, RICH applies the same discretization principle as hash-center methods (approximating a continuous semantic space with a discrete supervisory signal) but at the intra-class level, without extra annotations.

K_{opt} Determination. The expressiveness of the residual head depends on the number of sub-clusters K. A larger K allows finer intra-class encoding but reduces per-cluster sample size, making clusters unstable. RICH fixes K by a closed-form rule rather than by cluster-quality model selection: the largest K that leaves enough samples in every sub-cluster, capped at $2C$. Concretely, the algorithm takes three quantities: 1) the average per-class sample count $\bar{N}_c = \frac{1}{C}\sum_c N_c$, 2) a fixed floor $\tau_{\min}^{\text{floor}}$ on the number of samples a sub-cluster must contain, and 3) the resulting per-cluster minimum $\tau_{\min} = \max\big(\tau_{\min}^{\text{floor}}, \bar{N}_c/(2C)\big)$. The global cap is $K_{\text{opt}}^{\max} = \min\big(2C, \lfloor \bar{N}_c/\tau_{\min} \rfloor\big)$, and each class c with N_c samples uses $K_{\text{opt},c} = \min\big(K_{\text{opt}}^{\max}, \max(1, \lfloor N_c/\tau_c \rfloor)\big)$ with its own floor $\tau_c = \max(\tau_{\min}^{\text{floor}}, N_c/(2C))$, so that its sub-clusters hold τ_c samples on average. The upper bound $K \le 2C$ is empirically motivated, matching the scale of the center head's codebook size ($M{=}2C$) so that sub-cluster boundaries do not exceed the inter-class separation scale. The residual sub-head size d_{res} is set to the smallest power of two satisfying $d_{\text{res}} \ge \lceil \log_2(2K_{\text{opt}}^{\max}) \rceil$, following the same capacity rule as the center head. When bits for the inter-class codebook are insufficient, or no stable clustering is possible, the model falls back to operating as a single-center head. The full procedure is given in Appendix F. The sensitivity of K_{opt} to $\tau_{\min}^{\text{floor}}$ is analyzed in Appendix G. Retrieval performance is largely insensitive to this choice on the two ImageNet benchmarks and varies more on CIFAR-100, and $\tau_{\min}^{\text{floor}}{=}10$ gives the best average mAP@all.

Sub-cluster Label Generation. Because pretrained features already internalize semantic similarity, we generate sub-cluster labels by running K-means (Lloyd, 1982) on the ℓ_2-normalized pretrained features for each class c, obtaining labels $s_i \in \{0,\dots,K_{\text{opt},y_i}-1\}$, indexed within the sample's own class, without extra annotation. The residual head treats these labels as supervision.

3.5 Hash Center Optimization

We update the center assignments of the two heads independently using a periodic E-step schedule in Appendix H. Because the joint problem of choosing an assignment together with the codebook is combinatorial, both heads use the greedy one-to-one rule of Yin et al. (2026) (pseudocode in Appendix I), in which clusters are visited in a random order drawn at each E-step, and each takes the nearest center not yet assigned, so no cluster consistently picks first.

Center Assignment. Both heads reassign centers using the same procedure. First, we compute the average binarized hash output over the samples in each cluster c, $\boldsymbol{\mu}_c = \frac{1}{|\mathcal{S}_c|}\sum_{\mathbf{x}\in\mathcal{S}_c} \operatorname{sign}(h_\bullet(f(\mathbf{x})))$, where $h_\bullet$ is the head being updated ($\bullet \in \{\text{ctr},\text{res}\}$), f the shared backbone, and $\mathcal{S}_c$ the set of training samples in cluster c. Then, visiting clusters in a random permutation π, we perform greedy one-to-one assignment from the corresponding head's codebook:

$$\tilde{\mathbf{c}}_{\pi(j)} = \operatorname*{arg\,min}_{\mathbf{b}\in\mathcal{B}\setminus\mathcal{A}_j} \operatorname{dist}_H\big(\boldsymbol{\mu}_{\pi(j)},\ \mathbf{b}\big)\,, \quad \mathcal{A}_{j+1} = \mathcal{A}_j \cup \big\{\tilde{\mathbf{c}}_{\pi(j)}\big\}, \tag{3}$$

where $\mathcal{B}$ is the head's candidate codebook, $\mathcal{A}_0 = \emptyset$, and π is freshly sampled at every E-step. For the center head c ranges over the C superclasses; for the residual head, the rule runs inside each class over its $K_{\text{opt},c}$ sub-clusters, with $\mathcal{A}$ reset per class, so two classes may select the same residual center.

3.6 Training Objective

The two heads have independent losses but share backbone parameters, so inter-class and intra-class information are jointly built on the same feature representation. The residual loss therefore also shapes the features the center head reads, so the two objectives need not trade off; Section 4.2 measures the net effect. The overall loss, averaged over the samples of a batch, is the sum of the two heads' per-sample losses:

$$\mathcal{L} = \mathcal{L}_{\text{center}} + \alpha \mathcal{L}_{\text{res}} \tag{4}$$

Both heads use an OrthoHash-style cosine-margin cross-entropy (Hoe et al., 2021) together with a separate quantization regularizer (OrthoHash absorbs quantization into its single objective, whereas we keep the regularizer as an explicit term following Yuan et al. (2020)). We set $\alpha = 1$ by default, assigning equal weight to the two objectives. The sensitivity of α is analyzed in Section 4.2. Write $\mathbf{z}_i^{\bullet} = h_{\bullet}(\mathbf{f}_i)$ for the pre-activation output of head $\bullet \in \{\text{ctr}, \text{res}\}$. Logits are computed as the inner product between the normalized output $\hat{\mathbf{z}}_i^{\bullet} = \mathbf{z}_i^{\bullet} / \|\mathbf{z}_i^{\bullet}\|_2$ and the normalized codebook centers.

The center head loss uses the superclass label y_i as supervision and learns inter-class structure:

$$\mathcal{L}_{\text{center}} = \mathcal{L}_{\text{CE}}\Big(\gamma_{\text{ctr}}\, \hat{\mathbf{z}}_i^{\text{ctr}} \cdot \hat{\mathbf{C}}_{\text{ctr}}^{\top},\ y_i\Big) + \lambda \mathcal{L}_Q(\mathbf{z}_i^{\text{ctr}}). \tag{5}$$

The residual head loss uses the sub-cluster label s_i as supervision and learns the intra-class structure:

$$\mathcal{L}_{\text{res}} = \mathcal{L}_{\text{CE}}\Big(\gamma_{\text{res}}\, \hat{\mathbf{z}}_i^{\text{res}} \cdot \hat{\mathbf{C}}_{\text{res},y_i}^{\top},\ s_i\Big) + \lambda \mathcal{L}_Q(\mathbf{z}_i^{\text{res}}). \tag{6}$$

Eqs. 5 and 6 are stated for one sample i, unlike the dataset-level $\mathcal{L}_{\text{center}}$ of Eq. 1. Here $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss and $\mathcal{L}_Q(\mathbf{z}) = \frac{1}{N_h} \| \, |\mathbf{z}| - \mathbf{1} \|_2^2$ is the quantization loss for an N_h-bit head output $\mathbf{z}$ ($\lambda = 0.1$; sensitivity analysis in Appendix J). The logit scales are set to $\gamma_{\text{ctr}} = \sqrt{N_{\text{ctr}}}$ and $\gamma_{\text{res}} = \sqrt{N_{\text{res}}}$. Eqs. 5 and 6 omit the cosine margin m and the tanh relaxation of Sign used in training (Appendix H). Because cosine logits are bounded in $[-1, 1]$, this fixed scale sharpens the softmax, and tying it to the head's bit count follows Hoe et al. (2021). We jointly optimize both heads with Adam (Kingma & Ba, 2015).

Training Loop. Training runs in three stages, preparation (K_{opt}, sub-cluster labels, codebook initialization), batch training of both heads with Adam, and periodic E-step reassignment. The full pseudocode is Algorithm 2 in Appendix I.

3.7 Retrieval Strategy

If a standard Hamming search is performed on the full N-bit code, the residual bits interfere with the inter-class ranking and degrade superclass retrieval (Section 4.3). Proposition 4 (Appendix D) shows that this interference is structural. Single-pass retrieval on the full code never ranks a different-class item before a same-class item if $\Delta \leq \delta$, where δ is the minimum pairwise Hamming distance between the assigned class centers and Δ is the maximum pairwise distance in the residual pool. The Plotkin bound (Plotkin, 1960) caps δ near $N_{\text{ctr}}/2$, while the residual centers are max-min spread over the same $N_{\text{res}} = N_{\text{ctr}}$ bits, so nothing keeps Δ below δ. The inter-class columns of Table 2 show the resulting interference in all nine settings.

To preserve inter-class retrieval while exploiting intra-class similarity, we propose a two-stage retrieval strategy requiring no additional training or parameter changes. For a query $\mathbf{x}_q$ with code $\mathbf{b}_q = [\mathbf{b}_q^{\text{ctr}}, \mathbf{b}_q^{\text{res}}]$ and a database $\mathcal{X}$: (1) *inter-class stage* sorts the database by $\text{dist}_H(\mathbf{b}_q^{\text{ctr}}, \mathbf{b}_i^{\text{ctr}})$ on the center bits only, giving a coarse superclass-level ranking; (2) *intra-class stage* breaks the ties of stage (1) by the residual-bit distance $\text{dist}_H(\mathbf{b}_q^{\text{res}}, \mathbf{b}_i^{\text{res}})$, which within a tie group equals ordering by the full-code distance since the center term is constant.

Proposition 5 (Appendix D) guarantees that this procedure never inverts a pair that stage (1) separated because the ranking is lexicographic on $(\text{dist}_H^{\text{ctr}}, \text{dist}_H^{\text{res}})$, so when the center distances differ, the residual-bit distance is never consulted. Stage (2) resolves only the ties that stage (1) left.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets. Our setup uses only coarse superclass labels during training and evaluates intra-class quality against the fine-grained subclass labels as ground truth. We use three primary benchmarks: CIFAR-100 (Krizhevsky, 2009) (C=20 superclasses, 5 subclasses each), ImageNet-S20 (C=20, 10 subclasses each, grouped by WordNet (Miller, 1995) from ImageNet (Deng et al., 2009)), and ImageNet-S50 (C=50, 10 subclasses each). We additionally evaluate on two CIFAR-100 variants with class imbalance defined by Grčić et al. (2024), CIFAR68 (68 subclasses after removing 32) and CIFAR-SI (all 100 subclasses with per-subclass sample retention from 32% to 99%), and on CUB-super37 (Wah et al., 2011), constructed from CUB-200-2011 by grouping 167 species into C=37 family-level superclasses with 2–25 subclasses each. Training/database/query split sizes for all six datasets are listed in Appendix H.

Baselines. We compare against the state-of-the-art supervised hash-center methods CSQ (Yuan et al., 2020), OrthoHash (Hoe et al., 2021), MDS (Wang et al., 2023), SHC (Chen et al., 2025), and CRH (Yin et al., 2026), the most recent of which appeared in 2026. DSTH (Han et al., 2026) also places several centers per class, but targets cross-modal retrieval and is not directly comparable here. For a fair comparison, all methods use the same ResNet-34 (He et al., 2016) backbone.

Evaluation Metrics. We report mAP@all as the inter-class retrieval metric, where relevance is defined by the superclass label, computed on center bits for RICH and on the full code for baselines. *Subclass mAP@100* uses the ground-truth subclass label as the relevance criterion, directly measuring fine-grained retrieval quality. The cutoff 100 is sufficiently small relative to the per-class database size (2,700 per class across all three primary datasets). We further evaluate using two intra-class structure-preservation metrics, defined in full in Appendix H.

Intra-class Spearman ρ (Spearman, 1904) measures how faithfully the pairwise distance ranking between a class's sub-clusters survives in the binary space. Its reference centers come from the pretrained backbone while sub-cluster membership comes from the evaluated model, so ρ charges training and binarization together.

Semantic Recall@k (SR@k) reflects what binarization alone preserves. Adapting the standard Recall@k of approximate nearest neighbor search to the intra-class setting, it measures, among the database items of the query's own superclass, the overlap between the k nearest neighbors under the model's continuous features and the top k returned by hash retrieval. We normalize by $\min(k, n_{y_q})$, where n_{y_q} is the number of database items in that superclass. As k grows, the chance of overlap between the two neighbor sets increases and the metric loses discriminative power, so we use k=100.

Implementation Details. ResNet-34 is the default backbone for all main experiments, matching the setting recent work evaluates these baselines in (Chen et al., 2025; Yin et al., 2026). We initialize it with ImageNet weights and precompute the K-means sub-cluster labels in its fixed feature space. Appendix H lists the optimizer, schedule, and checkpoint selection, and Section 4.4 repeats the evaluation with a ViT-B/16 backbone.

4.2 MAIN RESULTS

Table 1 compares RICH with the hash-center baselines on the three datasets in terms of mAP, subclass mAP@100, SR@100, and intra-class ρ at 16/32/64 bits. RICH outperforms every baseline by large margins on all three intra-class metrics (subclass mAP@100, SR@100, intra-class ρ) while maintaining competitive inter-class mAP at matched bit budget.

Inter-class retrieval. At matched inter-class bit budget, RICH leads on all nine settings, by $+0.16$ to $+6.12$%p over the second-best baseline. The margin is smallest on CIFAR-100 16b ($+0.16$%p), where the center head has 16 bits for 20 superclasses, and largest on ImageNet-S50 ($+4.19$ to $+6.12$%p). When the total code length is matched instead, inter-class retrieval uses only the $N/2$ center bits of an N-bit RICH code (Table 2, center-only (ours) rows). RICH then leads the best baseline in three of the nine settings (all three datasets at 64 bits; $+1.0$–$+1.7$%p) and trails in the other six by 0.2–3.3%p. The inter-class lead is not unconditional: on CUB-super37 and CIFAR68,

Table 1: Per-method results on three datasets. Inter-class mAP compares baselines' full N-bit code against RICH's N_{ctr} center bits at matched bit width, so RICH's inter-class entry at width N comes from a $2N$-bit model. Intra-class metrics compare all methods at the same total N.

Dataset	Method	Inter-class			Intra-class								
		mAP@all (%) ↑			Sub mAP@100 (%) ↑			SR@100 (%) ↑			intra ρ ↑		
		16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b
CIFAR-100	CSQ (Yuan et al., 2020)	61.67	64.03	64.44	19.57	20.00	20.20	17.89	20.32	24.23	0.165	0.213	0.112
	OrthoHash (Hoe et al., 2021)	56.11	58.52	58.90	19.71	20.52	20.98	14.08	17.65	21.70	0.142	0.087	0.042
	MDS (Wang et al., 2023)	57.29	62.64	62.95	18.62	20.39	29.63	14.26	16.41	22.86	0.097	0.093	0.176
	SHC (Chen et al., 2025)	58.92	61.66	65.91	19.64	21.83	21.66	14.47	16.11	16.22	0.163	0.063	0.117
	CRH (Yin et al., 2026)	59.81	60.75	60.79	18.79	20.84	19.61	16.05	18.11	17.87	0.099	0.115	0.129
	RICH **(ours)**	**61.83**	**67.09**	**67.00**	**43.59**	**46.46**	**49.49**	**36.29**	**40.10**	**49.65**	**0.276**	**0.257**	**0.264**
ImageNet-S20	CSQ (Yuan et al., 2020)	81.88	83.00	84.51	13.88	14.25	14.58	10.32	12.89	14.89	0.215	0.232	0.219
	OrthoHash (Hoe et al., 2021)	80.12	81.98	82.58	13.59	13.90	13.83	8.47	9.55	11.30	0.193	0.214	0.230
	MDS (Wang et al., 2023)	78.74	80.00	85.47	13.45	14.18	18.33	9.79	11.18	14.05	0.194	0.225	0.304
	SHC (Chen et al., 2025)	80.29	84.76	85.99	13.25	14.75	18.54	8.69	9.91	13.88	0.227	0.253	0.322
	CRH (Yin et al., 2026)	80.39	80.62	82.45	13.76	13.76	13.92	8.71	9.77	10.38	0.233	0.187	0.240
	RICH **(ours)**	**84.58**	**87.02**	**87.73**	**54.70**	**66.12**	**64.83**	**37.15**	**52.42**	**56.92**	**0.516**	**0.484**	**0.631**
ImageNet-S50	CSQ (Yuan et al., 2020)	53.91	58.16	60.68	13.08	13.27	13.75	14.91	18.30	22.60	0.239	0.199	0.235
	OrthoHash (Hoe et al., 2021)	54.06	56.00	58.48	12.79	13.58	14.14	13.25	15.68	18.69	0.231	0.231	0.251
	MDS (Wang et al., 2023)	49.98	56.75	58.69	13.04	13.65	17.79	14.00	16.32	21.82	0.250	0.255	0.286
	SHC (Chen et al., 2025)	54.54	60.09	62.57	12.94	13.66	18.51	12.73	14.86	20.60	0.218	0.247	0.281
	CRH (Yin et al., 2026)	55.18	58.73	60.40	13.02	13.44	13.70	13.27	15.99	17.33	0.244	0.253	0.275
	RICH **(ours)**	**59.68**	**64.28**	**68.69**	**48.60**	**52.34**	**55.37**	**40.68**	**46.54**	**54.85**	**0.428**	**0.492**	**0.496**

RICH trails the best baseline in inter-class mAP by 0.5–3.9%p while keeping its intra-class lead consistently (Appendices K and L). Sub-cluster supervision pushes the shared backbone toward richer features that also help the center head: weakening the residual loss degrades inter-class mAP on CIFAR-100 (Appendix K), which isolates the effect of the extra bits because the architecture is held fixed. The benefit holds where superclasses are visually separable and reverses on CUB-super37. Results are stable across random seeds (Appendix M).

Intra-class structure preservation. Subclass mAP@100 measures subclass-level retrieval from a model trained on superclass labels only. The prior baselines remain at 12∼30% across all three datasets and bit widths, while RICH reaches 1.7∼3.5× the second-best at 64 bits (CIFAR-100 49.49%, ImageNet-S20 64.83%, ImageNet-S50 55.37%). The residual bits retain the fine-grained structure that the pretrained embedding captures and that superclass-label-only hashing discards, and handing the same sub-cluster labels to the single-head baselines reaches subclass mAP@100 of up to 49.4%, above RICH at 32 bits, while collapsing their inter-class mAP@all to 8∼29% against RICH's 62∼67%: it is the dual head, not the labels alone, that keeps both (Appendix N).

The structure-preservation metrics show the same pattern. At 64 bits, the SR@100 margin over the second-best baseline is +25.4 to +42.0%p and intra-class ρ is 1.5–2.0× the second-best on the three datasets (Table 1). The gain is in the learning rather than in the extra bits: replacing the residual bits with ITQ fitted on the same pretrained features, with the center head left as it is, costs 6.5–34.9%p of subclass mAP@100 and 18.7–36.4%p of SR@100 in every setting (Appendix A).

Residual weight α. On CUB-super37, where 37 visually similar bird families make the inter-class task difficult, reducing α from 1.0 to 0.5 improves inter-class mAP from 82.47% to 83.92% and subclass mAP@100 from 41.05% to 44.91% at the cost of intra-class ρ (0.372 to 0.312). α is thus a practical knob when the inter-class task is itself fine-grained (Appendix K).

4.3 Ablation Study

Bit-space separation. In the inter-class columns of Table 2, full-code retrieval drops superclass mAP@all below center-bit retrieval on all three datasets and bit widths, consistent with the bit-space conflict of Section 3.1 and the interference condition of Proposition 4, while the center bits alone suffice for inter-class retrieval.

Retrieval strategy. The intra-class columns of Table 2 compare center-only, full-code single-pass, and two-stage retrieval on subclass mAP@100, with the superclass oracle of Table 12 as the reference point. Center-only stays at 12∼22%, so the residual bits are the source of the subclass-level gain. Full-code single-pass is slightly higher than two-stage in most settings because residual similarity offsets small center-distance differences within a class, the same mechanism that sacrifices

Table 2: Bit-space separation and retrieval strategy ablation at 16/32/64 total bits. Bold marks the best result per dataset and bit width. The superclass oracle, which replaces stage-1 with ground-truth superclass filtering and keeps the same lexicographic stage 2, is reported separately in Table 12.

Dataset	Inter-class				Intra-class			
	Configuration	mAP@all (%) ↑			Configuration	Sub mAP@100 (%) ↑		
		16b	32b	64b		16b	32b	64b
CIFAR-100	center-only (ours)	**58.37**	**61.83**	**67.09**	center-only	19.09	19.51	21.33
	full code	38.09	50.17	60.21	two-stage (ours)	43.59	46.46	49.49
					full code	**43.68**	**48.96**	**53.76**
ImageNet-S20	center-only (ours)	**79.50**	**84.58**	**87.02**	center-only	13.68	13.93	14.87
	full code	47.95	70.82	81.36	two-stage (ours)	54.70	66.12	64.83
					full code	**55.23**	**67.40**	**69.06**
ImageNet-S50	center-only (ours)	**52.79**	**59.68**	**64.28**	center-only	12.40	13.09	14.50
	full code	23.50	43.06	55.23	two-stage (ours)	**48.60**	52.34	55.37
					full code	48.16	**54.49**	**59.00**

mAP@all in stage 1 (Proposition 4). Because inter-class discrimination comes first, we use two-stage retrieval, which preserves the center-head ranking (Proposition 5), as the default. Within the top-100, the two strategies retrieve the same superclasses, and the advantage of two-stage retrieval lies in superclass recall at depth, where single-pass full code loses 5.7–31.6%p of mAP@all. Two-stage retrieval differs from the oracle by at most 10.9 %p, so the center head already filters superclasses effectively (per-query breakdown in Appendix O).

Other ablations. Three further design choices are swept in the appendix: the bit allocation between the heads (Appendix E), the clustering floor $\tau_{\text{min}}^{\text{floor}}$ (Appendix G), and the quantization weight λ (Appendix J).

4.4 Robustness and Qualitative Analysis

Qualitative and embedding analysis. Figure 2 (Appendix B) illustrates the intra-class contrast: for a kayak query from the `watercraft` superclass, RICH returns kayaks in all top-3 results, whereas the baselines return a mix of lifeboats, sailboats, and other subclasses of the same superclass. A t-SNE analysis of the pretrained features (Appendix P) shows that where subclasses overlap in the embedding, the finer K_{opt} clustering still yields locally consistent sub-partitions.

Backbone, class imbalance, seeds, and failure modes. With a ViT-B/16 backbone (Appendix Q), RICH again achieves the highest subclass mAP@100 and SR@100 on all three datasets and bit widths, so its intra-class advantage survives the backbone change, though intra-class ρ is harder for every method under ViT. On two class-imbalanced CIFAR-100 variants (Grčić et al., 2024), RICH shows the same pattern as on the balanced benchmarks (Appendix L). Appendix M sweeps random seeds. The failure condition, a pretrained embedding whose sub-clusters are not separable, is shown on the best and worst superclasses in Appendix P and discussed in Appendix R.

5 Conclusion

This paper formalized a structural limitation of coarse-to-fine representation learning in binary codes. One center per class forces the inter-class and intra-class objectives to conflict in a single bit space, and more bits do not resolve the conflict. To the best of our knowledge, we are the first to formalize intra-class semantic ordering as an independent retrieval-quality dimension in coarse-label hashing and to introduce metrics for it (Intra-class Spearman ρ and Semantic Recall@k). RICH separates the two objectives into an N_{ctr}-bit center head and an N_{res}-bit residual head with a data-driven K_{opt}, and retrieves center bits first, residual bits within each class. On CIFAR-100, ImageNet-S20, and ImageNet-S50, splitting the code cost no inter-class quality where the superclasses are visually separable, because the sub-cluster loss also improves the shared backbone: RICH leads on all nine inter-class settings at matched inter-class bit budget, improves intra-class ρ by $1.2\sim2.2\times$ in every setting, and achieves $1.7\sim4.5\times$ subclass mAP@100 over the second-best baseline at every bit width. Its limitations, the α trade-off on visually similar superclasses and the dependence on the pretrained embedding, are discussed in Appendix R.

AI USE STATEMENT

Generative AI was used in preparing this manuscript in the following ways, all under the direction of the authors. A large language model was used to edit the paper for readability, to draft revisions of individual sentences and paragraphs that the authors then reviewed, to manage and refactor experiment scripts, to tabulate experimental results, to identify and summarize related literature and to format and verify bibliography entries against public registries (Crossref, arXiv, OpenReview), to check the consistency of the numbers quoted in the text against the result tables, and to draft the statements in this section. For the tasks for which ICLR requires disclosure, generative AI was not used to generate data, to develop the theoretical framework, to formulate or prove the propositions, to propose hypotheses, to design the methodology or the experiments, to implement the method, to clean or reformat datasets, or to interpret the results. The authors verified all AI-assisted text by reading each edited sentence against the source it summarizes or the table it cites, and checked every added citation against the cited work's own record. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses public image benchmarks (CIFAR-100, ImageNet subsets, CUB-200-2011) with their existing coarse labels and a publicly available pretrained backbone; no human subjects, personal data, or new data collection are involved. Appendix S discusses the societal considerations of improved image retrieval, including privacy and surveillance risks that apply to retrieval systems in general, and the corresponding mitigations.

REPRODUCIBILITY STATEMENT

All components needed to reproduce the results are specified in the paper and its appendices: the datasets, splits, and superclass constructions in Section 4.1; the network, optimizer, loss, and E-step schedule in Appendix H; the parameter determination procedure as pseudocode in Appendix F (Algorithm 1) and the full training loop in Appendix I (Algorithm 2); the formal statements and proofs of Propositions 1–5 in Appendix D; and multi-seed statistics in Appendix M. The anonymized code for RICH and for the baseline reproductions, run under the same setting, is available at `https://anonymous.4open.science/r/RICH-286C`.

REFERENCES

Wenbin An, Feng Tian, Ping Chen, Siliang Tang, Qinghua Zheng, and QianYing Wang. Fine-grained category discovery under coarse-grained supervision with hierarchical weighted self-contrastive learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1314–1323, 2022. doi: 10.18653/v1/2022.emnlp-main.85.

Guy Bukchin, Eli Schwartz, Kate Saenko, Ori Shahar, Rogerio Feris, Raja Giryes, and Leonid Karlinsky. Fine-grained angular contrastive learning with coarse labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Philip S. Yu. HashNet: Deep learning to hash by continuation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5608–5617, 2017.

Li Chen, Rui Liu, Yuxiang Zhou, Xudong Ma, Yong Chen, and Dell Zhang. Deep hashing with semantic hash centers for image retrieval. *ACM Transactions on Information Systems*, 43(6): 1–38, 2025. doi: 10.1145/3749983.

Zhen-Duo Chen, Li-Jun Zhao, Zi-Chao Zhang, Xin Luo, and Xin-Shun Xu. Characteristics matching based hash codes generation for efficient fine-grained image retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17273–17281, 2024.

Zhen-Duo Chen, Li-Jun Zhao, Yongxin Wang, Yu-Wei Zhan, Xiao-Qian Liu, Xin Luo, and Xin-Shun Xu. Shared latent characteristic anchored hash codes generation for efficient fine-grained image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–16, 2026. doi: 10.1109/TPAMI.2026.3723930.

Quan Cui, Qing-Yuan Jiang, Xiu-Shen Wei, Wu-Jun Li, and Osamu Yoshie. ExchNet: A unified hashing network for large-scale fine-grained image retrieval. In *Proceedings of the European Conference on Computer Vision*, volume 12348 of *Lecture Notes in Computer Science*, 2020. doi: 10.1007/978-3-030-58580-8_12.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Chen Feng and Ioannis Patras. MaskCon: Masked contrastive learning for coarse-labelled dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2013. doi: 10.1109/TPAMI.2012.193.

Matej Grčić, Artyom Gadetsky, and Maria Brbić. Fine-grained classes and how to find them. In *Proceedings of the International Conference on Machine Learning*, PMLR, 2024.

Shuo Han, Qibing Qin, Jinkui Hou, Wenfeng Zhang, and Lei Huang. Deep SoftTriple hashing for multi-label cross-modal retrieval. *Neural Networks*, 198:108713, 2026. doi: 10.1016/j.neunet.2026.108713.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Jiun Tian Hoe, Kam Woh Ng, Tianyu Zhang, Chee Seng Chan, Yi-Zhe Song, and Tao Xiang. One loss for all: Deep hashing with a single cosine similarity based learning objective. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pp. 604–613, 1998.

Young Kyun Jang and Nam Ik Cho. Self-supervised product quantization for deep unsupervised image retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12085–12094, 2021.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002. doi: 10.1145/582415.582418.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Wu-Jun Li, Sheng Wang, and Wang-Cheng Kang. Feature learning based deep supervised hashing with pairwise labels. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 1711–1717, 2016.

Xue Li, Jiong Yu, Junlong Cheng, Ziyang Li, and Chen Bian. Semantic preservation-based hash code generation for fine-grained image retrieval. *Expert Systems with Applications*, 271:126668, 2025. doi: 10.1016/j.eswa.2025.126668.

Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28 (2):129–137, 1982. doi: 10.1109/TIT.1982.1056489.

George A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11): 39–41, 1995.

Morris Plotkin. Binary codes with specified minimum distance. *IRE Transactions on Information Theory*, 6(4):445–450, 1960. doi: 10.1109/TIT.1960.1057584.

Zexuan Qiu, Jiahong Liu, Yankai Chen, and Irwin King. HiHPQ: Hierarchical hyperbolic product quantization for unsupervised image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Yang Shen, Xuhao Sun, Xiu-Shen Wei, Qing-Yuan Jiang, and Jian Yang. SEMICON: A learning-to-hash solution for large-scale fine-grained image retrieval. In *Proceedings of the European Conference on Computer Vision*, volume 13674 of *Lecture Notes in Computer Science*, 2022. doi: 10.1007/978-3-031-19781-9_31.

Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.

Hugo Touvron, Alexandre Sablayrolles, Matthijs Douze, Matthieu Cord, and Hervé Jégou. Grafit: Learning fine-grained image representations with coarse labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Dan Wang, Heyan Huang, Chi Lu, Bo-Si Feng, Guihua Wen, Liqiang Nie, and Xian-Ling Mao. Supervised deep hashing for hierarchical labeled data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018a.

Jingdong Wang, Ting Zhang, Jingkuan Song, Nicu Sebe, and Heng Tao Shen. A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):769–790, 2018b. doi: 10.1109/TPAMI.2017.2699960.

Liangdao Wang, Yan Pan, Cong Liu, Hanjiang Lai, Jian Yin, and Ye Liu. Deep hashing with minimal-distance-separated hash centers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23455–23464, 2023.

Peng Wang, Yong Li, Lin Zhao, and Xiu-Shen Wei. Learning attribute-aware hash codes for fine-grained image retrieval via query optimization. In *Proceedings of the International Conference on Machine Learning*, volume 267 of *PMLR*, 2025.

Xiu-Shen Wei, Yang Shen, Xuhao Sun, Han-Jia Ye, and Jian Yang. A^2-Net: Learning attribute-aware hash codes for large-scale fine-grained image retrieval. In *Advances in Neural Information Processing Systems*, volume 34, pp. 5720–5730, 2021.

Xiu-Shen Wei, Yang Shen, Xuhao Sun, Peng Wang, and Yuxin Peng. Attribute-aware deep hashing with self-consistency for large-scale fine-grained image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):13904–13920, 2023. doi: 10.1109/TPAMI.2023.3299563.

Zhirong Wu, Alexei A. Efros, and Stella X. Yu. Improving generalization via scalable neighborhood component analysis. In *Proceedings of the European Conference on Computer Vision*, 2018.

Shuo Yin, Zhiyuan Yin, Yuqing Hou, Rui Liu, Yong Chen, and Dell Zhang. Codebook-centric deep hashing: End-to-end joint learning of semantic hash centers and neural hash function. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 12018–12026, 2026. doi: 10.1609/aaai.v40i14.38190.

Li Yuan, Tao Wang, Xiaopeng Zhang, Francis E. H. Tay, Zequn Jie, Wei Liu, and Jiashi Feng. Central similarity quantization for efficient image and video retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3083–3092, 2020.

Haien Zeng, Hanjiang Lai, and Jian Yin. Simultaneous region localization and hash coding for fine-grained image retrieval. *arXiv preprint arXiv:1911.08028*, 2019.

Ming Zhang, Xuefei Zhe, Le Ou-Yang, Shifeng Chen, and Hong Yan. Semantic hierarchy preserving deep hashing for large-scale image retrieval. In *Proceedings of the International Conference on Machine Vision Applications (MVA)*, 2021. doi: 10.23919/MVA51890.2021.9511401.

A UNSUPERVISED RESIDUAL BINARIZATION

Table 3 verifies that RICH's intra-class performance stems from the learned residual head rather than discretization of the pretrained embedding: we replace the $N/2$ residual bits with $N/2$ ITQ (Gong et al., 2013) bits fitted on the same fixed pretrained features used for sub-cluster discovery, keeping the center head unchanged. Because the center head is shared, stage-1 retrieval is identical between RICH and this variant (RICH+ITQ), and only the stage-2 bits differ. We additionally pair the five baseline hash-center methods with the same ITQ bits and include Sign and ITQ as unsupervised references, reporting the metrics of Table 1. Subclass mAP@100 uses the same lexicographic two-stage retrieval as Table 1 for methods with a center–residual split and full-code single-pass for Sign and ITQ.

Table 3: Unsupervised residual binarization at 16, 32, and 64 total bits. Each "X+ITQ" appends $N/2$ ITQ bits fit on training-set features to baseline X's full $N/2$-bit code. Sign and ITQ binarize pretrained features directly. Inter-class mAP compares baselines' full $N/2$-bit code against RICH's N_{ctr} center bits at matched bit width.

Dataset	Method	Inter-class			Intra-class								
		mAP@all (%) ↑			Sub mAP@100 (%) ↑			SR@100 (%) ↑			intra ρ ↑		
		16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b
CIFAR-100	Sign	14.36	14.74	13.13	21.52	29.88	36.48	24.00	32.25	38.91	0.669	0.704	0.743
	ITQ	19.42	22.01	23.12	23.29	32.67	38.94	24.54	34.46	43.84	**0.696**	**0.718**	**0.746**
	CSQ+ITQ	55.30	61.67	64.03	29.47	36.14	43.70	15.43	19.03	21.78	0.251	0.054	0.132
	OrthoHash+ITQ	54.89	56.11	58.52	29.44	34.88	41.86	15.02	21.58	27.14	0.189	0.289	0.251
	MDS+ITQ	51.64	57.29	62.64	29.57	35.87	42.94	15.55	20.31	27.20	0.282	0.186	0.292
	SHC+ITQ	54.02	58.92	61.66	28.86	35.36	41.90	15.40	22.90	27.45	0.252	0.310	0.243
	CRH+ITQ	57.27	59.81	60.75	29.82	36.78	41.79	14.81	19.68	26.49	0.208	0.239	0.266
	RICH+ITQ	**58.37**	**61.83**	**67.09**	30.00	35.85	43.02	14.75	21.45	26.22	0.266	0.236	0.254
	RICH **(ours)**	**58.37**	**61.83**	**67.09**	**43.59**	**46.46**	**49.49**	**36.29**	**40.10**	**49.65**	0.276	0.257	0.264
ImageNet-S20	Sign	40.97	37.76	28.27	28.29	49.33	64.35	18.72	34.63	49.78	**0.654**	0.778	0.814
	ITQ	59.32	63.17	60.12	27.60	48.99	**65.56**	16.80	32.75	49.70	0.633	**0.788**	**0.860**
	CSQ+ITQ	76.79	81.88	83.00	22.17	30.38	47.40	11.76	16.17	22.71	0.403	0.402	0.427
	OrthoHash+ITQ	78.22	80.12	81.98	22.11	30.82	49.96	11.50	15.83	23.78	0.383	0.416	0.472
	MDS+ITQ	72.32	78.74	80.00	22.12	30.37	46.79	10.42	15.94	26.65	0.324	0.435	0.552
	SHC+ITQ	76.50	80.29	84.76	22.38	30.61	47.36	11.85	15.36	26.30	0.369	0.408	0.536
	CRH+ITQ	78.39	80.39	80.62	22.58	30.66	49.14	11.61	15.83	22.80	0.395	0.419	0.442
	RICH+ITQ	**79.50**	**84.58**	**87.02**	22.78	31.24	48.95	11.05	15.99	28.99	0.420	0.434	0.585
	RICH **(ours)**	**79.50**	**84.58**	**87.02**	**54.70**	**66.12**	64.83	**37.15**	**52.42**	**56.92**	0.516	0.484	0.631
ImageNet-S50	Sign	14.32	18.67	19.81	19.19	35.05	51.89	20.06	32.68	47.27	**0.680**	0.746	0.787
	ITQ	19.40	27.38	32.28	19.14	35.02	51.67	19.28	31.52	46.78	0.676	**0.757**	**0.814**
	CSQ+ITQ	50.30	53.91	58.16	20.33	28.87	40.25	14.08	21.86	26.17	0.261	0.459	0.419
	OrthoHash+ITQ	49.65	54.06	56.00	20.41	29.20	40.36	13.76	20.89	26.97	0.261	0.473	0.478
	MDS+ITQ	46.62	49.98	56.75	19.92	29.21	39.91	15.21	21.48	28.38	0.422	0.449	0.491
	SHC+ITQ	49.21	54.54	60.09	20.22	29.77	40.18	13.86	20.40	27.17	0.311	0.458	0.501
	CRH+ITQ	**53.12**	55.18	58.73	20.14	29.79	40.97	14.40	20.93	27.12	0.269	0.476	0.495
	RICH+ITQ	52.79	**59.68**	**64.28**	20.55	30.44	41.45	13.62	20.86	29.73	0.418	0.461	0.463
	RICH **(ours)**	52.79	**59.68**	**64.28**	**48.60**	**52.34**	**55.37**	**40.68**	**46.54**	**54.85**	0.428	0.492	0.496

At 32 and 64 total bits the inter-class mAP of the center head equals the matched-width entries of Table 1, where RICH matches or exceeds every baseline. At 16 total bits, the center head has only 8 bits, yet still leads the best baseline by 1.1%p on CIFAR-100 and ImageNet-S20 and trails it by 0.3%p on ImageNet-S50. Sign and ITQ, which binarize pretrained features without a learned hash function, stay below 64% mAP@all on the full N bits across all configurations.

On subclass mAP@100, RICH outperforms RICH+ITQ by a large margin in every configuration. SR@100 shows a similar pattern. None of the X+ITQ baselines matches RICH on either intra-class metric, even when its main code achieves competitive inter-class mAP. Among the unsupervised references, ITQ marginally exceeds RICH on subclass mAP@100 in one configuration (ImageNet-S20, 64 bits) but falls behind on all others.

Sign and ITQ achieve the highest intra-class ρ on all three datasets (0.63–0.86) because they preserve pretrained feature distances by construction. They stay below RICH on SR@100 in every configuration and on subclass mAP@100 in every configuration but ImageNet-S20 at 64 bits, confirming that faithfulness to pretrained distances does not imply effective subclass retrieval. Together, these results show that the residual head learns intra-class structure beyond what unsupervised discretization provides.

B Qualitative Results

This section visually compares the Top-3 and Top-5 retrieval results of RICH and the baselines on ImageNet-S20 at 32 bits.



Figure 2: Top-3 retrieval results on ImageNet-S20 at 32 bits for a `watercraft` (superclass) query.

Figure 3 shows retrieval results for four superclasses (songbirds, furniture, primates, ungulates). The baselines generally succeed at superclass-level classification but mix together a variety of subclasses within the same superclass. RICH, in contrast, tends to retrieve consistently within the same subclass as the query: it returns the query's robin-like birds for songbirds, bookcases for furniture, orangutans for primates, and zebras for ungulates. This visually confirms that the residual bits preserve subclass-level semantic structure that conventional single-head methods collapse into a shared code.

C Continuous Re-ranking

Section 1 argues that re-ranking hash candidates with continuous features cannot restore intra-class order, because re-ranking permutes a candidate set that the code has already fixed. Table 4 measures this on all three primary datasets. *+top-100 re-rank* reorders each method's Hamming top-100 (for RICH, the top-100 under two-stage ordering) by cosine similarity in the ImageNet-pretrained ResNet-34 space. *+class re-rank* predicts the superclass from each method's own hash centers and ranks the entire predicted class by cosine similarity. It approximates the best achievable by storing continuous features for the whole database and uses no ground-truth label, so it is not an oracle. No method is retrained for this experiment.

Because +top-100 re-rank only permutes a fixed candidate set, subclass P@100 is, by construction, identical to the Hamming column in all fifty-four settings. Only subclass mAP@100 rises, because that metric also rewards ordering within the list. The candidate set is decided by the code alone, and at 64 bits RICH reaches a subclass P@100 of $42.51/56.59/45.15\%$ on CIFAR-100/ImageNet-S20/ImageNet-S50 against $21.59/12.21/10.28\%$ for the best baseline, a factor of 2.0–4.6 that no re-ranking of the top-100 can change. Re-ranking narrows the subclass mAP@100 gap from $+19.9$–$+51.4$ %p to $+3.9$–$+12.1$ %p, since moving the few relevant items a baseline does retrieve to the top raises its AP, but the gap remains. RICH still leads every baseline in all nine settings. The advantage therefore lies in the candidate set the code produces, not in the ordering applied afterward. Class-wide re-ranking (the *+class re-rank* columns) does close the gap, as expected, but only by abandoning the Hamming candidate set and exhaustively scoring the entire predicted class with stored continuous features. In that setting, RICH stays within 3.5 %p of the best baseline in every setting and is ahead in five of the eighteen, while predicting the superclass from its $N/2$ center bits where the baselines use all N.

D Theoretical Analysis

This section gives the formal statements and proofs of the five propositions referenced in the main body. Proposition 1 is referenced in Section 3.1. Propositions 2 and 3 are referenced in Section 3.3. Propositions 4 and 5 are referenced in Section 3.7.

Proposition 1 (Bit-space conflict)**.** *Let* $\mathbf{b}(\mathbf{x}) = \operatorname{sign}(h(\mathbf{x})) \in \{-1, +1\}^N$ *be the binary code of a hash function trained under the hash-center objective* $\mathcal{L}_{center}$ *in Eq. 1 with a single binary center* $\mathbf{c}_c$



Figure 3: Per-method Top-5 retrieval results on ImageNet-S20 at 32 bits. Each panel shows a query image from a single superclass, along with the corresponding retrieval results for each method.

per class c, and assume the loss is minimized in the strict sense that $h(\mathbf{x}_i)$ is driven to its target $\mathbf{c}_{y_i}$, hence $\mathbf{b}(\mathbf{x}_i) = \mathbf{c}_{y_i}$, for every training sample. Then $\mathbf{b}$ is constant on each class of the training set, i.e., $\mathbf{b}(\mathbf{x}_i) = \mathbf{b}(\mathbf{x}_j)$ whenever $y_i = y_j$. Consequently, $\mathbf{b}$ cannot encode any intra-class semantic ranking among training samples.

Proof. Strict minimization of $\mathcal{L}_{\text{center}}$ assigns every sample of class c to the same target $\mathbf{c}_c$, so $\mathbf{b}(\mathbf{x}_i) = \mathbf{b}(\mathbf{x}_j) = \mathbf{c}_c$ for any pair $(\mathbf{x}_i, \mathbf{x}_j)$ with $y_i = y_j = c$. Therefore $\text{dist}_H(\mathbf{b}(\mathbf{x}_i), \mathbf{b}(\mathbf{x}_j)) = 0$ for

Table 4: Effect of re-ranking on subclass retrieval quality at 16/32/64 bits. *Hamming*: pure Hamming-distance retrieval. *+top-100 re-rank*: re-rank the Hamming top-100 by pretrained cosine similarity. *+class re-rank*: predict superclass from hash centers, retrieve within the predicted class by cosine. For RICH, the Hamming ranking (and hence the top-100 candidates for re-ranking) follows the two-stage ordering, and superclass prediction uses only the center bits; baselines use their full code throughout. Sub mAP@100 and Sub P@100 measure subclass-level retrieval quality among the top-100 retrieved items. Both re-ranking conditions require a 512-d float embedding (2 KB) stored for every database item, against 8 B for a 64-bit code.

Dataset	Method	Hamming						+top-100 re-rank						+class re-rank					
		Sub mAP@100			Sub P@100			Sub mAP@100			Sub P@100			Sub mAP@100			Sub P@100		
		16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b
CIFAR-100	CSQ (Yuan et al., 2020)	19.57	20.00	20.20	14.56	14.71	15.28	42.80	43.74	43.58	14.56	14.71	15.28	56.53	**57.18**	56.30	45.99	**46.27**	46.08
	OrthoHash (Hoe et al., 2021)	19.71	20.52	20.98	14.32	14.65	15.23	44.79	44.28	44.71	14.32	14.65	15.23	55.90	55.67	55.40	44.09	44.47	44.04
	MDS (Wang et al., 2023)	18.62	20.39	29.63	13.60	15.36	21.59	42.26	44.59	51.10	13.60	15.36	21.59	56.62	56.41	56.12	45.66	45.71	44.98
	SHC (Chen et al., 2025)	19.64	21.83	21.66	14.31	16.23	16.36	43.55	45.91	46.96	14.31	16.23	16.36	55.09	56.19	**57.94**	44.08	45.49	**46.92**
	CRH (Yin et al., 2026)	18.79	20.84	19.61	14.01	14.59	14.05	43.49	44.26	42.25	14.01	14.59	14.05	57.02	55.99	56.75	**46.53**	44.25	45.97
	RICH **(ours)**	**43.59**	**46.46**	**49.49**	**36.64**	**39.36**	**42.51**	**56.84**	**56.52**	**58.88**	**36.64**	**39.36**	**42.51**	**57.15**	56.87	57.75	46.01	45.63	46.87
ImageNet-S20	CSQ (Yuan et al., 2020)	13.88	14.25	14.58	8.99	9.20	9.53	65.99	66.05	66.09	8.99	9.20	9.53	75.73	75.59	75.50	65.44	65.14	65.09
	OrthoHash (Hoe et al., 2021)	13.59	13.90	13.83	8.80	8.93	8.96	65.65	65.92	66.67	8.80	8.93	8.96	75.65	76.00	75.82	65.10	65.50	65.39
	MDS (Wang et al., 2023)	13.45	14.18	18.33	8.59	8.96	11.54	66.52	66.62	68.76	8.59	8.96	11.54	75.57	75.40	75.85	64.65	63.79	65.50
	SHC (Chen et al., 2025)	13.25	14.75	18.54	8.69	9.85	12.21	66.28	66.58	69.25	8.69	9.85	12.21	75.36	**76.29**	76.11	64.98	**65.77**	65.71
	CRH (Yin et al., 2026)	13.76	13.76	13.92	8.87	8.75	8.90	67.12	66.02	65.73	8.87	8.75	8.90	**75.95**	75.81	75.78	**65.64**	64.89	65.61
	RICH **(ours)**	**54.70**	**66.12**	**64.83**	**45.30**	**57.57**	**56.59**	**74.81**	**76.61**	**76.51**	**45.30**	**57.57**	**56.59**	75.81	75.49	**76.21**	65.09	64.25	**66.69**
ImageNet-S50	CSQ (Yuan et al., 2020)	13.08	13.27	13.75	7.50	7.44	7.84	65.17	65.30	64.57	7.50	7.44	7.84	68.31	68.36	68.69	53.71	54.38	54.68
	OrthoHash (Hoe et al., 2021)	12.79	13.58	14.14	7.38	7.49	7.77	64.80	65.97	65.18	7.38	7.49	7.77	68.18	68.48	68.72	53.79	54.15	54.38
	MDS (Wang et al., 2023)	13.04	13.65	17.79	7.01	7.48	10.09	65.41	65.64	66.42	7.01	7.48	10.09	68.02	68.20	67.94	52.88	54.05	53.46
	SHC (Chen et al., 2025)	12.94	13.66	18.51	7.09	7.68	10.28	65.69	64.99	66.81	7.09	7.68	10.28	**68.74**	**68.77**	68.26	54.10	**54.93**	54.51
	CRH (Yin et al., 2026)	13.02	13.44	13.70	7.27	7.50	7.61	65.12	65.63	65.34	7.27	7.50	7.61	68.55	67.91	68.50	**54.45**	54.71	54.71
	RICH **(ours)**	**48.60**	**52.34**	**55.37**	**35.36**	**41.37**	**45.15**	**70.33**	**69.84**	**70.94**	**35.36**	**41.37**	**45.15**	68.04	67.99	**69.45**	53.45	51.47	**56.84**

every same-class pair. Any ranking induced by Hamming distance over the same-class samples is a constant zero and cannot reflect intra-class semantic ordering. □

Proposition 2 (Codebook capacity for multi-head decomposition)**.** *Let a head of dimension r be split into n sub-heads of equal size d_{sub}, so that d_{sub} divides r and $r = n \cdot d_{sub}$. The concatenated code then ranges over 2^r distinct values. For a target codebook size M, choosing $d_{sub} \geq \lceil \log_2 M \rceil$ guarantees that M distinct codes can be assigned to M classes (or sub-clusters) without collision in any single sub-head. Both d_{sub} and the head widths N_{ctr}, N_{res} are powers of two in RICH, with $d_{sub} \leq N_h$, so the divisibility condition holds in every configuration reported.*

Proof. Each sub-head outputs a binary vector of length d_{sub}, taking $2^{d_{\text{sub}}}$ distinct values. The multi-head code, formed by concatenation, takes values in the Cartesian product of n such spaces, of size $(2^{d_{\text{sub}}})^n = 2^{n \cdot d_{\text{sub}}} = 2^r$. For the no-collision claim, it suffices to verify that a single sub-head can host M distinct codes, which holds whenever $2^{d_{\text{sub}}} \geq M$, i.e., $d_{\text{sub}} \geq \lceil \log_2 M \rceil$. □

Proposition 3 (Residual disambiguation under shared center pool)**.** *Let $\mathbf{b}(\mathbf{x}) = [\mathbf{b}^{ctr}(\mathbf{x}), \mathbf{b}^{res}(\mathbf{x})]$ be the concatenated N-bit code from Eq. 2. For two classes $c \neq c'$, suppose the center head reaches its targets, $\mathbf{b}^{ctr}(\mathbf{x}) = \mathbf{c}_c^{ctr}$ for $y(\mathbf{x}) = c$ (and similarly for c'). Then for any pair $(\mathbf{x}, \mathbf{x}')$ with $y(\mathbf{x}) = c$, $y(\mathbf{x}') = c'$,*

$$dist_H(\mathbf{b}(\mathbf{x}), \mathbf{b}(\mathbf{x}')) \;\geq\; dist_H(\mathbf{c}_c^{ctr}, \mathbf{c}_{c'}^{ctr}),$$

regardless of whether the residual codes $\mathbf{b}^{res}(\mathbf{x}), \mathbf{b}^{res}(\mathbf{x}')$ are assigned to the same residual center.

Proof. Hamming distance is additive across disjoint coordinate blocks: $\text{dist}_H(\mathbf{b}, \mathbf{b}') = \text{dist}_H(\mathbf{b}^{\text{ctr}}, \mathbf{b}'^{\text{ctr}}) + \text{dist}_H(\mathbf{b}^{\text{res}}, \mathbf{b}'^{\text{res}})$. Both terms are nonnegative. By assumption, $\text{dist}_H(\mathbf{b}^{\text{ctr}}, \mathbf{b}'^{\text{ctr}}) = \text{dist}_H(\mathbf{c}_c^{\text{ctr}}, \mathbf{c}_{c'}^{\text{ctr}})$, which is strictly positive whenever $c \neq c'$ because the greedy one-to-one E-step (Section 3.5) gives each class a distinct center. The residual term is bounded below by zero (with equality when residual centers match), so the total distance is bounded below by $\text{dist}_H(\mathbf{c}_c^{\text{ctr}}, \mathbf{c}_{c'}^{\text{ctr}})$. □

Proposition 4 (Interference threshold for single-pass retrieval)**.** *Let both heads reach their targets, i.e., $\mathbf{b}^{ctr}(\mathbf{x}) = \mathbf{c}_{y(\mathbf{x})}^{ctr}$ and $\mathbf{b}^{res}(\mathbf{x})$ is a residual center of class $y(\mathbf{x})$, for every sample. Let δ be the minimum pairwise Hamming distance between the assigned class centers, and Δ the maximum pairwise Hamming distance between residual centers in the shared pool. If $\Delta \leq \delta$, single-pass*

Hamming retrieval on the full code never ranks a different-class item strictly before a same-class item, for every query and every residual assignment. Conversely, if $\Delta > \delta$, there is an assignment giving some class two distinct residual centers, under which a different-class item is ranked strictly before a same-class item. The threshold is therefore tight whenever a class receives more than one residual center; if every class receives exactly one, all same-class codes coincide and no interference arises for any Δ.

Proof. (*Sufficiency.*) For a query $\mathbf{x}_q$ with class c, a same-class item $\mathbf{x}_s$ ($y_s = c$) has center distance 0 and full-code distance at most Δ. A different-class item $\mathbf{x}_d$ ($y_d \neq c$) has center distance at least δ, so by the additivity of Hamming distance across disjoint blocks, $\text{dist}_H(\mathbf{b}_q, \mathbf{b}_d) \geq \delta$. When $\Delta \leq \delta$, every same-class item is at most as far as every different-class item, so no different-class item is ranked strictly before it.

(*Tightness.*) When $\Delta > \delta$, let classes c, c' be at center distance δ and let $\mathbf{r}_1, \mathbf{r}_2$ be pool centers at distance Δ. Let class c select both, which requires $K_{\text{opt},c} \geq 2$, and let class c' also select $\mathbf{r}_1$; the shared pool permits both (Section 3.3). A query of class c at $\mathbf{r}_1$ is then at full-code distance Δ from a same-class item at $\mathbf{r}_2$ but at distance δ from a class-c' item at $\mathbf{r}_1$. Since $\Delta > \delta$, the different-class item is ranked strictly before the same-class item. □

The condition $\Delta \leq \delta$ is not guaranteed in the RICH setting. Placing the C assigned centers in a code of length N_{ctr}, the Plotkin bound (Plotkin, 1960) caps $\delta \leq (N_{\text{ctr}}/2) \cdot C/(C-1)$, which converges to $N_{\text{ctr}}/2$. The residual pool is max-min initialized over the same budget $N_{\text{res}} = N_{\text{ctr}}$, so its largest pairwise distance Δ is of the same order, and nothing in training keeps it below δ. Since both sides scale with the bit budget, adding bits does not resolve this. The inter-class columns of Table 2 reflect this in all nine settings.

Proposition 5 (Rank preservation under two-stage retrieval)**.** *Let two-stage retrieval sort the database by $dist_H(\mathbf{b}_q^{ctr}, \mathbf{b}_i^{ctr})$ and break ties by the residual-bit distance $dist_H(\mathbf{b}_q^{res}, \mathbf{b}_i^{res})$ (equivalently the full-code distance, which differs from it only by the constant center term inside a tie group). Then for any query q and any database items $\mathbf{x}_i, \mathbf{x}_j$ with $dist_H(\mathbf{b}_q^{ctr}, \mathbf{b}_i^{ctr}) < dist_H(\mathbf{b}_q^{ctr}, \mathbf{b}_j^{ctr})$, the procedure places $\mathbf{x}_i$ strictly before $\mathbf{x}_j$. That is, stage 2 resolves only the ties that stage 1 left and never inverts a pair that stage 1 separated.*

Proof. The procedure sorts lexicographically by the pair $(\text{dist}_H^{\text{ctr}}, \text{dist}_H^{\text{res}})$. When the first components differ, the second is never consulted, so the stage-1 ordering is retained. The residual-bit distance is applied only within the equivalence classes of equal center distance, where $\text{dist}_H(\mathbf{b}_q, \mathbf{b}_i) = \text{dist}_H^{\text{ctr}} + \text{dist}_H^{\text{res}}$ has a constant first term, so ordering by either distance is the same. □

E Bit Allocation Sensitivity

Table 5 analyzes the effect of varying the bit allocation ratio between the center and residual heads. While RICH uses an equal split ($N_{\text{ctr}} = N_{\text{res}} = N/2$) by default, we evaluate three ratios at 32 bits on CIFAR-100, ImageNet-S20, and ImageNet-S50.

As the center-ratio increases, mAP@all improves while SR@100 and intra-class ρ decrease on all three datasets, and subclass mAP@100 decreases on CIFAR-100 and ImageNet-S50 (on ImageNet-S20 it is flat between 0.25 and 0.50), confirming the inherent trade-off between inter-class and intra-class objectives. Subclass mAP@100 reflects this trade-off through the two-stage lexicographic retrieval. At lower center-ratios, more bits are allocated to the residual head, so the second stage refines larger tied groups with richer intra-class information. Subclass mAP@100 is therefore highest at 0.25 on CIFAR-100 and ImageNet-S50 and within 0.13%p of the maximum on ImageNet-S20, despite the weaker inter-class ranking. On all three datasets, moving from 0.25 to 0.50 gains more mAP@all than it costs in subclass mAP@100, while moving from 0.50 to 0.75 costs more than it gains, so we adopt the equal split as the default.

F Parameter Determination

Algorithm 1 gives the complete parameter determination procedure described in Section 3.4.

Table 5: Bit allocation sensitivity at 32 bits total. The center-ratio column denotes the fraction of bits allocated to the center head.

Dataset	center-ratio	Inter-class		Intra-class			
		Bits	mAP@all (%)	Bits	Sub mAP@100 (%)	SR@100 (%)	intra ρ
CIFAR-100	0.25 (1:3)	8	56.77	32	49.40	44.25	0.406
	0.50 (2:2)	16	61.83	32	46.46	40.10	0.257
	0.75 (3:1)	24	64.05	32	43.92	36.31	0.190
ImageNet-S20	0.25 (1:3)	8	81.42	32	65.99	53.85	0.584
	0.50 (2:2)	16	84.58	32	66.12	52.42	0.484
	0.75 (3:1)	24	85.69	32	54.37	39.32	0.437
ImageNet-S50	0.25 (1:3)	8	52.29	32	57.28	51.88	0.595
	0.50 (2:2)	16	59.68	32	52.34	46.54	0.492
	0.75 (3:1)	24	61.73	32	45.98	39.01	0.381

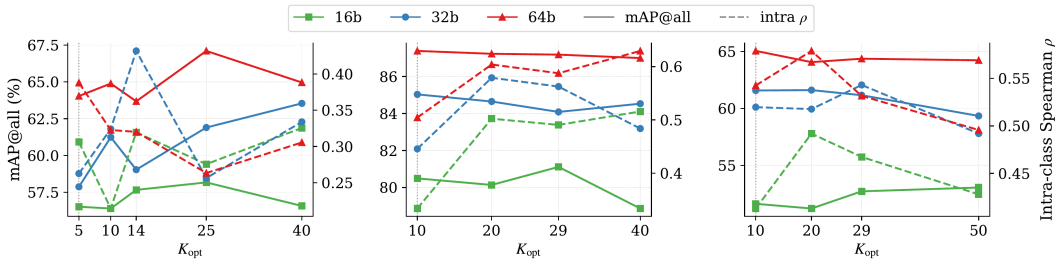
Algorithm 1 RICH Parameter Determination

Require: Total bits N; number of classes C; per-class sample counts $\{N_c\}_{c=1}^{C}$; minimum cluster size $\tau_{\text{min}}^{\text{floor}}$

Ensure: N_{ctr}, N_{res}, $K_{\text{opt}}^{\max}$, $\{K_{\text{opt},c}\}_{c=1}^{C}$

1: $r \leftarrow \lfloor N/2 \rfloor$ ▷ equal-importance bit budget per head
2: $d_{\text{ctr}} \leftarrow \text{smallest_pow2}\big(\lceil \log_2(2C) \rceil\big)$ ▷ center sub-head size, satisfies Prop. 2 for $M{=}2C$
3: $n_{\text{base}} \leftarrow \lfloor r/d_{\text{ctr}} \rfloor$ ▷ number of center sub-heads that fit in r bits
4: **if** $n_{\text{base}} < 1$ **then return** FALLBACK TO SINGLE HEAD
5: **end if** ▷ insufficient bits for inter-class codebook
6: $\bar{N}_c \leftarrow \frac{1}{C}\sum_{c=1}^{C} N_c$ ▷ average per-class sample count
7: $\tau_{\text{min}} \leftarrow \max\big(\tau_{\text{min}}^{\text{floor}},\ \bar{N}_c/(2C)\big)$ ▷ per-cluster minimum-sample-count floor
8: $K_{\text{opt}}^{\max} \leftarrow \min\big(2C,\ \lfloor \bar{N}_c/\tau_{\text{min}} \rfloor\big)$ ▷ cap by $2C$; cluster reliability via τ_{min}
9: **if** $K_{\text{opt}}^{\max} < 2$ **then return** FALLBACK TO SINGLE HEAD
10: **end if** ▷ cannot form stable sub-clusters
11: $d_{\text{res}} \leftarrow \text{smallest_pow2}\big(\lceil \log_2(2K_{\text{opt}}^{\max}) \rceil\big)$ ▷ residual sub-head size, Prop. 2 for $M{=}2K_{\text{opt}}^{\max}$
12: $n_{\text{res}} \leftarrow \lfloor r/d_{\text{res}} \rfloor$ ▷ number of residual sub-heads that fit in r bits
13: **if** $n_{\text{res}} = 0$ **then**
14: $n_{\text{res}} \leftarrow 1$; $d_{\text{res}} \leftarrow r$ ▷ degenerate fallback: single sub-head of width r
15: **end if**
16: $N_{\text{ctr}} \leftarrow n_{\text{base}} \times d_{\text{ctr}}$; $N_{\text{res}} \leftarrow n_{\text{res}} \times d_{\text{res}}$ ▷ realized bit budgets per head
17: **for** $c = 1$ **to** C **do**
18: $\tau_c \leftarrow \max\big(\tau_{\text{min}}^{\text{floor}},\ N_c/(2C)\big)$ ▷ per-class reliability floor
19: $K_{\text{opt},c} \leftarrow \min\big(K_{\text{opt}}^{\max},\ \max(1,\ \lfloor N_c/\tau_c \rfloor)\big)$ ▷ per-class sub-cluster count, capped by global $K_{\text{opt}}^{\max}$
20: **end for**
21: **return** N_{ctr}, N_{res}, $K_{\text{opt}}^{\max}$, $\{K_{\text{opt},c}\}_{c=1}^{C}$

Walkthrough. The algorithm balances three constraints in sequence: (i) the bit budget is split equally between the two heads (line 1, $r = \lfloor N/2 \rfloor$); (ii) the center head's bit budget must accommodate $M{=}2C$ candidate centers without sub-head collision, so d_{ctr} is set by Proposition 2 (line 2) and the number of center sub-heads n_{base} is determined by what fits in r bits (line 3); (iii) the residual head's sub-cluster count $K_{\text{opt}}^{\max}$ is jointly bounded by the cluster-reliability floor τ_{min} (line 7) and the empirical separation cap $K \leq 2C$ (line 8), then the residual head's bit dimension d_{res} is set analogously to d_{ctr} (line 11). When either head's bit budget is insufficient to store its codebook (lines 4 and 9), the algorithm gracefully degrades to a single-center head, reducing to a standard hash-center method. Lines 17–20 compute per-class $K_{\text{opt},c}$ to allow class-level adaptation when class sizes vary; for the three balanced primary datasets (Section 4.1), $K_{\text{opt},c}$ is constant across c, whereas on CIFAR68 and CIFAR-SI (Appendix L) it varies with N_c. Here smallest_pow2(x) returns the smallest



(a) CIFAR-100-super (true $K=5$) **(b) ImageNet-super20 (true $K=10$)** **(c) ImageNet-super50 (true $K=10$)**

Figure 4: K_{opt} sensitivity. The x-axis shows K_{opt}. The left y-axis shows query-set mAP@all (%, solid lines) on the N_{ctr} center bits and the right y-axis shows intra-class Spearman ρ (dashed lines) on the full code. Colors correspond to total 16b / 32b / 64b. The vertical dotted lines mark the true K (actual number of subclasses).

power of two that is $\geq x$, ensuring sub-head dimensions are amenable to standard binary-code routines.

Time complexity. The algorithm itself is $O(C)$ in arithmetic operations and runs once before training; its cost is negligible compared to the one-time K-means and the network update.

G K_{OPT} Sensitivity Analysis

We analyze the sensitivity to $\tau_{\text{min}}^{\text{floor}}$, the only external parameter of the K_{opt} algorithm, on all three datasets. Varying $\tau_{\text{min}}^{\text{floor}} \in \{5, 10, 17, 25, 50\}$ yields $K_{\text{opt}} \in \{40, 25, 14, 10, 5\}$ on CIFAR-100; for $\tau_{\text{min}}^{\text{floor}} \in \{10, 17, 25, 50\}$, $K_{\text{opt}} \in \{40, 29, 20, 10\}$ on ImageNet-S20 and $K_{\text{opt}} \in \{50, 29, 20, 10\}$ on ImageNet-S50.

Figure 4 shows that the effect of K depends on the dataset. On ImageNet-S20 and ImageNet-S50, mAP@all is largely insensitive to K (within about 2 %p). On CIFAR-100, which has fewer subclasses, both metrics vary non-monotonically with K, and mAP@all fluctuates more. This empirical pattern matches the design rationale of Section 3.4. Increasing K refines the residual supervisory signal but reduces per-cluster sample size, and the relative weight of these two effects depends on how well-separated the subclasses are in the embedding space. Since inter-class retrieval remains the primary goal of deep hashing, we give priority to retrieval performance and select $\tau_{\text{min}}^{\text{floor}}$=10, which gives the highest mAP@all on the query set averaged on the nine dataset–bit settings.

H Implementation Details

This section describes the implementation details required to reproduce the experiments.

Metric definitions. Intra-class Spearman ρ averages, over classes, the rank correlation between sub-cluster distances in the binary space and in the pretrained feature space:

$$\rho_{\text{intra}} = \frac{1}{C}\sum_{c=1}^{C} \text{Spearman}\Big(\{d_{\cos}(\bar{\mathbf{b}}_{c,k}, \bar{\mathbf{b}}_{c,l})\}_{k<l}\,,\ \{d_{\cos}(\boldsymbol{\nu}_{c,k}, \boldsymbol{\nu}_{c,l})\}_{k<l}\Big)\,, \tag{7}$$

Here $\boldsymbol{\nu}_{c,k}$ is the ℓ_2-normalized $K_{\text{opt},c}$-means center of sub-cluster k of class c, computed once on the pretrained backbone (Section 3.4), $\bar{\mathbf{b}}_{c,k}$ is the ℓ_2-normalized mean hash code of the database items assigned to that center by the evaluated model's features, and $d_{\cos}(\mathbf{a}, \mathbf{b}) = 1 - \mathbf{a}^\top \mathbf{b}$ denotes cosine distance. Higher values indicate that the binary space better preserves the discovered intra-class structure without requiring ground-truth subclass labels. In short, ρ reflects how much pretrained intra-class structure survives training and binarization together.

Semantic Recall@k is

$$\text{SR@k}(q) = \frac{\left|\mathcal{N}_k^{\text{feat}}(q) \cap \mathcal{N}_k^{\text{hash}}(q)\right|}{\min(k,\, n_{y_q})}, \tag{8}$$

with n_{y_q} is the number of database items in the query's superclass.

Dataset Splits. Table 6 lists the training, database, and query set sizes used for every dataset in Section 4.1. For CIFAR68 and CIFAR-SI, the images remaining after subclass removal or subsampling are split in the same proportions as CIFAR-100 (1/12 training, 1/60 query, and the rest database).

Table 6: Dataset splits (number of images).

Dataset	C	Training	Database	Query
CIFAR-100	20	5,000	54,000	1,000
ImageNet-S20	20	10,000	54,000	2,000
ImageNet-S50	50	25,000	135,000	5,000
CIFAR68	20	3,400	36,720	680
CIFAR-SI	20	3,224	34,829	644
CUB-super37	37	5,004	3,998	835

Network Architecture. We initialize ResNet-34 with ImageNet pretrained weights, remove the original fully connected layer, and place two independent linear layers on top: `hash_layer_center`: Linear$(512 \to N_{\text{ctr}})$ and `hash_layer_res`: Linear$(512 \to N_{\text{res}})$.

Optimization. We use the Adam optimizer (lr$= 10^{-4}$, $\beta_1 = 0.5$, $\beta_2 = 0.999$, weight decay$= 10^{-5}$) for up to 300 epochs, with gradient clipping (L_2 norm ≤ 1.0) and automatic mixed-precision (AMP) training in fp16. The batch size is 128, and the learning rate follows cosine annealing to 10^{-7} over the training epochs. All baselines are trained with the same optimizer, schedule, and batch size.

Loss Function. Both heads apply the cosine-margin cross-entropy of Eqs. 5 and 6 independently with $\alpha = 1$. We use a cosine margin $m = 0.2$, the scales $\gamma_{\text{ctr}} = \sqrt{N_{\text{ctr}}}$ and $\gamma_{\text{res}} = \sqrt{N_{\text{res}}}$, and a quantization regularizer with $\lambda = 0.1$ (computed after applying tanh).

E-step Schedule. For the first 20 epochs, the center assignment is updated every epoch; afterwards it is updated every 5 epochs. Running averages are reset after each update. We initialize both the center and residual codebooks using the max-min Hamming-distance scheme.

Evaluation. The model is evaluated every 10 epochs (and at the final epoch) on the query set against a fixed 1/10 subset of the database. This evaluation protocol is identical for all methods, and all reported numbers are computed on the full database. Since baselines do not explicitly optimize intra-class structure, checkpoint selection for all baselines follows the standard best mAP@all criterion. For RICH, the primary criterion is the two-stage (lexicographic) mAP@all, and intra-class ρ serves as a tie-breaker. A new checkpoint replaces the current one if its mAP@all exceeds the best value so far by more than 1 percentage point, or if it lies within ± 1 percentage point of that value and has a higher ρ than the current checkpoint. ρ is computed on the same database subset using the K_{opt} sub-clusters and requires no subclass labels. Because RICH jointly optimizes inter-class and intra-class objectives, selecting among near-equal mAP checkpoints by the secondary intra-class criterion is a natural part of the method. Checkpoint selection therefore reads the query set, as is common in this literature. Every method selects on the same query set and database subset, but the absolute numbers should be read as best-epoch results rather than as held-out estimates. For mAP@100, the per-query average precision averages precision values at relevant ranks, normalized by the number of relevant items retrieved within the top 100. Queries for which no same-subclass item appears in the top 100 are excluded from the mAP@100 average, as their average precision is undefined; the number of remaining queries is the sum of the two n columns of Table 13.

Data Preprocessing. During training we apply Resize(256) $\to$ RandomCrop(224) $\to$ RandomHorizontalFlip. For the database and queries, we apply Resize(256) $\to$ CenterCrop(224). All images are normalized with the ImageNet standard statistics (mean$= [0.485, 0.456, 0.406]$, std$= [0.229, 0.224, 0.225]$). All experiments are conducted on NVIDIA L40S (46 GB, $\times 2$) and RTX PRO 5000 (48 GB, $\times 6$) GPUs with a fixed random seed. Appendix M provides multi-seed robustness results.

I Training Procedure

Algorithm 2 gives the complete training procedure summarized in Section 3.6.

Algorithm 2 RICH Training Procedure

Require: Training set $\mathcal{D}$; fixed pretrained backbone f_{pt}; total bits N; epochs T
Ensure: Trained hash functions $h_{\text{ctr}},\ h_{\text{res}}$

— Preparation —

1: $K_{\text{opt},c} \leftarrow \text{K_OPT}(N, C, N_c)$ for each class c ▷ determine sub-cluster count via Algorithm 1
2: **for** $c = 1$ **to** C **do**
3: $K_{\text{opt},c}$-means on $\{\ell_2\text{-norm}(f_{\text{pt}}(\mathbf{x}_i)) \mid y_i = c\} \rightarrow$ sub-cluster labels $\{s_i\}$ ▷ discrete intra-class supervision; precomputed once
4: **end for**
5: Initialize $\mathcal{B}_{\text{ctr}}$ and $\mathcal{B}_{\text{res}}$ with the max-min Hamming scheme ▷ maximally separated initial centers

— Training loop —

6: **for** $t = 1$ **to** T **do**
7: **for** each mini-batch of $\mathcal{D}$ **do**
8: $\mathcal{L} \leftarrow \mathcal{L}_{\text{center}} + \alpha\mathcal{L}_{\text{res}}$ (Eq. 4); $\theta \leftarrow \theta - \nabla_\theta \mathcal{L}$ via Adam ▷ joint update of both heads on shared backbone
9: **end for**
10: **Center E-step:** greedy one-to-one center reassignment under a random permutation π (Eq. 3) ▷ adapt $\mathbf{C}_{\text{ctr}}$ to current network output
11: **Residual E-step:** for each class c, greedy one-to-one reassignment under a random permutation π (Eq. 3) ▷ adapt $\mathbf{C}_{\text{res},c}$ at the per-class level
12: **end for**

Walkthrough. The training loop is a generalized EM scheme over two coupled assignment problems. The *Preparation* block (lines 1–5) is run once: line 1 sets the architecture parameters via Algorithm 1, lines 2–4 produce the $K_{\text{opt},c}$-means sub-cluster labels in the fixed pretrained feature space (these are the supervisory signal for the residual head and remain constant throughout training), and line 5 initializes both codebooks so that all centers are maximally Hamming-separated. The *Training loop* block (lines 6–12) alternates three operations per epoch: (i) batch-level network update via Adam over the joint loss $\mathcal{L} = \mathcal{L}_{\text{center}} + \alpha\mathcal{L}_{\text{res}}$ (lines 7–9, the M-step over the network parameters θ), (ii) Center E-step (line 10), which reassigns the C class centers in $\mathbf{C}_{\text{ctr}}$ to better fit the current network output via Eq. 3, and (iii) Residual E-step (line 11), which independently reassigns the $K_{\text{opt},c}$ sub-cluster centers per class from $\mathcal{B}_{\text{res}}$. The two E-steps are independent (they touch disjoint codebooks), so they can be executed in parallel.

Time complexity. The dominant per-epoch cost is the network update at $O(|\mathcal{D}| \cdot N \cdot D_f)$ for forward and backward passes through the linear hash heads (excluding the backbone, which is shared with all baselines). The Center E-step is a bipartite minimum-Hamming-distance assignment of C classes to $2C$ codebook entries, which is solvable in $O(C^2)$ time under the greedy one-to-one rule. The Residual E-step is similar but is run C times (once per class) on the shared $2K_{\text{opt}}^{\max}$ pool, totaling $O(C \cdot (K_{\text{opt}}^{\max})^2)$. Both E-steps are dominated by the network update at typical scales (e.g., $|\mathcal{D}| = 10^4$, $C = 20$, $K_{\text{opt}} = 25$). At inference time, only a single forward pass and a Hamming distance comparison are required, and the complexity matches that of a standard hash-center method.

Convergence behavior. The alternating E/M-steps follow the same structure as the codebook-centric optimization of CRH (Yin et al., 2026), for which empirical convergence has been established. We do not claim global optimality. The random visiting order in the greedy one-to-one rule prevents systematic local-minimum bias toward any particular cluster. Empirically, the training mAP plateaus within the first 100–150 epochs across all dataset/bit configurations.

J Quantization Loss Sensitivity

Figure 5 shows the sensitivity to the quantization regularizer weight λ across all three datasets at 16/32/64 bits.

As λ increases, mAP@all remains stable across $\lambda \in [0.01, 0.5]$ on all three datasets. The effect on intra-class ρ varies across datasets, fluctuating on CIFAR-100, stabilizing on ImageNet-S20, and

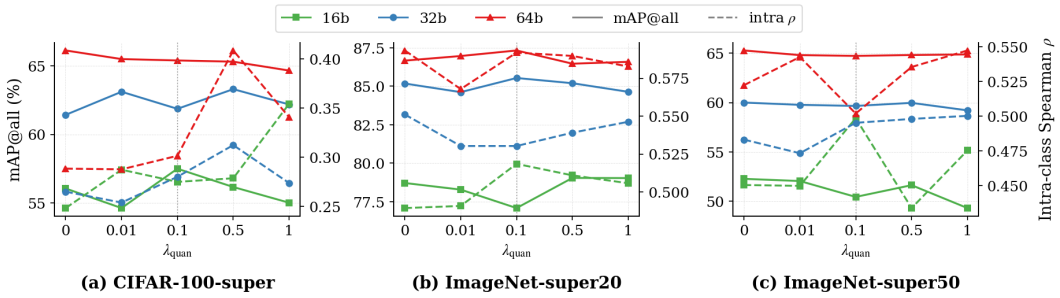


Figure 5: λ sensitivity. The left y-axis shows mAP@all (%, solid lines) on the N_{ctr} center bits and the right y-axis shows intra-class Spearman ρ (dashed lines) on the full code. Colors correspond to total 16b / 32b / 64b. The vertical dotted line marks $\lambda = 0.1$.

first decreasing and then increasing on ImageNet-S50. Overall, both metrics remain robust across a wide range of λ, and we select $\lambda = 0.1$ as the default following Yin et al. (2026).

K Fine-Grained Superclasses and the Residual Weight α

We analyze the effect of the residual weight α in $\mathcal{L} = \mathcal{L}_{\text{center}} + \alpha\mathcal{L}_{\text{res}}$ to assess whether the residual loss consistently improves inter-class performance across settings. In addition to CIFAR-100, where the 20 superclasses are visually distinct, we evaluate on CUB-super37, a fine-grained dataset where all 37 superclasses are bird families with small inter-class visual differences.

Table 7: Results on CUB-super37 at 16/32/64 bits. Inter-class mAP compares baselines' full N-bit code against RICH's N_{ctr} center bits at matched bit width. Intra-class metrics compare all methods at the same total N. Negative ρ means that sub-cluster pairs closer in the pretrained embedding tend to be farther apart in the code; values near zero indicate no preserved ordering.

Method	Inter-class			Intra-class								
	mAP@all (%) ↑			Sub mAP@100 (%) ↑			SR@100 (%) ↑			intra ρ ↑		
	16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b
CSQ (Yuan et al., 2020)	83.17	85.48	**86.59**	24.10	23.93	24.17	69.96	69.68	70.58	0.089	0.107	0.178
OrthoHash (Hoe et al., 2021)	81.47	82.07	82.96	24.19	24.34	24.21	68.97	68.98	69.16	0.036	0.160	0.032
MDS (Wang et al., 2023)	77.45	**85.52**	83.33	24.38	24.19	24.25	69.27	70.91	70.69	0.120	−0.031	0.241
SHC (Chen et al., 2025)	81.95	82.77	85.61	23.93	23.74	24.70	68.53	68.77	69.83	0.129	0.073	−0.110
CRH (Yin et al., 2026)	**83.66**	83.89	83.56	23.99	23.66	23.84	68.54	69.16	69.30	0.120	0.081	0.194
RICH **(ours)**	80.90	82.47	82.71	**41.33**	**45.14**	**41.05**	**80.39**	**83.43**	**86.63**	**0.249**	**0.253**	**0.372**

Table 8: Residual weight α sensitivity on CIFAR-100 and CUB-super37. mAP@all is measured on the N_{ctr} center bits.

Dataset	α	Inter-class			Intra-class								
		mAP@all (%) ↑			Sub mAP@100 (%) ↑			SR@100 (%) ↑			intra ρ ↑		
		8b	16b	32b	16b	32b	64b	16b	32b	64b	16b	32b	64b
CIFAR-100	1.0	58.37	61.83	67.09	43.59	46.46	49.49	36.29	40.10	49.65	0.276	0.257	0.264
	0.5	57.71	63.82	65.07	42.32	46.84	47.15	30.09	38.96	45.86	0.272	0.289	0.407
	0.1	58.81	63.50	63.13	43.43	48.72	43.09	34.49	37.02	38.52	0.239	0.264	0.383
	0.01	57.74	57.81	62.91	44.06	42.13	51.09	28.22	30.38	46.71	0.202	0.291	0.215
CUB-super37	1.0	78.17	80.90	82.47	41.33	45.14	41.05	80.39	83.43	86.63	0.249	0.253	0.372
	0.5	79.32	82.10	83.92	42.01	46.32	44.91	80.68	83.02	86.71	0.365	0.187	0.312
	0.1	81.02	83.56	84.31	42.14	45.14	46.43	79.85	82.31	83.91	0.330	0.277	0.125
	0.01	81.57	83.03	83.48	42.87	40.62	39.91	80.01	83.63	86.68	0.128	0.208	0.132

Table 7 compares RICH (α=1) against the baselines on CUB-super37. RICH achieves the highest scores on all three intra-class metrics across all bit widths, but inter-class mAP falls behind the strongest baselines. Reducing α lessens the residual head's influence on the shared backbone, im-

proving inter-class mAP to the level of the top baselines (Table 8), at the cost of reduced intra-class performance.

Table 8 reports the α sensitivity on both datasets. On CIFAR-100, mAP@all remains stable or even improves as α increases, suggesting that the residual head provides a synergistic benefit to inter-class discrimination rather than competing with it. When α is reduced to 0.01, inter-class mAP and SR@100 degrade at 16 and 32 bits, as the residual head receives insufficient learning signal, although subclass mAP@100 does not. SR@100 is highest at α=1.0 at every bit width, while intra-class ρ shows no consistent trend. On CUB-super37, a clear trade-off emerges. As α decreases, inter-class mAP generally improves (e.g., from 82.47% to 84.31% at 64b), while intra-class ρ degrades at 64b and SR@100 stays within 3%p. At 64b, subclass mAP@100 peaks at α=0.1 (46.43% vs. 41.05% at α=1.0), where the inter-class improvement outweighs the intra-class loss. Reducing α further to 0.01 degrades subclass mAP@100 at 32 and 64 bits and intra-class ρ at 16 and 64 bits, for the same reason.

L Class Imbalance Robustness

We evaluate RICH on two CIFAR-100 variants with class imbalance (Grčić et al., 2024) to verify that the intra-class advantage observed on the balanced benchmarks is not an artifact of uniform class structure. CIFAR68 removes 32 of the 100 subclasses, creating subclass-count imbalance across superclasses. CIFAR-SI retains all 100 subclasses but varies per-subclass sample retention from 32% to 99%, creating sample-count imbalance.

Table 9: Results on CIFAR68 and CIFAR-SI at 16/32/64 bits. Inter-class mAP compares baselines' full N-bit code against RICH's N_{ctr} center bits at matched bit width. Intra-class metrics compare all methods at the same total N.

Dataset	Method	Inter-class			Intra-class								
		mAP@all (%) ↑			Sub mAP@100 (%) ↑			SR@100 (%) ↑			intra ρ ↑		
		16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b
CIFAR68	CSQ (Yuan et al., 2020)	**68.88**	69.16	69.20	27.78	27.61	27.06	16.84	19.39	23.57	0.042	0.070	0.126
	OrthoHash (Hoe et al., 2021)	62.25	63.16	63.43	26.61	27.82	29.69	13.57	18.45	22.96	0.101	0.166	0.113
	MDS (Wang et al., 2023)	68.19	**70.70**	**74.82**	27.57	28.29	31.28	13.25	15.44	18.72	0.144	0.122	0.145
	SHC (Chen et al., 2025)	64.92	66.38	72.43	26.44	29.64	29.97	14.38	18.20	17.11	0.140	0.129	0.129
	CRH (Yin et al., 2026)	65.45	66.07	67.74	27.19	26.39	27.55	14.69	15.50	15.75	0.107	0.155	0.168
	RICH **(ours)**	67.23	70.21	72.38	**51.56**	**54.52**	**55.94**	**41.50**	**47.26**	**54.34**	**0.300**	**0.366**	**0.279**
CIFAR-SI	CSQ (Yuan et al., 2020)	53.86	58.52	58.44	19.87	20.16	21.38	21.60	25.71	30.53	0.134	0.119	0.187
	OrthoHash (Hoe et al., 2021)	52.10	53.48	53.89	19.07	19.58	22.47	16.95	19.79	28.42	0.148	0.164	0.113
	MDS (Wang et al., 2023)	57.78	**62.40**	60.91	20.92	22.94	26.29	15.51	19.37	26.75	0.168	0.158	0.221
	SHC (Chen et al., 2025)	53.76	56.02	61.07	19.06	20.18	22.76	18.96	20.10	21.24	0.140	0.152	0.181
	CRH (Yin et al., 2026)	55.03	55.65	57.34	21.10	21.36	21.54	19.02	22.77	25.94	0.122	0.137	0.136
	RICH **(ours)**	**59.55**	59.23	**63.72**	**37.40**	**43.42**	**44.79**	**36.95**	**45.52**	**60.00**	**0.346**	**0.384**	**0.274**

Table 9 reports the results. On both datasets, RICH achieves the highest scores on all three intra-class metrics across all bit widths, with margins comparable to those on the balanced benchmarks (Table 1). On CIFAR68, where some subclasses are removed entirely, inter-class mAP falls behind the best baseline (CSQ at 16b, MDS at 32b and 64b) by 0.49–2.44%p, suggesting that RICH is more sensitive to subclass-count imbalance than single-center methods. RICH still leads CRH by $+1.78$ to $+4.64$%p across all three settings. On CIFAR-SI, where all subclasses are present and only the sample counts vary, inter-class mAP leads on two of three settings, matching the pattern of the balanced benchmarks.

M Seed Robustness

Table 10 reports the mean and standard deviation of mAP@all, subclass mAP@100, SR@100, and intra-class ρ over three random seeds for all datasets and bit-width configurations.

RICH shows consistent performance across all seeds. The standard deviation of mAP@all stays below 0.8%p across all settings. Subclass mAP@100 and SR@100 stay below 2%p in most settings, though ImageNet-S20 at 16 bits shows higher variance (± 2.12%p and ± 2.58%p respectively). Intra-class ρ stays below 0.05 across all settings, confirming that the reported results are not sensitive to random initialization.

Table 10: Seed robustness of RICH. Mean ± standard deviation over three runs.

Dataset	Inter-class		Intra-class			
	Bits	mAP@all (%)	Bits	Sub mAP@100 (%)	SR@100 (%)	intra ρ
CIFAR-100	8	57.90 ± 0.33	16	44.24 ± 0.99	36.83 ± 0.48	0.264 ± 0.025
	16	61.79 ± 0.06	32	46.28 ± 0.35	40.91 ± 0.68	0.309 ± 0.042
	32	66.59 ± 0.50	64	50.11 ± 0.88	50.60 ± 1.59	0.278 ± 0.011
ImageNet-S20	8	80.23 ± 0.64	16	55.91 ± 2.12	39.15 ± 2.58	0.510 ± 0.016
	16	84.46 ± 0.08	32	63.80 ± 1.65	49.71 ± 1.92	0.523 ± 0.029
	32	87.41 ± 0.37	64	65.69 ± 0.65	56.30 ± 0.46	0.600 ± 0.022
ImageNet-S50	8	51.92 ± 0.76	16	48.34 ± 1.10	39.27 ± 1.78	0.444 ± 0.020
	16	60.16 ± 0.34	32	52.50 ± 0.22	46.65 ± 0.18	0.497 ± 0.013
	32	64.60 ± 0.26	64	55.82 ± 0.40	54.90 ± 0.09	0.512 ± 0.017

Table 11: Comparison under shared sub-cluster supervision on CIFAR-100. All baselines are retrained with the same K_{opt}-means sub-cluster labels used by RICH. We reproduce RICH numbers from Table 1. Inter-class mAP compares baselines' full N-bit code against RICH's N_{ctr} center bits at matched bit width. Intra-class metrics compare all methods at the same total N.

Method	Inter-class			Intra-class								
	mAP@all (%) ↑			Sub mAP@100 (%) ↑			SR@100 (%) ↑			intra ρ ↑		
	16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b
CSQ	8.06	10.00	12.20	30.79	39.37	37.84	30.72	44.88	47.42	0.306	0.289	0.385
OrthoHash	8.09	10.10	11.36	35.40	44.43	46.78	34.39	46.49	49.33	0.287	0.291	0.355
MDS	16.96	16.87	22.25	14.40	25.30	35.45	13.97	23.32	35.85	0.310	0.313	0.268
SHC	9.91	10.36	11.15	35.87	43.89	46.50	36.21	46.46	48.82	0.236	0.303	0.345
CRH	20.57	23.54	28.69	34.73	**49.42**	48.78	31.80	**50.70**	**51.59**	**0.477**	**0.476**	**0.534**
RICH **(ours)**	**61.83**	**67.09**	**67.00**	**43.59**	46.46	**49.49**	**36.29**	40.10	49.65	0.276	0.257	0.264

N Effect of Sub-cluster Supervision

To isolate the contribution of RICH's dual-head architecture from the effect of sub-cluster supervision, we retrain all baselines with the same K_{opt}-means sub-cluster labels used by RICH on CIFAR-100. Each baseline replaces its original superclass labels with the finer sub-cluster labels, expanding the label space from C superclasses to $\sum_c K_{\text{opt},c}$ sub-clusters, and is otherwise kept with its original hyperparameters.

Table 11 shows that adding the same K_{opt}-means sub-cluster labels to single-head baselines collapses their inter-class mAP to 8–29%, far below the original Table 1 numbers (56–66%), while RICH retains 61.83–67.09%. On the intra-class side, several baselines match or even surpass RICH. At 32 bits, CRH achieves subclass mAP@100 of 49.42% and SR@100 of 50.70%, both above RICH (46.46% and 40.10%). This confirms that sub-cluster supervision alone can improve intra-class retrieval, but a single shared code cannot simultaneously encode class identity and subclass structure. The dual-head split is therefore essential because the center head preserves inter-class discrimination while the residual head captures intra-class detail without interference.

O Cascade Error Decomposition

Table 13 decomposes the two-stage retrieval errors. For each query we check whether every item in its top-100 (ranked by center bits) belongs to the correct superclass ("perfect") or not ("impure"). Center-only P@100 is the fraction of top-100 items that share the query's superclass when ranking by center bits alone. Full-code P@100 is the same fraction when ranking by the full code in a single pass. Subclass mAP@100 is reported separately for the perfect and impure query groups, with the overall average in the last column.

Three patterns emerge. (i) CIFAR-100 has almost no perfect queries (0, 0, 43 out of 924, 939, 944 evaluated). With 20 visually similar superclasses, the center code cannot fully isolate a single super-

Table 12: Superclass oracle: stage 1 replaced by ground-truth superclass filtering, stage 2 unchanged (lexicographic on the residual bits). It bounds two-stage retrieval rather than full-code single-pass, and the bound is not exact because the two configurations exclude different queries from the mAP@100 average (Appendix H).

Dataset	Oracle Sub mAP@100 (%) ↑		
	16b	32b	64b
CIFAR-100	51.26	55.46	60.34
ImageNet-S20	54.46	66.46	68.66
ImageNet-S50	50.84	58.36	63.27

Table 13: Cascade error decomposition of two-stage retrieval. "Perfect": all top-100 items share the query superclass after center-bit ranking. "Impure": at least one top-100 item belongs to a different superclass. n counts queries whose top-100 list contains at least one relevant subclass item. Subclass mAP@100 is the subclass-level metric used in Table 1.

		Superclass P@100 (%)		Perfect queries		Impure queries		
Dataset	Bits	center-only	full code	n	subclass mAP	n	subclass mAP	Overall
CIFAR-100	16 (8+8)	67.15	67.58	0	–	924	43.59	43.59
	32 (16+16)	69.31	71.38	0	–	939	46.46	46.46
	64 (32+32)	73.49	76.03	43	33.92	901	50.23	49.49
ImageNet-S20	16 (8+8)	86.04	83.38	407	65.66	1431	51.58	54.70
	32 (16+16)	88.83	88.95	669	74.95	1194	61.17	66.12
	64 (32+32)	89.73	90.17	999	73.49	876	54.95	64.83
ImageNet-S50	16 (8+8)	66.32	61.28	235	63.83	3936	47.69	48.60
	32 (16+16)	70.18	71.27	720	66.82	3651	49.48	52.34
	64 (32+32)	72.81	74.44	480	78.14	3936	52.60	55.37

class in the top 100. All subclass retrieval therefore operates under impure conditions. (ii) Perfect queries achieve substantially higher subclass mAP@100. On ImageNet-S20 and ImageNet-S50, the gap between perfect and impure groups ranges from 14 to 26 %p, confirming that stage-1 superclass accuracy directly governs subclass retrieval quality. (iii) Center-only and full-code P@100 are close in every setting (within 5.1 %p, and within 2.6 %p in seven of nine). The residual bits reorder items near the boundary but rarely change which superclass occupies the top 100. The interference instead appears in inter-class mAP@all (inter-class columns of Table 2), where we evaluate the full ranking.

Table 14: Superclass precision at retrieval depths k=100, 500, and 1,000. Center-only ranks by center bits alone. Full code ranks by the full N-bit code in a single pass.

		P@100 (%)		P@500 (%)		P@1,000 (%)	
Dataset	Bits	center-only	full code	center-only	full code	center-only	full code
CIFAR-100	16 (8+8)	67.15	67.58	67.13	59.47	67.00	52.64
	32 (16+16)	69.31	71.38	70.24	67.45	70.71	63.75
	64 (32+32)	73.49	76.03	74.56	72.93	74.90	70.88
ImageNet-S20	16 (8+8)	86.04	83.38	86.27	73.25	84.46	64.24
	32 (16+16)	88.83	88.95	89.24	86.50	89.11	83.41
	64 (32+32)	89.73	90.17	90.01	88.55	89.83	87.58
ImageNet-S50	16 (8+8)	66.32	61.28	66.35	47.66	63.80	39.12
	32 (16+16)	70.18	71.27	71.04	66.41	69.80	61.09
	64 (32+32)	72.81	74.44	73.13	70.49	73.02	68.03

Table 14 extends the comparison to deeper retrieval depths. At k=100 the two retrieval strategies achieve similar superclass precision (within 5.1 %p), consistent with observation (iii) above. At k=500 and k=1,000, however, full-code precision drops sharply while center-only precision remains stable. The gap is largest at 16 bits, reaching 14–25 %p at k=1,000, and narrows at higher bit widths as the center bits become more discriminative. The residual bits therefore progressively disrupt superclass ordering at deeper ranks, and two-stage retrieval avoids this degradation by preserving the center-bit ranking.

P SUB-CLUSTER EMBEDDING ANALYSIS

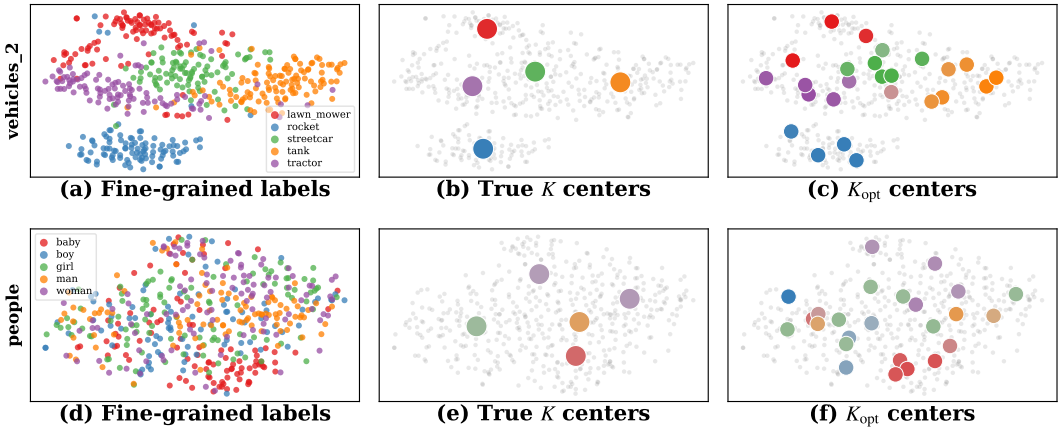


Figure 6: t-SNE of CIFAR-100 ResNet-34 pretrained features (each row: one superclass, N=500). **(a)** Ground-truth subclass labels. **(b)** K-means cluster centers at the true K (K=5). **(c)** K-means cluster centers at K_{opt} (K=25). The color of each circle indicates the dominant subclass in that cluster, and the saturation is proportional to the cluster's purity. **Top:** `vehicles_2` (lawn_mower / rocket / streetcar / tank / tractor). **Bottom:** `people` (baby / boy / girl / man / woman).

The premise of the K_{opt} design is that meaningful intra-class substructure already exists in the pretrained embedding space and that K-means can reliably convert it into a discrete supervisory signal. Figure 6 visualizes the t-SNE of CIFAR-100 ResNet-34 pretrained features for two contrasting superclasses. In `vehicles_2` (top), the subclasses (lawn mower, rocket, streetcar, tank, tractor) are clearly separated in the embedding space, and both the true-K and K_{opt} clusterings recover the actual subclass boundaries well. In contrast, for `people` (bottom), the subclasses (baby, boy, girl, man, woman) are not cleanly separated in the embedding space, and the true-K=5 clustering fails to recover the class boundaries. However, the finer clustering at K_{opt}=25 forms tighter sub-partitions aligned with the local density structure of the embedding space, providing the residual head with a richer supervisory signal of intra-class diversity than a single class center could provide.

Q VIT-B/16 BACKBONE EXPERIMENTS

Table 15 verifies that RICH generalizes beyond the ResNet-34 backbone used in the main experiments, repeating the evaluation with a ViT-B/16 backbone pretrained on ImageNet. All methods use the same ViT-B/16 feature extractor under identical training settings.

RICH consistently achieves the highest subclass mAP@100 and SR@100 across all three datasets and bit widths. The near-zero or negative intra-class ρ of all baselines is more pronounced than under ResNet-34. This suggests that ViT's larger representational capacity amplifies feature-space reorganization during fine-tuning, making intra-class preservation harder for every method. Under this harsher condition, the subclass mAP@100 and SR@100 gap between RICH and the baselines widens compared to the ResNet-34 results, confirming that the residual head remains effective even when the pretrained intra-class signal is more strongly disrupted. RICH also achieves the highest intra-class ρ on all three datasets and bit widths, although the absolute values are lower than under ResNet-34 (0.07–0.30) and the margin is smallest on ImageNet-S50 at 64 bits (0.073 vs. 0.070). At matched inter-class bit budget, RICH leads on ImageNet-S20 and ImageNet-S50 across all bit widths. On CIFAR-100, RICH leads at 16b and 32b but falls behind MDS by 3.26%p at 64b (80.54 vs. 83.80).

Table 15: Results with ViT-B/16 backbone on three datasets. Inter-class mAP compares baselines' full N-bit code against RICH's N_{ctr} center bits at matched bit width. Intra-class metrics compare all methods at the same total N. Negative ρ means that sub-cluster pairs closer in the pretrained embedding tend to be farther apart in the code; values near zero indicate no preserved ordering.

Dataset	Method	Inter-class			Intra-class								
		mAP@all (%) ↑			Sub mAP@100 (%) ↑			SR@100 (%) ↑			intra ρ ↑		
		16b	32b	64b	16b	32b	64b	16b	32b	64b	16b	32b	64b
CIFAR-100	CSQ	71.06	72.05	72.56	20.69	21.31	22.28	12.65	13.31	15.65	−0.065	−0.006	0.083
	OrthoHash	69.88	72.15	70.65	20.72	20.28	21.08	11.32	10.73	12.00	−0.006	0.069	0.092
	MDS	70.75	77.50	**83.80**	20.32	21.88	28.99	10.20	10.59	13.62	0.038	0.003	0.078
	SHC	71.89	73.35	77.81	20.91	21.20	23.20	10.17	11.01	12.22	−0.043	−0.011	0.025
	CRH	70.33	72.39	72.93	19.99	20.60	21.94	11.96	10.42	12.61	−0.038	−0.018	0.108
	RICH **(ours)**	**75.77**	**77.56**	80.54	**58.87**	**65.58**	**65.93**	**35.23**	**48.86**	**53.53**	**0.223**	**0.180**	**0.212**
ImageNet-S20	CSQ	88.00	87.30	87.62	13.86	14.13	16.25	6.90	7.46	11.01	−0.036	−0.071	0.005
	OrthoHash	86.59	89.52	87.69	13.72	13.63	13.79	6.76	6.52	8.28	−0.069	−0.096	0.018
	MDS	84.21	87.24	88.28	13.63	14.02	21.21	8.14	8.63	12.31	−0.023	0.032	0.047
	SHC	87.63	90.04	90.72	13.65	14.80	19.92	6.48	6.90	10.97	−0.070	−0.029	−0.001
	CRH	87.03	87.82	88.18	13.68	13.66	13.93	7.16	6.89	7.50	−0.061	−0.049	−0.072
	RICH **(ours)**	**92.20**	**91.89**	**91.95**	**81.04**	**83.33**	**83.88**	**42.55**	**48.86**	**53.20**	**0.299**	**0.243**	**0.154**
ImageNet-S50	CSQ	70.08	69.60	69.41	13.75	13.62	14.38	11.68	14.15	16.83	0.068	0.041	0.070
	OrthoHash	71.59	73.04	74.80	13.72	13.70	14.48	9.19	10.88	12.18	−0.029	−0.027	−0.008
	MDS	47.83	66.38	68.44	13.32	13.64	20.41	15.23	13.10	18.69	0.001	−0.009	0.046
	SHC	72.47	74.02	76.64	13.64	13.91	17.63	9.06	10.46	12.93	−0.014	−0.024	0.011
	CRH	71.60	72.07	73.71	13.69	13.82	14.21	9.36	11.24	11.98	0.003	−0.048	−0.030
	RICH **(ours)**	**74.44**	**75.54**	**79.35**	**75.17**	**77.42**	**78.77**	**42.39**	**47.68**	**53.14**	**0.109**	**0.126**	**0.073**

R LIMITATIONS

We extend the brief discussion in Section 5 with a more complete account of RICH's known limitations and the conditions under which they matter.

Pretrained-embedding dependence. The residual head's supervisory signal is generated by K-means clustering on a fixed pretrained backbone (ResNet-34 in our experiments; Appendix H). The quality of the resulting sub-cluster labels depends on whether the pretrained embedding already captures meaningful intra-class structure. Appendix P (Figure 6) shows that on CIFAR-100, this premise holds for some superclasses (e.g., `vehicles_2`) but is weaker for others (e.g., `people`) where the visual subclasses are not well-separated in the pretrained feature space. On domain-specific fine-grained datasets (medical imaging, satellite imagery, niche product catalogs), a generic ImageNet-pretrained backbone may fail to produce useful sub-cluster structure, in which case a domain-adapted or self-supervised backbone would likely be required.

Inter-class capacity and the residual weight. Splitting the code halves the bits available to the center head, so at N=16 the inter-class codebook has only 8 bits, and the default equal weighting (α=1) assumes the center head can still learn inter-class structure. Where coarse classes are visually similar the assumption weakens: on CUB-super37, 37 bird families, RICH trails the best baseline in inter-class mAP at every width (-2.8 to -3.9%p, Table 7), and on CIFAR68 by 0.5–2.4%p (Table 9), while leading every intra-class metric. Even on the primary benchmarks the narrowest lead, 0.16%p on CIFAR-100 at 16 bits, is where the center head has fewest bits. Reducing α recovers part of the gap (CUB-super37 64b: $82.47\% \rightarrow 83.92\%$ at α=0.5, Table 8) at the cost of the intra-class metrics, so α is the control for this trade-off (Appendix K).

Class-imbalance sensitivity. The K_{opt} algorithm (Appendix F) adapts $K_{\text{opt},c}$ to per-class sample counts, and CIFAR68 and CIFAR-SI confirm that the intra-class advantage survives moderate imbalance (Appendix L). Behavior under heavy long-tailed imbalance has not been validated.

Scope of evaluation. The empirical evaluation covers single-label coarse-superclass image retrieval on CIFAR-100, ImageNet-S20, ImageNet-S50, CIFAR68, CIFAR-SI, and CUB-super37. Multi-label retrieval, video retrieval, and cross-modal retrieval are out of scope.

Deeper hierarchies. RICH is a two-tier design (center head, residual head). A third tier would add a sub-sub-cluster head, and Proposition 2 gives the cost. Each tier needs $d \geq \lceil \log_2 M \rceil$ bits per sub-head for its codebook size M, so a 24-bit code split three ways leaves 8 bits per tier and caps every

tier's codebook at 2^8 codes before any redundancy for Hamming separation. The supervisory signal for a third tier also has to exist in the pretrained embedding, which Appendix P shows is not the case even for two tiers on some superclasses. The K_{opt} fallback (Appendix F) is the same safeguard a third tier would need. We therefore leave hierarchies beyond two tiers, at $N \geq 96$, to future work.

Backbone scope and cross-dataset transfer. Our main experiments use ResNet-34 for fair comparison with the baselines, which were evaluated on the same backbone in recent work (Chen et al., 2025; Yin et al., 2026), and an extension to ViT-B/16 is reported in Appendix Q. Because a hash-center codebook is bound to the training class set, we do not evaluate transfer across datasets with different class sets here, as in the baseline papers. The backbone change is the generalization test we report. Modern backbones such as DINOv2 and domain-specific pretrained models remain unexplored and may produce embeddings with different intra-class structure characteristics.

S Broader Impact

RICH improves intra-class retrieval quality in supervised image hashing without altering the standard supervised hashing setup (coarse labels, public benchmarks, a public pretrained backbone). We discuss the relevant societal impacts here.

Positive impact. Improved intra-class retrieval enables image-search systems to return semantically more coherent results. When a user queries an image, the returned set is more likely to share fine-grained semantic content with the query rather than merely sharing a coarse label. Concrete benefits include better organization of large public image archives (e.g., scientific specimen databases, historical-photograph archives), improved relevance in large-scale image retrieval systems, and more useful results in accessibility tools that retrieve visually similar examples for users with low vision. Because the residual head is parameter-light (a single linear layer on top of a shared backbone), these benefits do not come with substantial extra compute, training time, or carbon footprint.

Potential negative impact. The risks of more accurate image retrieval are not specific to RICH and apply to image retrieval as a category. Improved retrieval quality can amplify privacy concerns when applied to surveillance imagery or scraped personal photos, and can make face-matching or person-tracking systems more capable. RICH does not introduce a new attack vector, nor does it increase the data or annotations a system would need to enable such misuse, and the contribution is foundational research rather than an application-specific system.

Mitigation. The standard mitigations for image-retrieval misuse apply: dataset curation (avoid training on scraped personal data without consent), deployment review (do not deploy in surveillance contexts without legal authorization), and access controls on production systems. None of these requires RICH-specific intervention. The proposed method requires no safeguards beyond those already accepted in the broader image-retrieval community.

Failure modes. When the pretrained backbone's embedding does not separate intra-class variants (Appendix R), RICH's sub-cluster labels can become noisy, and the residual bits encode less meaningful structure. On fine-grained domains where coarse classes are visually similar (e.g., CUB-super37), the center head may not converge sufficiently, degrading inter-class retrieval. The residual weight α can mitigate this trade-off (Appendix K), but users should verify both sub-cluster quality and center-head convergence on their domain before relying on the intra-class ordering for downstream decisions.