Characteristics Matching Based Hash Codes Generation for Efficient Fine-grained Image Retrieval

Zhen-Duo Chen, Li-Jun Zhao, Zi-Chao Zhang, Xin Luo, Xin-Shun Xu*
School of Software, Shandong University, China
chenzd.sdu@gmail.com, 202215247@mail.sdu.edu.cn, zhangzichao1008@163.com
luoxin.lxin@gmail.com, xuxinshun@sdu.edu.cn

Abstract

The rapidly growing scale of data in practice poses demands on the efficiency of retrieval models. However, for fine-grained image retrieval task, there are inherent contradictions in the design of hashing based efficient models. Firstly, the limited information embedding capacity of low-dimensional binary hash codes, coupled with the detailed information required to describe fine-grained categories, results in a contradiction in feature learning. Secondly, there is also a contradiction between the complexity of fine-grained feature extraction models and retrieval efficiency. To address these issues, in this paper, we propose the characteristics matching based hash codes generation method. Coupled with the cross-layer semantic information transfer module and the multi-region feature embedding module, the proposed method can generate hash codes that effectively capture fine-grained differences among samples while ensuring efficient inference. Extensive experiments on widely used datasets demonstrate that our method can significantly outperform state-of-the-art methods.

1. Introduction

Recently, there has been a growing interest in efficient Fine-Grained Image Retrieval (FGIR) due to the ever-increasing data scale in practical applications. As a representative solution to efficient retrieval, hashing based model has been introduced for this task. Different from common coarse-grained datasets that most traditional hashing methods are designed for, the primary feature of fine-grained datasets is that categories involved in one fine-grained dataset usually belongs to the same meta-category (like bird or aircraft) [22]. It results in small inter-category difference and relatively large intra-category variance (because of pose and angle). As a result, how to fully discover and extract subtle fine-grained features and learn discriminative hash codes in an effective and efficient approach becomes the emphasis and difficulty in the design of fine-grained hashing methods.

In the past several years, a series of fine-grained hashing methods have been published and achieved significant progress on the task of efficient FGIR. Motivated by the aforementioned difficulty in discriminative fine-grained hash codes generation, the main attention of researchers has been paid to the design of feature extraction modules. They sufficiently draw lessons from the basic ideas of well-developed fine-grained feature learning or classification methods, and attempt to concatenate feature vectors extracted from multiple image regions and network layers for generating more discriminative hash codes.

However, there are inherent contradictions between the fine-grained feature learning and hashing based retrieval, which has been ignored by nearly all these methods. Firstly, it is the **contradiction in feature learning**. Concretely, integrating comprehensive information is critical for describing a fine-grained object, but the length of target hash codes is usually extremely low and fixed. Therefore, limited by the information embedding capability of low-dimensional binary codes, better feature extraction models and more high-dimensional fine-grained feature vectors cannot always achieve corresponding performance gains because of the information loss during the mapping from feature vectors to binary codes. Furthermore, there is also a **contradiction between effectiveness and efficiency**. A comprehensive fine-grained feature extraction module usually leads to more extra parameters and computational costs, which is detrimental to the primary target of hashing based retrieval, i.e., efficient inference and retrieval.

As a result, although fully learning discriminative fine-grained information is critical for hashing based FGIR tasks, it is obvious that the traditional idea, i.e., simply mapping the feature vectors extracted by a complex model to hash codes, is not the optimal choice to handle this issue. Furthermore, if a fine-grained hashing method is considered as a whole rather than a simple combination of feature learning and hash code learning modules, it can be found

*Corresponding author

that the ultimate goal of fine-grained hashing methods is employing low-dimensional binary codes to represent fine-grained differences among samples, rather than providing a detailed description of the input samples. This implies that there is no necessity to entirely embed the extracted fine-grained features into the hash codes, which is also impossible in fact. Instead, these fine-grained feature vectors can be abstracted into descriptions of critical characteristics for discriminating fine-grained categories. And then these characteristics can be stored as prior knowledge. As a result, the degree of matching between samples and these critical characteristic vectors can be mapped into hash codes.

Overall speaking, compared to the traditional process of concatenating feature vectors and then mapping them into hash codes, this characteristics matching based hash code generation strategy offers two significant advantages. 1) Firstly, this approach shifts the focus from concrete sample descriptions to the abstract level of matching with specific characteristics. It greatly simplifies the information that needs to be embedded in the hash codes while ensuring the descriptive power for significant differences among samples. This effectively alleviates the aforementioned contradiction in feature learning. 2) Additionally, through the training phase, this process can unify and abstractly represent the critical characteristics of each category, and preserve the relationships between the overall sample features and these characteristic vectors. Consequently, the extraction of fine-grained details for multiple regions and layers during testing can be discarded. This maximizes testing efficiency and mitigates the contradiction between efficiency and effectiveness. To a certain extent, this approach also aligns better with human habit when swiftly determining the similarity among multiple samples. It involves searching for whether each sample possesses key characteristics from one's own experience and memory, rather than meticulously checking every detail, which is usually more suitable for accurate retrieval where efficiency is not the concern.

Inspired by the above discussions, in this paper, we propose a novel efficient FGIR method according to aforementioned Characteristics Matching Based Hash codes generation strategy, namely CMBH. In addition, similar to the process of human learning and experience accumulation, two auxiliary modules that operate only during the training phase are designed to ensure that the critical characteristic vectors can effectively describe and represent every fine-grained category. Specifically, a cross-layer semantic information transfer module is designed to embed fine-grained information learned across a series of network layers to the characteristic vectors. Besides, a multi-region feature learning module is designed to associate the characteristic vectors with specific local details and preserve the distinctiveness among characteristic vectors.

In conclusion, the contributions of this paper can be summarized as follows:

- In this paper, we are the first to analyze the feature learning contradiction and efficiency contradiction in the task of efficient FGIR, and propose a Characteristics Matching Based Hashing method to address these issues.
- A cross-layer semantic information transfer module and a multi-region feature learning module is designed to comprehensively learn and embed fine-grained information for characteristic vectors optimizing, with nearly no extra parameters and computational costs introduced during inference and testing.
- Comprehensive experimental results demonstrate that the proposed method can significantly outperform SOTA methods in both effectiveness and efficiency.

2. Related Work

Fine-grained image retrieval is a crucial part of fine-grained image analysis and has attracted increasing attention recently. Early works like SCDA [20], CRL [29], DGCRL [30], MPFE [2], and KAE-Net [16] generally pay their main attention to achieving better retrieval accuracy with long real-valued vectors. In the last three years, the main research stream of this topic has turned to hashing based fine-grained image retrieval to seek a balance between accuracy and efficiency for dealing with increasing data size.

Hashing based retrieval is a representative strategy for Approximate Nearest Neighbor Search, and a well-developed field with many valuable works published [10, 17]. It proposes to map feature vectors into binary hash codes to achieve fast query speed and low storage costs. FPH [26] and DSaH [11] should be the earliest methods to introduce hashing into fine-grained image retrieval. Specifically, FPH proposes a two-pyramid hashing architecture to capture subtle differences from multiple network layers; DSaH designs an extra attention network to discover salient image regions. Both of them pay their main attention to extracting more comprehensive fine-grained feature vectors, and this basic idea is also followed by methods proposed after. For example, ExchNet [4] first localizes multiple regions with an attention mechanism, and then aligns the extracted part feature vectors across different images by an exchanging operation. A^2-Net [21] adopts a similar localization module and then tries to learn high-level attribute vectors for hash codes generation; and its improved version A^2-Net^{++} [23] further enhances model's self-consistency. As for the recently proposed methods, except for DLTH [12] which designs a novel list-wise triplet loss function to capture relative similarity, the rest also focuses on optimizing fine-grained feature extraction for hash code learning. sRLH [24] proposes a sub-region localization module to locate the peaks of non-overlap sub-regions for diverse local information capturing. Based on similar targets, FISH [3] and FCAENet [28] employ an attention-based erasing strategy. In addition, FISH further introduces a feature refinement module, and FCAENet adds an enhancing space relation loss. SEMICON [18] proposes a suppression-enhancing mask to explore the relation between discovered regions. Thereafter, CNET [27] designs a cascaded network and an attention-guided data augmentation strategy; and it also introduces a novel approach to balance the loss of multi-task. Finally, the most recently published AGMH [13] proposes an attention dispersion loss and a step-wise interactive external attention module to group and embed the category-specific visual attributes in multiple descriptors for comprehensive feature representation.

3. Methodology

3.1. Framework and Notations

The overall framework of the proposed method is illustrated in Fig. 1. Specifically, the training phase of the method consists of a main component, i.e., the characteristics matching based hash codes generation and training module, which also includes a backbone network for feature extraction. It is designed to calculate the matching score between instance feature vectors and Characteristic vectors (C-vectors), to generate binary codes accordingly. In addition, there are two auxiliary modules, i.e., the cross-layer semantic information transfer module and multi-region feature learning module. These modules integrate information extracted from different network layers and multiple image regions to train and optimize the C-vectors, and further ensure that they can well represent each fine-grained category. After training, only the main hash codes generation component is kept for the inference.

In the following sections, a raw pixel input image is represented as $\mathbf{I}_i \in \{\mathbf{I}_1, \dots, \mathbf{I}_n\}$, where n is the total number of training instances. The backbone network is represented as $N(\cdot)$ and the feature extraction procedure is simplified as

$$\{\mathbf{F}_i^j | j \in \{1, \dots, L\}\} = N(\mathbf{I}_a), \tag{1}$$

where feature map $\mathbf{F}_i^j \in \mathbb{R}^{c_j \times w_j \times h_j}$ is the output of the j-th stage (top to bottom) of the backbone ResNet [6]. For following processing, feature map $\mathbf{F}_i^j$ is further transformed into vector as

$$\mathbf{f}_i^j = N_{sub-j}(\mathbf{F}_i^j) \in \mathbb{R}^d, \tag{2}$$

where each $N_{sub-j}(\cdot)$ is a sub-network that consists of two convolutional layers, a global max pooling layer, and a fully connected layer. As a supervised method, the target of CMBH is learning to transform instance $\mathbf{I}_i$ into b-bit binary codes $\mathbf{b}_i \in \{-1, 1\}^b$ with the assistant of one-hot label vector $\mathbf{y}_i \in \{0, 1\}^y$, where y is the number of categories. In addition, the C-vectors are predefined as $\mathbf{C} \in \mathbb{R}^{y \times k \times d}$, in which k is the number of critical characteristic preset for each category. $\mathbf{C}$ can be randomly initialized and will be optimized as introduced in the following sections.

3.2. C-vectors Matching and Hash Codes Learning

As aforementioned, the key idea of this paper is adopting matching score between C-vectors and one output image feature vector $\mathbf{f}_i^l \in \{\mathbf{f}_i^1, \dots, \mathbf{f}_i^L\}$ to compress information to be embedded into hash codes. Given the symbols defined above, the matching score for input instance $\mathbf{I}_i$ is defined as $\mathbf{M}_i \in \mathbb{R}^{y \times k}$, and each element of $\mathbf{M}_i$ is calculated as

$$\mathbf{M}_{i,(a,b)} = \Big(\Big(\frac{\mathbf{C}_{a,b,*}}{||\mathbf{C}_{a,b,*}||_2}\Big)^{\top} \frac{\mathbf{f}_i^l}{||\mathbf{f}_i^l||_2}\Big), \tag{3}$$

where $||\cdot||_2$ represents the L2 normalization. Thereafter, the to-be-learned hash code $\mathbf{b}_i$ can be obtained by a mapping layer $N_{hash}(\cdot)$, which is formulated as

$$\mathbf{b}_i = N_{hash}(\mathbf{M}_i) = \text{sign}(\mathbf{W}_h(\text{vec}(\mathbf{M}_i)), \tag{4}$$

in which $\text{sign}(\cdot)$ is an element-wise sign function, $\mathbf{W}_h \in \mathbb{R}^{b \times (yk)}$ is the hash codes mapping matrix, and $\text{vec}(\cdot)$ is a simple function to reshape the input matrices to vectors.

As for hash codes learning, since designing the loss function is not the focus of this paper, we design a very simple loss function to combine pair-wise and proxy-based similarity preservation training, i.e.,

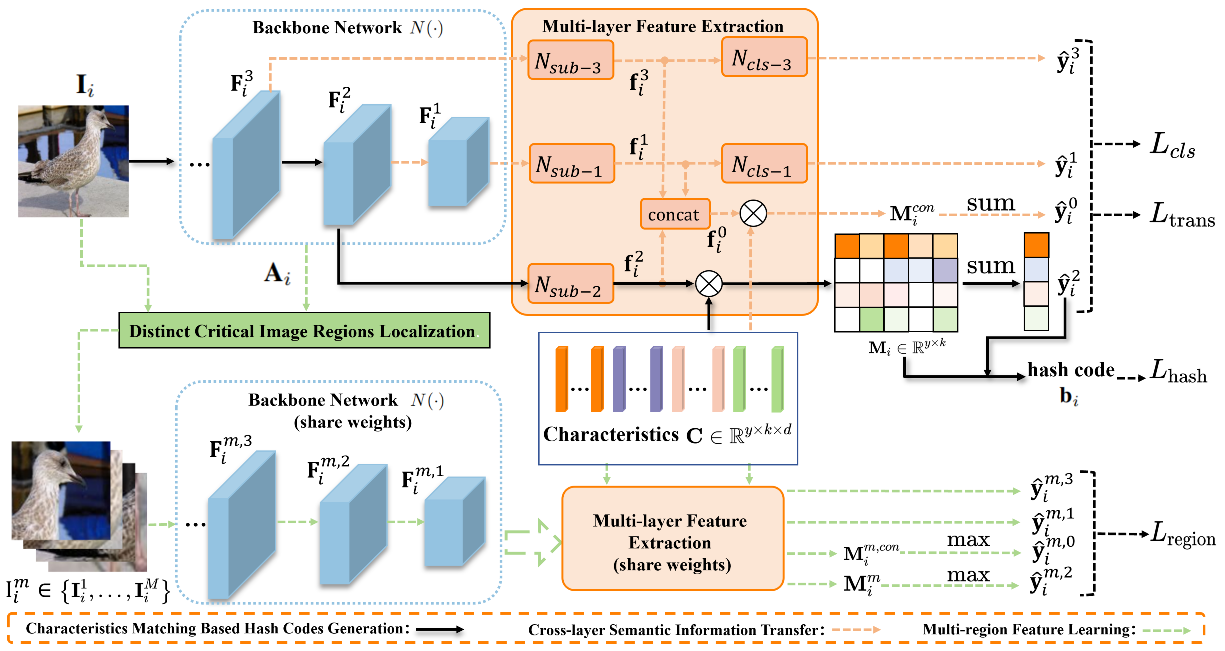
$$L_{hash} = L_{hash-pair} + L_{hash-proxy}. \tag{5}$$

Specifically, for the t-th training epoch, the loss functions are defined as

$$L_{hash-pair} = \frac{1}{nn}\sum_{i=1}^{n}\sum_{j=1}^{n}(\mathbf{u}_i^{(t)\top}\mathbf{b}_j^{(t-1)} - b\mathbf{S}_{i,j}^p)^2, \tag{6}$$

$$L_{hash-proxy} = \frac{1}{ny}\sum_{i=1}^{n}\sum_{j=1}^{y}(\mathbf{u}_i^{(t)\top}\mathbf{c}_j^{(t-1)} - b\mathbf{S}_{i,j}^c)^2, \tag{7}$$

where $\mathbf{b}_j^{(t-1)}$ is the recorded codes of j-th instance generated in the last training epoch, $\mathbf{u}_i^{(t)}$ is the relaxed version of $\mathbf{b}_i^{(t)}$ obtained by replacing the sign operation with tanh in Eq. (4), and $\mathbf{c}_j^{(t-1)}$ is the binary version of the average of $\mathbf{u}^{(t-1)}$ that belongs to the j-th category. During the first training epoch, both $\mathbf{c}_j^{(t-1)}$ and $\mathbf{b}_j^{(t-1)}$ can be randomly initialized. Besides, $\mathbf{S}^p$ is first defined as $\{1, 0\}^{n \times n}$, where $\mathbf{S}_{i,j}^p = 1$ if image i and j belong to the same category, 0 otherwise. And then every 0 in $\mathbf{S}^p$ is replaced with $-(\sum \mathbf{S}^p / \sum(1 - \mathbf{S}^p))$. Similarly, $\mathbf{S}^c$ is first defined as $\{1, 0\}^{n \times y}$, where $\mathbf{S}_{i,j}^c = 1$ if image i belong to the $j-$th category, 0 otherwise. And then every 0 in $\mathbf{S}^c$ is replaced with $-(\sum \mathbf{S}^c / \sum(1 - \mathbf{S}^c))$.



Backbone Network $N(\cdot)$
Multi-layer Feature Extraction
$\mathbf{I}_i$
$\mathbf{F}_i^3$
$\mathbf{F}_i^2$
$\mathbf{F}_i^1$
N_{sub-3}
N_{cls-3}
N_{sub-1}
N_{cls-1}
N_{sub-2}
$\mathbf{f}_i^3$
$\mathbf{f}_i^1$
$\mathbf{f}_i^2$
$\mathbf{f}_i^0$
concat
$\mathbf{M}_i^{con}$
sum
$\hat{\mathbf{y}}_i^3$
$\hat{\mathbf{y}}_i^1$
$\hat{\mathbf{y}}_i^0$
$\hat{\mathbf{y}}_i^2$
L_{cls}
L_{trans}
$\mathbf{M}_i \in \mathbb{R}^{y\times k}$
hash code $\mathbf{b}_i$
L_{hash}
$\mathbf{A}_i$
Distinct Critical Image Regions Localization
Characteristics $\mathbf{C} \in \mathbb{R}^{y\times k\times d}$
Backbone Network $N(\cdot)$ (share weights)
$\mathbf{F}_i^{m,3}$
$\mathbf{F}_i^{m,2}$
$\mathbf{F}_i^{m,1}$
$\mathbf{I}_i^m \in \{\mathbf{I}_i^1, \ldots, \mathbf{I}_i^M\}$
Multi-layer Feature Extraction (share weights)
$\mathbf{M}_i^{m,con}$
$\mathbf{M}_i^m$
max
$\hat{\mathbf{y}}_i^{m,3}$
$\hat{\mathbf{y}}_i^{m,1}$
$\hat{\mathbf{y}}_i^{m,0}$
$\hat{\mathbf{y}}_i^{m,2}$
L_{region}
Characteristics Matching Based Hash Codes Generation:
Cross-layer Semantic Information Transfer:
Multi-region Feature Learning:

Figure 1. The overall architecture of the proposed CMBH method. Operation represented by dashed lines are only used for training.

3.3. C-vectors Training and Optimization

In the previous section, the basic framework of characteristics matching based hash codes generation has been established. However, the prerequisite for this framework to work effectively is that the C-vectors can sufficiently describe fine-grained categories, thereby ensuring that using the matching scores as sample representations can adequately reflect the differences between samples. In this section, we will introduce how to train and optimize the C-vector through cross-layer and multi-region learning. Related modules are only utilized during the training phase, with the introduction of nearly no additional computational costs and parameters during the inference phase.

3.3.1 Cross-layer semantic information transfer

Integrating information of different granularities from multiple network layers is an effective approach to enhance the description of feature vectors for fine-grained class characteristics [5, 14, 25]. However, traditional cascading strategies inevitably introduce more parameters and computational costs, and result in higher-dimensional vectors, which is not a reasonable solution for hashing as discussed in Sec. 1. In this method, to ensure efficiency, only the output of a single network layer, i.e., $\mathbf{f}_i^l$, is ultimately involved in hash code generation. Information integrated from multiple network layers will be embedded into C-vectors, serving as an intermediary to transfer integrated information into the hash code generation process. Given feature vectors $\{\mathbf{f}_i^j | j \in \{1, \ldots, L\}\}$ generated in Eq. (1), the information transfer procedure includes two components.

The first component is based on the early fusion strategy. A unified cross-layer feature representation $\mathbf{f}_i^0$ can be obtained by concatenating feature maps from the different layers as follows,

$$\mathbf{f}_i^0 = N_{con}(\text{concat}(\mathbf{f}_i^1; \ldots; \mathbf{f}_i^L)) \in \mathbb{R}^d, \tag{8}$$

where N_{con} is a mapping layer including a fully-connected layer to unify feature dimension. $\mathbf{f}_i^0$ is also used to calculate a matching score matrix as

$$\mathbf{M}_{i,(a,b)}^{con} = \left(\left(\frac{\mathbf{C}_{a,b,*}}{||\mathbf{C}_{a,b,*}||_2}\right)^\top \frac{\mathbf{f}_i^0}{||\mathbf{f}_i^0||_2}\right). \tag{9}$$

By jointly optimizing Eq. (3) and (9), integrated information can be embedded and C-vectors can server as an intermediary for information transfer. Since $\mathbf{M}_i^{con}$ is not involved in hash codes learning, for training, classification losses are adopted to ensure that these matching score matrices are also consistent in semantic level:

$$L_{cls} = \sum_{i=1}^{n} \sum_{j=0}^{L} \text{CE}(\hat{\mathbf{y}}_i^j, \mathbf{y}_i), \tag{10}$$

where CE is a standard softmax-cross-entropy loss. In addition, $\hat{\mathbf{y}}_i^j$ is defined as the class vector generated from different layers. Specifically, $\hat{\mathbf{y}}_i^0 = \alpha \sum_{j=1}^{k} \mathbf{M}_{i,(*,j)}^{con}$ and $\hat{\mathbf{y}}_i^l = \alpha \sum_{j=1}^{k} \mathbf{M}_{i,(*,j)}$, where α is a hyper-parameter bigger

Algorithm 1 Distinct critical image regions localization.

```
Input: Feature map matrix $\mathbf{A}_i$, bounding box set $B = \emptyset$.
Output: Bounding box set $B$.
 1: for $m$ = 1 to $M$ do
 2:     $a \leftarrow$ The maximum value in $\mathbf{A}_i$;
 3:     $\mathbf{H}_i \leftarrow$ Values greater than $a \cdot (0.9 - m \cdot 0.05)$ in $\mathbf{A}_i$ are 1, and other values are 0;
 4:     $\bar{\mathbf{H}}_i \leftarrow$ Calculate the largest connected component of $\mathbf{H}_i$;
 5:     $[x_1, y_1, x_2, y_2] \leftarrow$ Calculate the bounding box coordinates of $\bar{\mathbf{H}}_i$;
 6:     $[\hat{x}_1, \hat{y}_1, \hat{x}_2, \hat{y}_2] \leftarrow$ Calculate the original image coordinates corresponding to $[x_1, y_1, x_2, y_2]$;
 7:     $\mathbf{A}_i \leftarrow \mathbf{A}_i \cdot (1 - \mathbf{H}_i) + (\mathbf{H}_i \cdot a - \mathbf{H}_i \cdot \mathbf{A}_i)$;
 8:     $B \leftarrow B \cup \{[\hat{x}_1, \hat{y}_1, \hat{x}_2, \hat{y}_2]\}$;
 9: end for
10: return Bounding box set $B$.
```

than 1 to magnify the gradient and accelerate the training procedure. Besides, $\hat{\mathbf{y}}_i^j = N_{cls-j}(\mathbf{f}_i^j)$ where $j \in \{1, ..., L\}$ and $j \neq l$, $N_{cls-j}(\cdot)$ is a one layer classification layer. These extra classification layers and losses are adopted to improve feature learning and following procedure.

The second component is based on the later fusion strategy. Given above classification vectors, the ensemble of classification results from multiple layers is generally better than any one of them. Inspired by the idea of Knowledge Distillation [7], the following cross-layer information transfer loss is designed to supervise the training of the output layer l with the integration of results from multiple layers,

$$L_{trans} = -\sum_{i=1}^{n}\Big(\text{softmax}(\sum_{j=0}^{L}\hat{\mathbf{y}}_i^j)^{\mathsf{T}}\log(\text{softmax}(\hat{\mathbf{y}}_i^l))\Big). \tag{11}$$

The training based on the aforementioned loss functions effectively ensures the semantic-level consistency among C-vectors. Given that $\mathbf{M}_i$ remains a high-dimensional vector, to further reduce the potential loss of critical information during the mapping process, especially for shorter hash codes, the semantic-preserving information obtained in this section can be further leveraged to reinforce the corresponding weights of C-vectors in hash code mapping. Specifically, Eq. (4) is modified as

$$\mathbf{b}_i = \text{sign}\Big(\mathbf{W}_h(\text{vec}(\frac{\exp(\hat{\mathbf{y}}_i^l)}{\sum_{j=1}^{y}\exp(\hat{\mathbf{y}}_{i,j}^l)} \odot \mathbf{M}_i))\Big), \tag{12}$$

where $\odot$ stands for element-wise multiplication with broadcasting operation.

3.3.2 Multi-region feature embedding

The ideal C-vectors should be able to sufficiently describe the distinct characteristics of a fine-grained category. Thus, another objective of the training and optimization process is to ensure distinctiveness among multiple C-vectors describing the same category. Typically, this objective can be achieved through an orthogonal constraint term. However, additional constraints inevitably affect the overall training efficiency of the model and may not accurately reflect the relationships among features. Multi-region feature extraction is an effective approach to fine-grained feature learning. In this section, we aim to achieve multi-region information embedding by extracting features from different regions of the input images and associating them with specific C-vectors, thereby ensuring distinctiveness among the C-vectors and improving feature learning.

Although there are many existing techniques or models for multi-region feature extraction, the introduction of additional trainable modules and losses inevitably affects model training and, consequently, may not be conducive to the assessment of the effectiveness of the idea presented in this paper. Therefore, an independent parameter-free algorithm is designed in this section. Based on the idea of SCDA [20] and NMS, it iteratively localizes multiple informative regions. Specifically, all feature maps generated in Eq. (1) are first resized to the same spatial size as the last stage feature map, i.e., $\{\overline{\mathbf{F}}_i^j \in \mathbb{R}^{c_j \times w_L \times h_L} | j \in \{1, \ldots, L\}\}$. Then the algorithm takes the channel-wise mean feature map $\mathbf{A}_i \in \mathbb{R}^{w_L \times h_L}$ as input, where

$$\mathbf{A}_i = \frac{1}{\sum_{j=1}^{L} c_j}\sum_{a=1}^{\sum_{j=1}^{L} c_j}\text{stack}(\overline{\mathbf{F}}_i^1; \ldots; \mathbf{F}_i^L)_{(a,*,*)}, \tag{13}$$

where $\text{stack}(\cdot)$ is a channel-wise stack operation. The bounding box coordinates of M distinct informative regions of input image can be generated according to Algorithm 1. Accordingly, M patches $\{\mathbf{I}_i^1, ..., \mathbf{I}_i^M\}$ are cropped out from image $\mathbf{I}_i$, and then processed in the same way as image $\mathbf{I}_i$ to generate corresponding feature vectors $\{\mathbf{f}_i^{m,j} | j \in \{1, \ldots, L\}, m \in \{1, \ldots, M\}\}$ with Eq. (1)-(2), as well as matching score $\mathbf{M}_i^m$ and $\mathbf{M}_i^{m,con}$ with Eq. (3), (8) and (9).

Thereafter, the max function will replace the corresponding sum function related to Eq. (10) to compute the corresponding class vector, i.e., $\hat{\mathbf{y}}_i^{m,0} = \alpha \max(\mathbf{M}_i^{m,con})$ and $\hat{\mathbf{y}}_i^{m,l} = \alpha \max(\mathbf{M}_i^m)$, where $\max(\cdot)$ is a row-wise max operation. During training, this ensures that feature vectors extracted from each patch are only related to its most similar C-vector, thereby promoting the gradual emergence of distinctiveness among C-vectors. Thereafter, by continuing to follow the same procedure introduced by Eq. (10) and (11), the loss function associated with the multi-region feature embedding discussed in this section can be obtained as

Datasets	Category	Training	Testing
CUB-200-2011 [19]	200	5,994	5,794
FGVC-Aircrafts [15]	100	6,667	3,333
Food101 [1]	101	75,750	25,250
NABirds [8]	555	23,929	24,633
VegFru [9]	292	29,200	116,931

Table 1. Statistics of experimental datasets.

$$L_{region} = \sum_{j=1}^{M} (L^j_{cls} + L^j_{trans}). \quad (14)$$

3.4. Training and Inference

The whole method can be trained end-to-end with batch based stochastic gradient descent and back-propagation algorithm, supervised by the following overall loss function:

$$L = L_{hash} + L_{cls} + L_{trans} + L_{region}, \quad (15)$$

in which each term has been introduced in Eq. (5), (10), (11) and (14) separately. After training, given a query image $\mathbf{I}_q$, the corresponding hash code can be obtained according to Eq. (1) - (3) and Eq. (12). It can be noticed that there is no cross-layer and multi-region feature extraction involved in the inference and test procedure, which promises the retrieval efficiency of the proposed method during testing.

4. Experiments

4.1. Experimental settings and Baselines

The proposed method is evaluated on five widely-used fine-grained datasets. The official training/testing data splitting is adopted, and details are summarized in Tab. 1. For fair comparison, ResNet-50 and ResNet-18 without the final pooling and classification layer are chosen as the backbone network. For hyper-parameters, the last three stages of the backbone are used, i.e., $L = 3$ in Eq. (1). In view of both effectiveness and efficiency, only the penultimate stage, i.e., $l = 2$, is used for hash codes generation, and the number of C-vectors for each category $k = 2$. The number of cropped patches M is empirically set to be 4 for comprehensively extracting features of different image regions. For training and testing settings, raw pixel images are used as inputs. All input images are first resized into 255 $\times$ 255 and then random/center cropped to 224 $\times$ 224 for training/testing. The standard SGD optimizer with momentum 0.9 and weight decay 5e-4 is used for model training, and the total number of training epochs is set to be 100. The learning rate is set to be 0.001, and will be divided by 10 at 70-th epoch. The batch size is 16 for CUB-200-2011 and Aircrafts datasets, and 64 for rest three large scale datasets to accelerate training. The performance of all models is evaluated with the most widely-adopted Mean Average Precision (MAP@all).

In order to fully represent the state-of-the-art performance and results, nearly all hashing based FGIR methods published in the recent three years, which are summarized in Tab. 2 and Tab. 3 and introduced in Sec. 2, are included as baselines and compared with our proposed method.

4.2. Comparison with SOTA Methods

The experimental results (MAP) of our proposed method implemented based on ResNet-50 and all ResNet-50 based baselines are given in Tab. 2. Besides, Tab. 3 also presents some additional results because MAP results on different code length or backbones are reported by the paper of involved methods. According to the tables, it can be observed that our proposed method significantly outperforms state-of-the-art methods on all datasets and all code lengths. In addition, the superiority of the proposed method is particularly evident when applied to extremely short code length, i.e., 12 bits. This further substantiates the effectiveness of the characteristics matching based hash code generation strategy introduced in this paper, compared to other methods that generate hash codes based on feature vectors. And it also underscores its effectiveness in addressing the aforementioned contradiction in feature learning. As approximate nearest neighbor search methods for large-scale data, efficiency is a more important target than accuracy for hashing based retrieval methods. So this is a very worthy attention result because short codes can further reduce retrieval time and storage costs.

4.3. Ablation Study

In order to demonstrate the effectiveness and necessity of each module in the proposed model, we further conduct an ablation study. Experimental results are summarized in Tab. 4. In this subsection, the proposed method is decomposed into four components: 1) (**B**) that represents the basic hash code generation and training procedure introduced in Sec. 3.2; 2) (**L**) that represents the C-vector optimization based on multi-layer semantic information integration in Sec. 3.3.1, except for 3) (**T**) that represents the cross-layer information transfer supervised by Eq. (11); and 4) (**R**) that represents the multi-region feature embedding introduced in Sec. 3.3.2

According to Tab. 4, it can be observed that each component can contribute to the final performance and the complete CMBH model achieves the best performance. Specifically, due to the lack of effective C-vector optimizing, the individual **B** alone cannot achieve the desired results. This further emphasizes that the effectiveness of the characteristics matching based hash code generation idea relies on the accurate description of fine-grained samples by C-vector.

dataset	bits	ExchNet	A2-Net	FCAENet	SEMICON	FISH	A2-Net^{++}	AGMH	CNET	Ours
CUB	12	25.14	33.83	34.76	37.76	76.77	37.83	56.42	77.10	**84.07**
	24	58.98	61.01	67.67	65.41	79.93	71.73	77.44	82.11	**85.79**
	32	67.74	71.61	73.85	72.61	80.09	78.39	81.95	83.09	**86.21**
	48	71.05	77.33	80.14	79.67	80.88	82.71	83.69	83.92	**86.47**
Aircrafts	12	33.27	42.72	43.92	49.87	88.29	57.53	71.64	86.15	**89.11**
	24	45.83	63.66	75.46	75.08	89.20	73.45	83.45	88.27	**91.45**
	32	51.83	72.51	81.61	80.45	89.28	81.59	83.60	88.40	**91.60**
	48	59.05	81.37	81.34	84.23	89.49	86.65	84.91	89.17	**92.88**
Food101	12	45.63	46.44	44.97	50.00	-	54.51	62.59	83.06	**87.85**
	24	55.48	66.87	76.56	76.57	-	81.46	80.94	85.85	**88.71**
	32	56.39	74.27	81.37	80.19	-	82.92	82.31	86.35	**89.28**
	48	64.19	82.13	83.14	82.44	-	83.66	83.21	86.42	**88.87**
NABirds	12	5.22	8.20	12.56	8.12	-	8.80	-	68.42	**74.42**
	24	15.69	19.15	23.90	19.44	-	22.65	-	75.73	**81.04**
	32	21.94	24.41	31.58	28.26	-	29.79	-	77.11	**81.64**
	48	34.81	35.64	49.74	41.15	-	42.94	-	78.81	**82.07**
Vegfru	12	23.55	25.52	21.76	30.32	79.17	30.54	43.99	81.63	**84.37**
	24	35.93	44.73	50.36	58.45	85.33	60.56	68.05	86.41	**88.63**
	32	48.27	52.75	67.46	69.92	85.43	73.38	76.73	86.80	**88.46**
	48	69.30	69.77	79.76	79.77	85.51	82.80	84.49	87.75	**89.06**

Table 2. MAP (%) results of the proposed method and other SOTA hashing based fine-grained image retrieval methods on all five datasets with code lengths from 12 to 48. All methods are based on ResNet-50. The best results are highlighted in boldface.

Method	Backbone	16bits	32bits	48bits	64bits
DLTH	ResNet50	68.84	77.82	79.97	81.32
sRLH	ResNet18	62.68	69.37	71.27	71.60
FISH	ResNet18	76.03	77.14	78.06	78.34
AGMH	ResNet18	59.68	76.71	80.73	81.43
Ours	ResNet18	**82.10**	**82.86**	**83.36**	**83.31**

Table 3. Additional MAP (%) results on CUB-200-2011 based on different backbone with code lengths from 16 to 64. The best results are highlighted in boldface.

B	**L**	**T**	**R**	12bits	24bits	32bits	48bits
✓				17.89	48.07	55.25	51.90
✓	✓			77.56	81.12	81.61	82.26
✓	✓	✓		80.30	82.16	82.68	82.80
✓	✓	✓	✓	84.07	85.79	86.21	86.47

Table 4. MAP (%) results of ablation study with code length from 12 to 48 on CUB-200-2011 dataset.

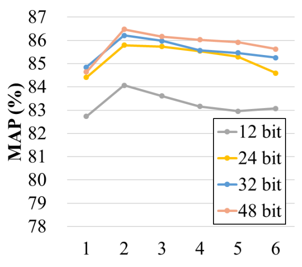
On this basis, through the integration of information across multiple layers and the training associated with semantic information, module **L** enhances the descriptive capacity of C-vector for fine-grained categories, leading to a significant performance improvement. The cross-layer information transfer module **T** further strengthens this effect. Finally, module **R** implements multi-region feature embedding, enhancing C-vectors' capture of local details and their distinctiveness, which also leads to better performance.

4.4. Hyper-parameter Analysis

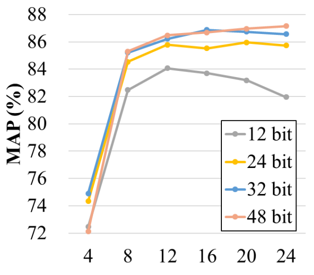
As the core hyper-parameter of the proposed C-vector matching based hash code generation, k defines the number of critical characteristics for each fine-grained category. As a results, it determines the level of detail in describing fine-grained categories and the amount of information to embed in the hash code. Fig. 2a illustrates the impact of different values of k to the final results. In general, when k is greater than 1, the model's performance is better than $k = 1$, especially when the hash code length is relatively high. This demonstrates the necessity of using multiple C-vectors to describe intra-class diversity within each fine-grained category. Moreover, excessively large k can lead to a decline in performance. The main reason may be the limited information representation capacity of the hash codes, and the over-fitting problem resulted by additional parameters. Finally, combining Tab. 2, 3, and Fig. 2a, the performance with k=1 has surpassed most existing SOTA baselines, fur-

Method	Backbone	Params(M) 12 bits	24 bits	32 bits	48 bits	Flops(G) 12 bits	24 bits	32 bits	48 bits
SEMICON	ResNet50	42.3937	42.3691	42.4346	42.4676	7.0910	7.091	7.0911	7.0911
CNET	ResNet50	41.1342	41.1797	41.2100	41.2706	9.3546	9.3546	9.3547	9.3547
FISH	ResNet50	23.9202	23.9227	23.9243	23.9275	4.1321	4.1321	4.1321	4.1321
Ours	ResNet50	14.3219	14.3267	14.3299	14.3363	4.3492	4.3492	4.3492	4.3492
FISH	ResNet18	11.2815	11.2839	11.2855	11.2887	1.8236	1.8236	1.8236	1.8236
sRLH	ResNet18	11.1827	11.1888	11.1929	11.2011	1.8235	1.8235	1.8235	1.8235
Ours	ResNet18	4.2330	4.2378	4.2410	4.2474	1.6696	1.6696	1.6696	1.6696

Table 5. Parameters and computational costs (Flops) for inference and testing. Results are based on CUB-200-2011 dataset.



(a) MAPs with different k.



(b) MAPs with different α.

Figure 2. Parameter analysis on the CUB-200-2011 dataset.

ther emphasizing the point highlighted in the Sec. 1: the essence of fine-grained hashing lies in reflecting differences between samples through hash codes, rather than comprehensive fine-grained feature extraction and embedding.

The L2 normalization involved in the computation of the matching score matrix significantly reduces the absolute values of its elements, thereby causing a shrinkage in gradients during the BP process, which affects the effective training of the model. To amplify the training gradients, the parameter α is introduced in the calculation of the class vector in Eq. (10). As illustrated in Fig. 2b, stable results can be achieved as long as the α reaches a certain magnitude. Furthermore, relatively higher α result in slightly better performance on longer hash codes, while relatively lower α lead to relatively higher performance on shorter hash codes, especially 12 bits. The potential reason is that bigger α can encourage the model to learn more diverse features with higher gradients, and more bits can encapsulate more information. In general, although different values may have specific advantages, setting α to 12 or 16 is sufficient to obtain relatively stable and effective results across various datasets. However, this also inspires the idea that we could adaptively adjust the value of α based on the hash code lengths, to achieve universal performance improvements. Nonetheless, this is not the main focus of this paper, and we plan to leave it for future work.

4.5. Parameters and Computational Costs

As stated in the Sec. 1, one of the primary objectives of this paper is to address the contradiction between efficiency and effectiveness in the development of hashing based models for efficient FGIR. In this section, we record the parameters and computational costs for inference of our method and some other baselines in Tab. 5. Combining Tab. 2, 3 and 5, it can be observed that our method can achieve significantly better performance with much fewer parameters and less or comparable computational costs. Particularly, our method is much more efficient than SEMICON, which is designed based on multi-region learning, as well as CNET that is developed based on mulit-layer integration. These results further demonstrate that the proposed characteristics matching based hash code generation method is an effective solution to aforementioned contradiction in feature learning and contradiction between effectiveness and efficiency.

5. Conclusion

In this paper, we discuss two inherent contradictions about feature learning and efficiency in the design of hash based efficient FGIR method. Based on the analysis, we propose the C-vector matching based hash codes generation method, and design the cross-layer semantic information transfer module and the multi-region feature learning module for training and optimization. Extensive experiments on widely-used datasets demonstrate the superiority of the proposed method in both effectiveness and efficiency.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 62202272, 62172256, 62202278, in part by Natural Science Foundation of Shandong Province under Grant ZR2019ZD06, and in part by the Young Scholars Program of Shandong University.

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 - mining discriminative components with random forests. In *ECCV*, pages 446–461, 2014. 6

[2] Gang Cao, Yingying Zhu, and Xiufan Lu. Fine-grained image retrieval via multiple part-level feature ensemble. In *ICME*, pages 1–6, 2021. 2

[3] Zhen-Duo Chen, Xin Luo, Yongxin Wang, Shanqing Guo, and Xin-Shun Xu. Fine-grained hashing with double filtering. *IEEE Transactions on Image Processing*, 31:1671–1683, 2022. 2

[4] 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 *ECCV*, pages 189–205, 2020. 2

[5] Ruoyi Du, Dongliang Chang, Ayan Kumar Bhunia, Jiyang Xie, Zhanyu Ma, Yi-Zhe Song, and Jun Guo. Fine-grained visual classification via progressive multi-granularity training of jigsaw patches. In *ECCV*, pages 153–168, 2020. 4

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3

[7] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. 5

[8] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge J. Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *CVPR*, pages 595–604, 2015. 6

[9] Saihui Hou, Yushan Feng, and Zilei Wang. Vegfru: A domain-specific dataset for fine-grained visual categorization. In *ICCV*, pages 541–549, 2017. 6

[10] Qing-Yuan Jiang and Wu-Jun Li. Asymmetric deep supervised hashing. In *AAAI*, pages 3342–3349, 2018. 2

[11] Sheng Jin, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, Lei Zhang, and Xian-Sheng Hua. Deep saliency hashing for fine-grained retrieval. *IEEE Transactions on Image Processing*, 29:5336–5351, 2020. 2

[12] Yuchen Liang, Yan Pan, Hanjiang Lai, Wei Liu, and Jian Yin. Deep listwise triplet hashing for fine-grained image retrieval. *IEEE Transactions on Image Processing*, 31:949–961, 2022. 2

[13] Xin Lu, Shikun Chen, Yichao Cao, Xin Zhou, and Xiaobo Lu. Attributes grouping and mining hashing for fine-grained image retrieval. In *ACM MM*, pages 6558–6566, 2023. 3

[14] Wei Luo, Xitong Yang, Xianjie Mo, Yuheng Lu, Larry S. Davis, Jun Li, Jian Yang, and Ser-Nam Lim. Cross-x learning for fine-grained visual categorization. In *ICCV*, pages 8241–8250, 2019. 4

[15] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *CoRR*, abs/1306.5151, 2013. 6

[16] Olga Moskvyak, Frédéric Maire, Feras Dayoub, and Mahsa Baktashmotlagh. Keypoint-aligned rmbeddings for image retrieval and re-identification. In *WACV*, pages 676–685, 2021. 2

[17] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. Supervised discrete hashing. In *CVPR*, pages 37–45, 2015. 2

[18] 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 *ECCV*, pages 531–548, 2022. 3

[19] Catherine Wah, Branson Steve, Welinder Peter, Perona Pietro, and Belongie Serge. The Caltech-UCSD Birds-200-2011 Dataset. *Caltech, Technical Report*, 2011. 6

[20] Xiu-Shen Wei, Jian-Hao Luo, Jianxin Wu, and Zhi-Hua Zhou. Selective convolutional descriptor aggregation for fine-grained image retrieval. *IEEE Transactions on Image Processing*, 26(6):2868–2881, 2017. 2, 5

[21] 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 *NeurIPS*, pages 5720–5730, 2021. 2

[22] Xiu-Shen Wei, Yi-Zhe Song, Oisin Mac Aodha, Jianxin Wu, Yuxin Peng, Jinhui Tang, Jian Yang, and Serge J. Belongie. Fine-grained image analysis with deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):8927–8948, 2022. 1

[23] 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. 2

[24] Xinguang Xiang, Yajie Zhang, Lu Jin, Zechao Li, and Jinhui Tang. Sub-region localized hashing for fine-grained image retrieval. *IEEE Transactions on Image Processing*, 31:314–326, 2022. 2

[25] Xuhui Yang, Yaowei Wang, Ke Chen, Yong Xu, and Yonghong Tian. Fine-grained object classification via self-supervised pose alignment. In *CVPR*, pages 7389–7398, 2022. 4

[26] Yifan Yang, Libing Geng, Hanjiang Lai, Yan Pan, and Jian Yin. Feature pyramid hashing. In *ICMR*, pages 114–122, 2019. 2

[27] Xianxian Zeng and Yanjun Zheng. Cascading hierarchical networks with multi-task balanced loss for fine-grained hashing. *CoRR*, abs/2303.11274, 2023. 3

[28] Qi Zhao, Xu Wang, Shuchang Lyu, Binghao Liu, and Yifan Yang. A feature consistency driven attention erasing network for fine-grained image retrieval. *Pattern Recognition*, 128:108618, 2022. 2

[29] Xiawu Zheng, Rongrong Ji, Xiaoshuai Sun, Yongjian Wu, Feiyue Huang, and Yanhua Yang. Centralized ranking loss with weakly supervised localization for fine-grained object retrieval. In *IJCAI*, pages 1226–1233, 2018. 2

[30] Xiawu Zheng, Rongrong Ji, Xiaoshuai Sun, Baochang Zhang, Yongjian Wu, and Feiyue Huang. Towards optimal fine grained retrieval via decorrelated centralized loss with normalize-scale layer. In *AAAI*, pages 9291–9298, 2019. 2