RESEARCH-ARTICLE

# FoodHash: Context-Aware Proxy Interaction and Fusion for Food Image Retrieval



**Accepted:** 08 December 2025
**Revised:** 16 September 2025
**Received:** 20 February 2025

# FoodHash: Context-Aware Proxy Interaction and Fusion for Food Image Retrieval

PINDAN CAO, School of Computer Science and Artificial Intelligence, Ludong University, China
WEIQING MIN, Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, China
GUORUI SHENG, School of Computer Science and Artificial Intelligence, Ludong University, China
YONGQIANG SONG, School of Computer Science and Artificial Intelligence, Ludong University, China
TAO YAO, School of Computer Science and Artificial Intelligence, Ludong University, China
LILI WANG, School of Computer Science and Artificial Intelligence, Ludong University, China
SHUQIANG JIANG, Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, China

Vision-based food image retrieval has garnered significant attention due to its potential for critical applications in dietary and health management. However, food images exhibit more complex feature distributions and lack the geometric regularity and structured patterns typically observed in general image retrieval tasks. This complexity poses a challenge for existing models to extract fine-grained features and semantic information, thereby compromising retrieval performance. To address this challenge, we propose FoodHash, a context-aware proxy interaction and fusion hashing method for food image retrieval. The method incorporates an Aggregation-Interaction-Propagation (AIP) module that facilitates contextual information exchange among patch tokens within the same feature map, guided by proxy tokens, thereby effectively capturing the intricate details of food images. Furthermore, to leverage the rich semantic information in food images, a Cross-Fusion Module is introduced to efficiently integrate multi-scale information and enhance feature representation. Additionally, we employ a novel loss function to optimize hash learning by ensuring consistency between hash codes and the semantic space, thereby enhancing the learning capability of hash coding. Extensive experiments on three publicly available food datasets demonstrate that FoodHash significantly surpasses existing models in retrieval performance. Specifically, on the ETH Food-101 dataset, FoodHash achieves improvements of 18.1%, 6.7%, 5.2% and 4.5% over the suboptimal method PTLCH for 16-bits, 32-bits, 48-bits and 64-bits hash codes, respectively. The source code will be made publicly available upon publication of the paper.

CCS Concepts: • **Computing methodologies → Visual content-based indexing and retrieval**.

Additional Key Words and Phrases: Food image retrieval, Food computing, Hash retrieval, Deep hashing learning

Authors' Contact Information: Pindan Cao, pindancao@gmail.com, School of Computer Science and Artificial Intelligence, Ludong University, Yantai, China; Weiqing Min, Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, minweiqing@ict.ac.cn; Guorui Sheng, School of Computer Science and Artificial Intelligence, Ludong University, Yantai, China, shengguorui@ldu.edu.cn; Yongqiang Song, School of Computer Science and Artificial Intelligence, Ludong University, Yantai, China, song15253166159@163.com; Tao Yao, School of Computer Science and Artificial Intelligence, Ludong University, Yantai, China, yaotao@ldu.edu.cn; Lili Wang, School of Computer Science and Artificial Intelligence, Ludong University, Yantai, China, wanglili@ldu.edu.cn; Shuqiang Jiang, Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, sqjiang@ict.ac.cn.


ACM 1551-6865/2026/1-ART
https://doi.org/10.1145/3787854

## 1 Introduction

Food plays a crucial role in human life, serving not only as a core resource for meeting basic physiological needs but also as a means of managing health and improving quality of life. Food images have become an essential medium for multimedia information due to their intuitive and easily understandable characteristics, making them one of the main sources of information for people. With rapid advancements in image data acquisition and analysis technologies, the accumulation of large-scale food image datasets has increased significantly, driving growing research interest in accurate and efficient food image processing. Among these research areas, food image retrieval has emerged as a critical subfield of multimedia information retrieval [36, 47] and a major branch of food computing [38], enabling the retrieval of relevant images from food image databases based on a given query image. It has emerged as a research hotspot, garnering considerable attention for its wide-ranging applications in health management, such as dietary recommendations, nutritional assessment systems, and healthcare [19, 46, 55, 57].

However, food images exhibit more heterogeneous feature distributions and face unique challenges compared to generalized image retrieval tasks. First, the visual features of food images are complex and diverse, with numerous categories and variable shapes. The same ingredient may appear completely different depending on the cooking method, while foods with similar colors and shapes are difficult to distinguish. These fine-grained differences increase the difficulty of feature extraction and classification for models. Second, food images frequently contain irrelevant background elements, such as tableware or tabletops, which interfere with feature extraction and hinder the model's ability to focus on key food regions. By contrast, target objects in general image retrieval tasks are often rigid structures (e.g., buildings, vehicles) with clear geometric features or entities with distinctive characteristics (e.g., animals, birds), making their visual features more stable and easier to extract. This disparity in feature distributions makes it challenging for existing image retrieval models to effectively capture the fine-grained details of food images, leading to degraded retrieval performance.

In summary, food image retrieval tasks place higher demands on the model's ability to represent features and capture subtle differences. To achieve effective performance in food image retrieval, researchers must design specialized model architectures and adaptive algorithms to handle the complex distributions and diverse features of food images. In recent years, content-based image retrieval (CBIR) [10] has emerged as a key area of research within food image retrieval. Researchers have proposed various improved methods leveraging visual features for image search. However, relying solely on CBIR often leads to high computational costs and inefficiency. By contrast, hash-based image retrieval offers significant advantages in storage efficiency and retrieval speed through the use of hash codes. As a result, hash-based methods have become a prominent research direction in image retrieval [5, 56].

Hashing methods retain unique advantages in food image retrieval, despite the rising popularity of deep feature embeddings and vector search in general image retrieval for their high accuracy in capturing semantic similarity. Hashing techniques convert high-dimensional features into compact binary codes, achieving a sublinear time complexity for approximate nearest neighbor search via the Hamming distance. This significantly reduces storage overhead and retrieval latency—crucial for real-time applications in food-related health management. In contrast, while deep embeddings offer rich semantic content, they are less suitable in resource-constrained environments due to their high computational cost and significant memory usage. With fast response times and low memory consumption, hash-based methods are widely regarded as one of the most efficient image retrieval techniques [12, 15, 21, 29, 31, 33, 34, 42, 52, 53, 58, 59, 61]. For food imagery, the visual diversity and disordered distribution necessitate retrieval techniques capable of efficiently processing subtle variations. Hashing's binary encoding mechanism achieves high-precision matching in resource-constrained scenarios by preserving key semantic relationships while compressing irrelevant noise—a capability particularly critical in practical applications. For instance, in mobile health applications, users' uploaded food images require instant retrieval of similar items for nutritional assessment or dietary recommendations. Hash's low latency and high efficiency significantly enhance

user experience and system usability, thereby driving our exploration of advanced hashing techniques to improve food image retrieval performance.

Current mainstream deep hashing methods rely on convolutional neural networks (CNNs) [24] to extract high-dimensional semantic features, mapping these features into binary hash codes via hashing functions. However, CNNs primarily depend on convolution kernels to compute pixel relationships between local regions during feature extraction. This limitation significantly impacts the overall performance of image retrieval. In recent years, the Vision Transformer (ViT) [9], an extension of the Transformer [51], has gained attention for its ability to capture long-range dependencies between different image regions through the self-attention mechanism. This capability allows ViT to effectively extract global features. As a novel architecture that replaces traditional CNNs, ViT has demonstrated outstanding performance in various vision tasks, including image segmentation [6], object detection [28] and image classification [37], consistently outperforming most CNN-based methods. However, ViT exhibits certain limitations in capturing fine-grained local features, a problem that becomes particularly pronounced when processing food images with complex structures. Therefore, this paper proposes a hash-hybrid framework for food image retrieval that combines CNN and Transformer architectures.

The main contributions of this paper can be summarized as follows:

- A new context-aware proxy interaction and fusion deep hashing framework for food image retrieval, called FoodHash, is constructed to efficiently capture the rich and diverse features in food images through a hybrid architecture of dual attention.
- The Aggregation-Interaction-Propagation module is introduced, which enables comprehensive information exchange between patch tokens within the same feature map through three key mechanisms. The mechanisms effectively address the complex detail features in food images, enhancing their visual feature representation.
- The Cross-Fusion Module (CFM) is constructed to establish an interactive channel between local detail features and global semantic features through a bidirectional key-value projection mechanism. This effectively aggregates semantic information across different scales, enhancing the model's ability to distinguish between similar images.
- A deep hash loss function combining polarization loss and an enhanced cross-entropy loss has been devised to enhance both intra-class consistency and inter-class separability within hash representations. The proposed method was experimentally evaluated across three extensively studied food datasets. Results demonstrate FoodHash's superior performance in food image retrieval tasks under complex semantic conditions.

## 2 Related Work

### 2.1 Food Image Retrieval

With the emergence of several publicly available food image datasets [2, 7, 23, 39, 40], food-related research has become a popular topic. Food image retrieval has gained widespread attention in recent years as one of the core research directions in food computing. To address the challenges posed by complex features and fine-grained variations in food images, researchers have proposed various solutions, ranging from traditional image descriptor methods [13] to deep learning-based models, achieving significant progress. For instance, Song et al. [49] introduced the noise-robust NoLoTransformer network, which adaptively adjusts the weights of different image patches to reduce the impact of irrelevant noise on feature extraction. They also incorporate convolutional structures between Transformer layers to extract more discriminative fine-grained features, effectively addressing the challenges of background noise and fine-grained details in food images.

Furthermore, Song et al. [48] improved the classic metric learning framework by proposing an adaptive maximization sampling method that enhances generalization by avoiding excessive distance compression in the

sample space. They also design a gradient-adaptive optimization strategy to reduce the optimization weight of boundary samples, mitigating intra-class over-compression. This approach significantly improves the model's generalization capability across domains with substantial category differences.

## 2.2 CNN-Based Image Retrieval Methods

With the rapid development of CNNs, CNN-based deep hashing methods for image retrieval have garnered significant attention. Deep Supervised Hashing (DSH) [32] uses CNNs to quantize network outputs into binary hash codes by applying a regularizer on the real-valued outputs to generate discrete binary values. Deep Pairwise Supervised Hashing (DPSH) [26] is the first method to learn feature representations and hash functions using pairwise labels simultaneously. HashNet [4] introduces an extended approach based on the tanh function to achieve a smooth transition from real-valued features to binary codes, addressing the ill-posed gradient problem encountered when optimizing deep networks with non-smooth binary activations using continuous methods. This approach accurately learns binary hash codes from imbalanced similarity data. Building upon HashNet, Deep Cauchy Hash for Hamming space retrieval (DCH) [3] proposes a novel pairwise cross-entropy loss based on the Cauchy distribution, which significantly penalizes similar image pairs with Hamming distances exceeding a given Hamming radius threshold. GreedyHash [50] applies the sign function at the hashing layer. IDHN [60] employs cross-entropy loss and mean squared error loss to enable multi-label image retrieval. Central Similarity Quantization (CSQ) [58] optimizes hash centers based on the central similarity of data points. Maximum margin Hamming hashing (MMHH) [22] achieves constant-time search in Hamming space by performing hash lookups, significantly improving retrieval efficiency for very large databases.

Deep Polarized Network (DPN) [12], designed for supervised learning of accurate binary hash codes, introduces a novel polarization loss that utilizes a bitwise hinge loss to push different output channels away from zero. This algorithm minimizes the original Hamming-distance-based loss without introducing quantization errors while avoiding the complexity of binary optimization problems. Bi-Half Net [30] proposes an unsupervised deep hashing network to generate high-quality hash codes. This network minimizes the continuous feature learning process through a parameter-free layer, approximating an optimal uniform distribution of hash codes. ADSH [20] introduces an asymmetric deep supervised hashing method, where deep hashing functions are learned for query points, while database points are directly learned. This approach allows for more effective utilization of label information. Hashing-guided hinge function (HHF) [54] proposes a novel hinge loss function for supervising networks to generate hash codes. This function prevents hash learning from converging to suboptimal local minima in metric learning, addresses semantic loss issues during network training, and promotes richer feature extraction. DFPH [43] and MDFF-SH [1] combine convolutional neural network backbones with feature pyramid networks to fuse low-level structural features and high-level semantic features across multiple scales, thereby generating more discriminative representations.

Although CNN architectures have demonstrated strong performance in the field of image retrieval, they still have inherent limitations. CNN extracts image features by using convolutional kernels to compute pixel relationships in small regions, which makes them more focused on capturing local information. Recently, researchers have observed that ViT, which focuses on global feature extraction, has exhibited superior performance in the image retrieval domain.

## 2.3 Transformer-Based Image Retrieval Methods

As a rising alternative to CNN architectures in computer vision, Transformer was initially introduced by A. Vaswani [51] for natural language processing (NLP) tasks to address the issue of long-range dependencies in data. A. Dosovitskiy [9] was the first to apply a vision transformer to image classification tasks, where images were directly processed as a sequence of patches, achieving significant effects. Since then, various ViT variants

have been proposed, demonstrating strong performance. Swin Transformer [35] introduces a hierarchical vision Transformer architecture, replacing standard global self-attention with a shifted window attention mechanism. BiFormer [62] proposes a novel Transformer model that employs two-stage routed attention to filter out irrelevant key-value pairs in coarse regions, thereby improving overall performance. Overall, ViT and its variants are rapidly evolving, with increasing research focused on exploring their applicability across various computer vision tasks.

Inspired by the latest advancements in ViT, TransHash [8] proposes a fully Transformer-based deep hashing framework, which effectively captures both global and local discriminative features through an innovative dual-stream feature learning module. HashFormer [25] further leverages ViT as the backbone network, treating binary codes as intermediate representations for the proxy task of image classification, and introducing a mean average precision loss to optimize retrieval performance. X. Ren et al. [45] designed a hash retrieval model combining contrastive learning and ViT. By incorporating contrastive learning into the feature learning process of ViT and designing a multi-objective loss function, the model maximized feature consistency across different views of the same image, thereby enhancing the representativeness of the hash codes. S. R. Dubey [11] proposed a hash-based image retrieval method leveraging ViT, where a pre-trained ViT on ImageNet was used as the backbone, with additional hash-specific heads. MSViT [27] extracts multi-scale features by processing image patches of different sizes, achieving more precise feature representations through efficient fusion.

However, the ability of ViT to learn local features is relatively limited compared to CNNs. As a result, exploring hybrid deep hashing methods that combine CNNs and Transformers has become a promising area of research. CMT [14] proposes a novel hybrid network based on Transformers, which leverages Transformers to capture long-range dependencies while using CNNs to extract local information. EdgeViT [41] combines local and global attention and successfully balances accuracy and efficiency with state-of-the-art CNNs and ViTs by introducing a cost-effective local-global-local information exchange bottleneck. Building on these architectures, several works [16, 17] have introduced improvements to the self-attention mechanism of vision transformers to enhance computational efficiency. PTLCH [44] integrates CNN pooling with ViT to simultaneously capture global information and spatial dimensions. Additionally, it proposes a novel loss framework that optimizes the weights of hard and noisy samples during the hashing process, achieving optimal matching. HybridHash [18] introduces a hierarchical backbone network with block aggregation capabilities, enabling localized self-attention. These recent advances in hybrid architectures based on vision Transformers and CNNs provide new insights and inspiration for image retrieval tasks. This paper proposes a hybrid architecture called FoodHash, which uses ViT as the backbone network and combines CNN with ViT to capture the fine-grained features and global semantic information of food images.

## 3 Method

### 3.1 Problem Description

Given a dataset $X$ contains $N$ images, and its expression is defined as $X = \{x_i\}_{i=1}^{N}$, where $x_i$ represents the $i$-th image in the dataset. An image $x_i$ in the dataset is associated with a corresponding label $y_i$. The label dataset $Y$ is composed of these label vectors $y_i$ and is defined as $Y = \{y_i \in (0,1)^M\}_{i=1}^{N}$, where $M$ denotes the total number of image categories. If the sample $x_i$ belongs to category $j$, the $j$-th element of the label vector $y_{i,j} = 1$; otherwise, $y_{i,j} = 0$.

The objective of deep hashing image retrieval is to learn a hash mapping function $H : X \to \{-1, +1\}^k$, which maps samples to a set of hash codes $U = \{u_i \in \{-1, +1\}^k\}_{i=1}^{N}$, where $k$ denotes the length of the hash codes. This mapping relationship in FoodHash can be divided into two parts. In the first part, the Transformer Encoder and the AIP module jointly learn the image features to generate the corresponding feature representations. The two types of feature representations are cross-fused to obtain the final feature representation $F = \{f_i\}_{i=1}^{N}$. In the

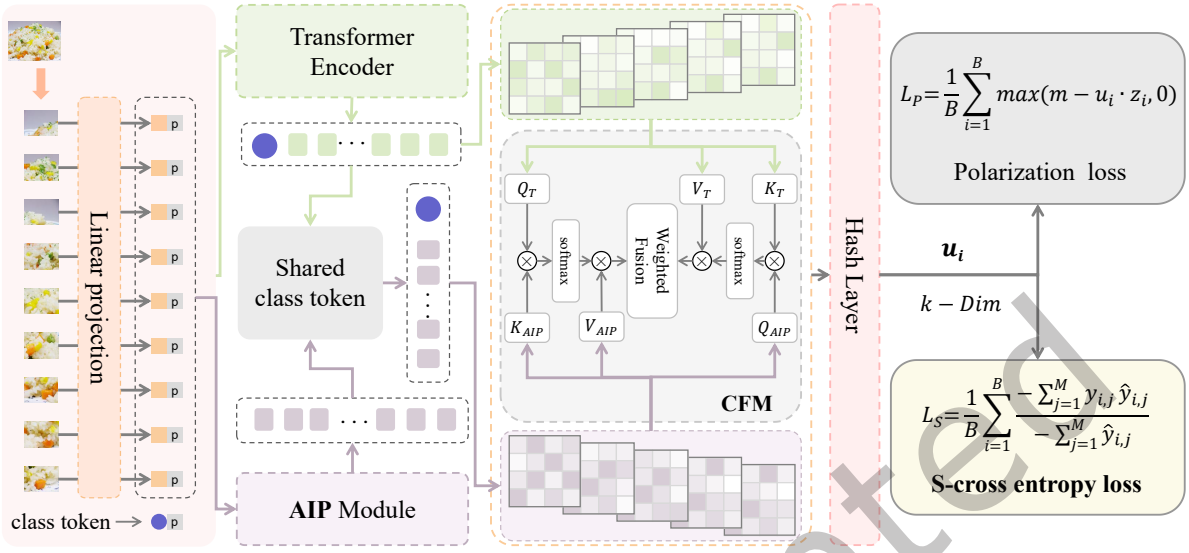


Fig. 1. The overall framework of FoodHash.

second part, the feature representation $F$ is transformed into the hash codes matrix $U$ using the sign function $u_i = \text{sign}(f_i)$. Specifically, when $f_i > 0$, $\text{sign}(f_i) = 1$; otherwise, $\text{sign}(f_i) = -1$.

## 3.2 FoodHash Network Architecture

We design a variant FoodHash using ViT as the base network. The overall network architecture of FoodHash is shown in Fig. 1.

*3.2.1 Patch and Position Embedding.* Given an input food image $x_i \in \mathbb{R}^{H\times W\times C}$, we first process the input image using a 2D convolution to divide it into several non-overlapping patches of fixed size. Next, a linear mapping is performed on each patch embedding, which is converted to a feature vector to obtain the initial representation $X_V$ of the image patches. A class token embedding $CT$ is also introduced as a learnable parameter initialized with zero. The class token is used as a representation of the global image, which is concatenated with the embedded representation $X_V$ of the image patch to generate the extended embedded representation $X_E$. To capture spatial positional information, a learnable positional embedding matrix $E_{pos}$ is added to preserve positional embedding information. The initial feature representation $X_0$ incorporating the position embeddings is defined as follows:

$$X_0 = X_E + E_{pos} \tag{1}$$

*3.2.2 Transformer Encoder.* The Transformer Encoder module consists of $G$ stacked transformer blocks. Each block includes a Multi-Head Self-Attention (MSA) and a Multilayer Perceptron (MLP). The Transformer Encoder employs Layer Normalization (LN) and residual connections to enhance the stability and propagation of the feature representations.

Specifically, for each transformer block, the input features $X_{T-1}$ are first processed with Layer Normalization and passed through the MSA layer to compute the updated feature representation:

$$X_L = MSA\,(LN(X_{T-1})) + X_{T-1} \tag{2}$$

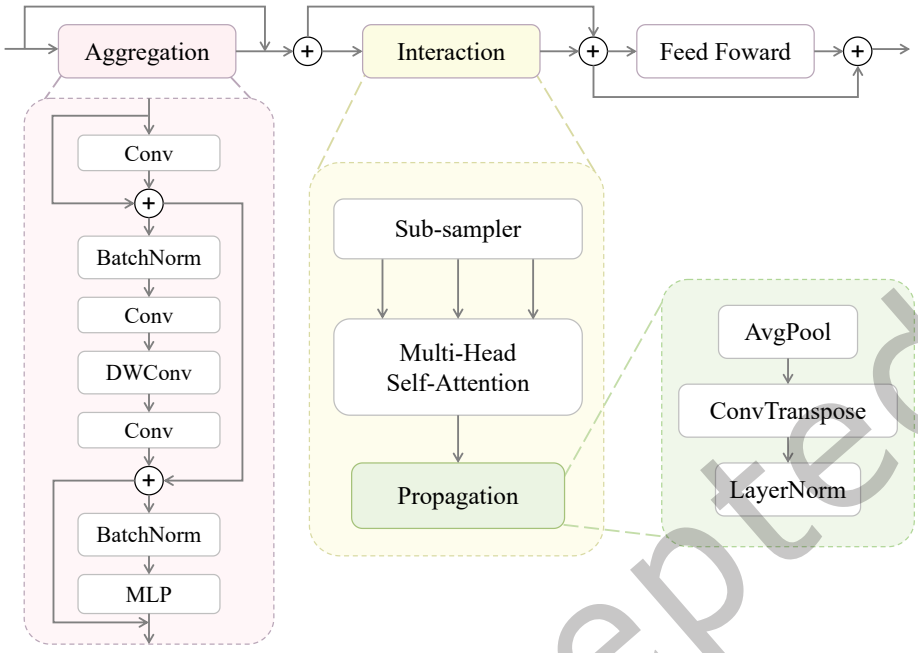


Fig. 2. The detailed structure of the proposed AIP module. AIP includes three main mechanisms: Aggregation (pink), Interaction (yellow) and Propagation (green).

Then, the updated feature representation $X_L$ is further processed with Layer Normalization and passed through the MLP layer for nonlinear feature projection to obtain the final output representation $X_T$:

$$X_T = MLP(LN(X_L)) + X_L \tag{3}$$

where $T \in \{1, 2, ..., G\}$, $X_{T-1}$ is the output of the previous layer block. $X_L$ is the intermediate representation updated by the MSA layer, and $X_T$ is the final output of the current block. Layer Normalization enhances the numerical stability of the input, while the MSA mechanism establishes global relationships between blocks and generates new feature representations. Subsequently, the MLP further performs nonlinear projection on the features, and the residual connections ensure the stability and continuity of the feature representations.

## 3.3 AIP Module

The AIP module, as shown in Fig. 2, primarily includes three key mechanisms: aggregation, interaction, and propagation.

*3.3.1 Aggregation.* A set of proxy tokens is generated by integrating information from local neighborhood tokens through an aggregation mechanism. In the aggregation phase, we perform feature aggregation for each image patch through depthwise convolution operations to further extract local features within the patch and generate richer feature representations. This operation effectively captures detailed features inside the patch by aggregating the local information within a window of size 2 × 2. As shown in Fig. 3(a), the process can be formulated as follows:

$$X_C = X_V + Conv(X_V) \tag{4}$$

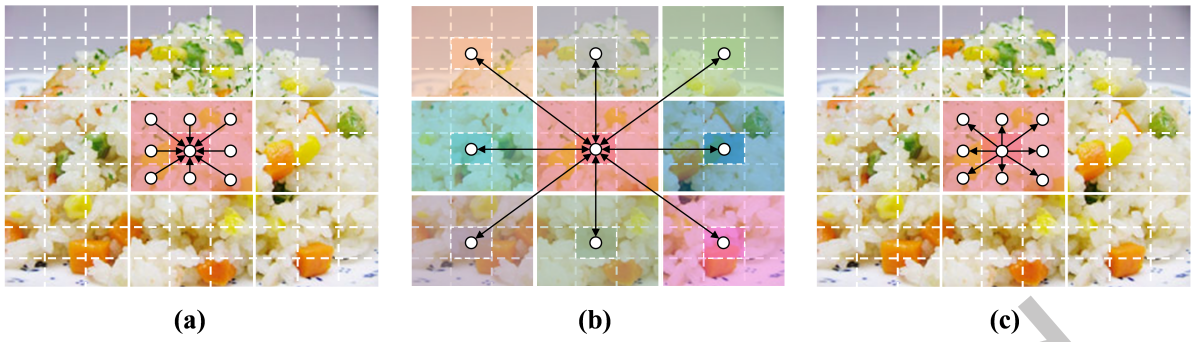


Fig. 3. Illustration of the three key mechanisms involved in the proposed AIP module. In the example, (a) Local information from neighboring tokens within the region is first aggregated to the proxy tokens. (b) Information is exchanged between proxy tokens through attention. (c) The global contextual information encoded in the proxy tokens is propagated to the neighboring tokens within the region.

$$X_D = X_C + Conv(DWConv(Conv(BN(X_C)))) \tag{5}$$

$$X_a = X_D + MLP(BN(X_D)) \tag{6}$$

where $X_C$ and $X_D$ are the intermediate representation after the convolution operation.

*3.3.2 Interaction.* Global sparse interaction attention that introduces long-term relationships between proxy tokens via an interaction mechanism. In the interaction phase, we adopt a sparse proxy patch token selection strategy by sampling the proxy patch tokens across the entire feature map space and performing self-attention operations on the selected proxy patch tokens. This strategy effectively reduces the computational overhead while preserving the ability to model global features. During this process, all proxy patch tokens in the space are involved in the computation of self-attention as queries. As shown in Fig. 3(b), the formula is as follows:

$$Q, K, V = reshape(QKV(X_a)) \tag{7}$$

$$Attn(Q, K, V) = reshape(softmax\left(\frac{QK^\top}{\sqrt{d_k}}\right)V) \tag{8}$$

$$X_P = LN(ConvTranspose(AvgPool(Attn(Q, K, V)))) \tag{9}$$

where $QKV(X)$ is the linear projection of the input $X$ to obtain $Q$, $K$ and $V$, respectively. $d_k$ is the dimension of each attention head, and the softmax operation normalizes all spatial tokens to ensure attention distribution. Through this sparse sampling strategy, the interaction phase enables effective modeling of global features with controlled computation.

*3.3.3 Propagation.* The information known to the proxy tokens is spread to the sub-tokens with the same proxy token through the propagation mechanism. In the propagation phase, we use the transposed convolution operations to propagate the global context information encoded in the interaction phase to their respective neighboring patch tokens. Specifically, the transposed convolution is able to expand the receptive field of the feature map while preserving the spatial structure, enabling the transfer and exchange of global information. As shown in Fig. 3(c), the AIP module is able to communicate comprehensive information between image patches, resulting in capturing the global features of the image through the above operations, which enhances

the coherence and consistency of the feature representation. Specifically, as shown in Equation 9. After that $X_P$ is projected to get the final feature $X_A$ output by the AIP module.

The overall workflow of the AIP module is as follows: Initially, $X_V$ leverages the locality bias of convolution to extract spatially coherent local features, followed by the introduction of batch normalization (BN) to normalize intermediate activation values, stabilizing training dynamics and accelerating convergence. Subsequently, depthwise separable convolution is applied to further optimize parameter efficiency while preserving spatial relationships. An MLP layer enhanced with residual connections introduces nonlinear transformations, enhancing the expressive power of local features. Next, a subsampling operation reduces the number of proxy tokens to control computational complexity, while the multi-head attention mechanism captures global dependencies through attention, and the softmax operation ensures the probability normalization of the attention distribution. AvgPool aggregates the attention output to summarize key features and reduce noise, while ConvTranspose serves as an upsampling operation to expand the receptive field, broadcasting global information while maintaining spatial structural consistency. Finally, LN is applied to normalize the entire block output, maintaining numerical stability and facilitating the learning capacity of subsequent layers. This design philosophy optimizes the information flow, transitioning from local feature extraction to global modeling, and further through compression, expansion, and stabilization, achieving module-level consistency. Through the three phases of Aggregation, Interaction and Propagation, the AIP module enables efficient information exchange and fusion across the feature map, effectively capturing both local and global features, making the final output features more discriminative.

### 3.4 Cross-Fusion Module

The Cross-Fusion Module achieves feature fusion through an innovative bidirectional cross-attention mechanism: it first utilizes mutually inverse attention paths (local features querying global context/global features querying local details) to enable dynamic bidirectional feature negotiation, and then adaptively fuses the bidirectional results using learnable weights. Compared to the linear combination methods of traditional concatenation or weighted averaging, our proposed CFM not only generates spatially adaptive fusion weights to address feature misalignment issues but also fully exploits complementary information through equal dialogue between features. The specific implementation is as follows:

First, we extract the global feature $CT$ from the output features $X_T$ of the Transformer Encoder as the representative feature of the global image. This feature is then concatenated with the output features $X_A$ of the AIP module. Through this fusion operation, the global and detailed information can be effectively combined, enhancing the model's ability to understand the details and global structure of the image. The concatenated features are defined as:

$$X_{AIP} = Concat(CT, X_A) \tag{10}$$

Next, the concatenated features $X_{AIP}$ and the main output features $X_T$ of the Transformer Encoder are separately subjected to linear projection to obtain the following feature matrices: for $X_{AIP}$, the query matrix $Q_{AIP}$, key matrix $K_{AIP}$ and value matrix $V_{AIP}$ are generated; for $X_T$, the query matrix $Q_T$, key matrix $K_T$ and value matrix $V_T$ are generated.

Subsequently, we perform two cross-attention calculations to capture the interaction relationships between the two sets of features, respectively:

(1) Using the query matrix $Q_{AIP}$ of $X_{AIP}$ and the key matrix $K_T$ and value matrix $V_T$ of $X_T$ as inputs:

$$F_1 = softmax\left(\frac{Q_{AIP}K_T^\top}{\sqrt{D}}\right)V_T \tag{11}$$

(2) Using the query matrix $Q_T$ of $X_T$ and the key matrix $K_{AIP}$ and value matrix $V_{AIP}$ of $X_{AIP}$ as inputs:

$$F_2 = softmax\left(\frac{Q_T K_{AIP}^{\top}}{\sqrt{D}}\right) V_{AIP} \tag{12}$$

where $\sqrt{D}$ is a scaling factor used to stabilize gradient computation.

Finally, the weighted fusion layer fuses the outputs $F_1$ and $F_2$ from the two cross-attention calculations to synthesize the interaction information between different features:

$$F = \omega F_1 + (1 - \omega) F_2 \tag{13}$$

where $\omega$ is a learnable parameter used to adjust the importance of the interaction between the two types of features.

## 3.5 Loss Function

*3.5.1 Polarization Loss.* Given $M$ categories and a hash bit length $k$, the target vector matrix $Z \in \{-1, +1\}^{M \times k}$.The target vector is initialized using a binary assignment strategy. The initialization rule is as follows:

$$Z_{i,j} = \begin{cases} -1, & if\, j \in S \\ +1, & otherwise. \end{cases} \tag{14}$$

where $p$ is a hyperparameter that controls the sparsity of the target vectors, $\forall i \in \{1, 2, ..., M\}$, $S \in \{1, 2, ..., k\}$, and $|S| = |p \cdot k|$.

When the hash code $u_i$ of the $i$-th sample is learned, the goal is to get as close as possible to the target vector matrix $Z$. Based on the label $y_i$, the corresponding target vector $z_i$ is selected for each $u_i$. The target vector $z_i$ can be directly extracted from the target matrix using the index of the maximum value in the label:

$$z_i = Z[\arg\max(y_i)] \tag{15}$$

The Polarization loss function aims to minimize the difference between the learned hash codes $u_i$ and its corresponding target vector $z_i$, which is defined by the formula:

$$\mathcal{L}_P = \frac{1}{B} \sum_{i=1}^{B} \max(0, m - u_i \cdot z_i) \tag{16}$$

where $m$ is a hyperparameter used to control the polarization threshold, and $B$ represents the batch size.

The gradient of the polarization loss can be derived as:

$$\frac{\partial \mathcal{L}_P}{\partial u_i} = \frac{1}{B} \cdot \begin{cases} -z_i, & if\, m - u_i \cdot z_i > 0 \\ 0, & otherwise. \end{cases} \tag{17}$$

When $u_i \cdot z_i < m$, the gradient points toward $-z_i$, driving $u_i$ to move in the direction of $z_i$, thereby increasing the inner product. When $u_i \cdot z_i \geq m$, the gradient becomes zero, halting the optimization for that sample. The polarization loss function encourages $u_i \cdot z_i$ to exceed the threshold $m$ by penalizing deviations below $m$, thereby promoting the polarization of the hash code distribution and enforcing class compactness in the Hamming space.

*3.5.2 S-Cross Entropy Loss.* We have incorporated the concept of normalization into the traditional cross-entropy loss by redistributing the contribution values of different category predictions, penalizing overconfident predictions and encouraging a more balanced distribution. First, the hash codes $u_i$ are mapped to a class probability distribution through the classification head:

$$\hat{y}_{i,j} = \log\left(\frac{e^{u_{i,j}}}{\sum_{j=1}^{M} e^{u_{i,j}}}\right) \tag{18}$$

where $\hat{y}_{i,j}$ represents the result of applying the log operation to the probability of the $i$-th sample belonging to the $j$-th category. Then the loss function is calculated, and for the $i$-th sample, the loss is:

$$\mathcal{L}_i = \frac{-\sum_{j=1}^{M} y_{i,j}\hat{y}_{i,j}}{-\sum_{j=1}^{M} \hat{y}_{i,j}} \tag{19}$$

The proposed normalization term introduces a regularization effect proportional to the negative entropy of the predicted distribution, thereby encouraging the model to maintain wider decision boundaries and improving class separation. Entropy regularization can enhance generalization ability by preventing the model from converging to overly sharp distributions. Finally, the S-cross entropy loss is defined as:

$$\mathcal{L}_S = \frac{1}{B}\sum_{i=1}^{B} \mathcal{L}_i \tag{20}$$

The gradient calculation of the S-cross entropy loss is as follows: First, compute the gradient of $\hat{y}_{i,j}$ with respect to $u_{i,j}$. Define the probability of $softmax$ as:

$$p_{i,j} = \frac{e^{u_{i,j}}}{\sum_{j=1}^{M} e^{u_{i,j}}}, \hat{y}_{i,j} = log(p_{i,j}) \tag{21}$$

The gradient of $\hat{y}_{i,j} = \log(p_{i,j})$ is:

$$\frac{\partial \hat{y}_{i,j}}{\partial u_{i,l}} = \frac{1}{p_{i,j}} \cdot \frac{\partial p_{i,j}}{\partial u_{i,l}} = \begin{cases} 1 - p_{i,l}, & \text{if } j = l, \\ -p_{i,j}, & \text{if } j \neq l. \end{cases} \tag{22}$$

For the $i$-th sample, first compute the gradient of the numerator of $\mathcal{L}_i$:

$$\frac{\partial}{\partial u_{i,l}}\left(-\sum_{j=1}^{M} y_{i,j}\hat{y}_{i,j}\right) = -\sum_{j=1}^{M} y_{i,j} \cdot \frac{\partial \hat{y}_{i,j}}{\partial u_{i,l}} \tag{23}$$

Substitute the gradient of $\hat{y}_{i,j}$:

$$\frac{\partial}{\partial u_{i,l}}\left(-\sum_{j=1}^{M} \hat{y}_{i,j}\right) = -\left((1 - p_{i,j}) + \sum_{j\neq l}^{M} -p_{i,l}\right) \tag{24}$$

Since $y_{i,j} \in \{0, 1\}$, only the class $j$ with $y_{i,j} = 1$ contributes to the gradient. Assuming $y_{i,j} = 1$ holds for only one class $j^*$, then:

$$\frac{\partial \mathcal{L}_i}{\partial u_{i,l}} \propto \begin{cases} -(1 - p_{i,j*}), & \text{if } j^* = l, \\ p_{i,l}, & \text{if } j^* \neq l. \end{cases} \tag{25}$$

During the training process, in order to stabilize the distribution of the hash codes, the hash code $u_i$ is updated using ternary quantization, specifically:

$$u_i = \begin{cases} \text{sign}(u_i), & if|u_i| > m \\ 0, & otherwise. \end{cases} \tag{26}$$

The hash codes matrix is updated to its signified form, specifically:

$$\mathrm{U} = \text{sign}(\mathrm{U}) \tag{27}$$

Table 1. Detailed settings of three food datasets.

| Datasets | Label | Class | Training set | Test set | Database |
|---|---|---|---|---|---|
| ETH Food-101 | Single | 101 | 12,120 | 3,030 | 97,970 |
| Vireo Food-172 | Single | 172 | 20,640 | 5,160 | 105,081 |
| UEC Food-256 | Single | 256 | 20,480 | 5,120 | 26,275 |

The target vector $Z$ is adaptively updated during the training process using the learned hash code matrix $U$ and label matrix $Y$. The update rule is as follows:

$$Z = \text{sign}(Z^\top U) \tag{28}$$

where $Y \in \{0,1\}^{B\times M}$ is the label matrix, and $U \in \{-1,+1\}^{B\times k}$ is the hash code matrix.

Combining the Polarization loss and the S-cross entropy loss, the total loss $\mathcal{L}$ is defined as:

$$\mathcal{L} = \mathcal{L}_P + \lambda \mathcal{L}_S \tag{29}$$

where $\lambda$ is a hyperparameter that balances the weights of the two losses. The optimization process of total loss can be divided into two convergent subproblems:

(1) Subproblem 1 (Fix Z, optimize U):

$$\min_U \mathcal{L}_P(U;Z) + \lambda \mathcal{L}_S(U;Y) \tag{30}$$

Update via gradient descent:

$$U^{(t+1)} = U^{(t)} - \eta \left( \frac{\partial \mathcal{L}_P}{\partial U} + \lambda \frac{\partial \mathcal{L}_S}{\partial U} \right) \tag{31}$$

(2) Subproblem 2 (Fix U, update Z):

$$Z^{(t+1)} = \text{sign}\left( (Z^{(t)})^\top U^{(t+1)} \right) \tag{32}$$

This iterative process satisfies:

$$\mathcal{L}(U^{(t+1)}, Z^{(t+1)}) \le \mathcal{L}(U^{(t)}, Z^{(t)}) \tag{33}$$

That is, the target function decreases monotonically, ensuring convergence to a local optimum.

## 4 Experiments

### 4.1 Datasets and Splits

In this paper, three popular food datasets are used. In order to avoid the effect of unbalanced data distribution, each category of the dataset is reclassified according to the retrieval protocol [4], as shown in Table 1. Each dataset is split into training and test sets, where training data is also used as database data. The details are as follows:

**ETH Food-101** [2] contains images from 101 food categories and is primarily used for image classification. There are 101,000 food images in total, of which there are 250 test images and 750 training images for each category. In this paper, the dataset is redivided, with 120 images randomly selected for each category for training and 30 images for testing.

**Vireo Food-172** [7] divides food into 172 major categories, containing 200 to 1,000 food images in each category, totaling 110,241 food images, which basically cover most of the food types in daily life. In this paper, 120 images are randomly selected for each category for training and 30 images per category were used for testing.

**UEC Food-256** [23] contains 31,395 images of 256 dishes. The dataset has an inconsistent number of each category, which requires the ability to understand complex visual features and extensive category recognition to distinguish category semantic similarities. In this paper, 80 images per category were randomly selected for training, and 20 images per category were used for testing.

### 4.2 Model Evaluation Metrics and Settings

The quality of image retrieval using FoodHash is evaluated based on five metrics [12, 50]: mean Average Precision (mAP), Precision-Recall (PR) curves, precision at different top-N returned results (P@N), average training and retrieval time, and memory usage. In this paper, mAP@1000 is used for all three datasets. PR curves are judged to have good performance when the curve of Algorithm A is able to encompass the curve of Algorithm B or when the curve of Algorithm A has a large area.

In the comparative experiments, we employed the same dataset and data partitioning strategy, with the batch size uniformly set to 128, in order to ensure the objectivity of the experimental results. The operating system version is Ubuntu 20.04 LTS. We trained on a NVIDIA A800 GPU (80GB), Intel(R) Xeon(R) Platinum 8358 CPU @ 2.60GHz, 64GB RAM, 1TB SSD. During the experimental data preprocessing, the input image size was resized to 256 × 256 and then randomly cropped to 224 × 224 [18, 27, 44]. Additionally, random horizontal flipping was applied to augment the images during model training. During testing, the input images were resized to 256 × 256 and then center-cropped to 224 × 224. All networks are trained directly on three datasets, optimized using Adam, with the initial learning rate set to $10^{-5}$. All models were trained for 150 epochs, with testing conducted every 30 epochs to record the best results. The hyperparameter $p$, which controls the sparsity of target vectors, is set to 0.5, and the threshold parameter $m$, which governs the polarization of hash codes, is set to 1. The parameter $\lambda$ for the equilibrium loss is set to 0.1.

### 4.3 Comparison of Representative Methods

This paper compares the proposed method with representative ResNet-based, AlexNet-based and Trannformer-based image retrieval methods. Table 2 shows the mAP values of these methods on three commonly used food datasets, with the bold data in the table indicating the best performance in the current column. Compared with the mAP of the suboptimal method, on the ETH Food-101 dataset, the mAP of the 16-bits, 32-bits, 48-bits and 64-bits hash codes of FoodHash increased by 18.1%, 6.7%, 5.2% and 4.5%, respectively. On the Vireo Food-172 dataset, the mAP of the 16-bits and 32-bits hash codes of FoodHash increased by 8.0% and 1.2%, respectively. On the UEC Food-256 dataset, the mAP of 16-bits, 32-bits, 48-bits and 64-bits hash codes of FoodHash increases by 17.5%, 6.0%, 2.4% and 0.6%, respectively. Although the mAP of FoodHash for 48-bits and 64-bits hash codes on the Vireo Food-172 dataset is second only to that of the PTLCH method, it improves by 18.6% and 5.3%, respectively, when compared to the next-best method, and FoodHash is the best in terms of mAP performance for 16-bits and 32-bits hash codes.

Fig. 4, Fig. 5 and Fig. 6 show the PR curves of FoodHash and seven representative deep hashing methods from the past five years on the ETH Food-101, Vireo Food-172 and UEC Food-256 datasets for hash codes lengths of 16-bits, 32-bits, 48-bits and 64-bits, respectively. The figures show that FoodHash consistently demonstrates a superior trade-off between precision and recall across all experimental settings, with its PR curves significantly higher than those of other methods. Fig. 7, Fig. 8 and Fig. 9 respectively present the P@N curves of FoodHash and the compared methods under different code lengths. The results clearly demonstrate that, in most cases, the precision curves of FoodHash are higher than those of the comparison methods, highlighting its significant advantage. The experiments show that FoodHash can effectively capture and preserve the global and detailed features of food images under various hash code lengths, which shows advantages in complex food image retrieval tasks.

Table 2. MAP scores (%) of different methods compared on three food datasets.

| Methods | ETH Food-101 | | | | Vireo Food-172 | | | | UEC Food-256 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 16-bits | 32-bits | 48-bits | 64-bits | 16-bits | 32-bits | 48-bits | 64-bits | 16-bits | 32-bits | 48-bits | 64-bits |
| CSQ | 52.0 | 60.2 | - | 63.1 | 65.4 | 73.0 | - | 75.1 | 50.1 | 62.2 | - | 67.0 |
| DBDH | 12.1 | 18.8 | 19.8 | 19.6 | 5.0 | 12.3 | 21.3 | 18.5 | 2.1 | 2.7 | 3.5 | 4.2 |
| DPN | 26.5 | 37.2 | 41.6 | 43.9 | 45.2 | 58.1 | 62.2 | 63.4 | 30.6 | 44.0 | 49.4 | 53.0 |
| DPSH | 10.7 | 19.2 | 24.3 | 27.3 | 1.3 | 16.0 | 22.2 | 27.2 | 1.1 | 7.0 | 10.0 | 12.7 |
| HashNet | 14.0 | 23.2 | 27.0 | 29.6 | 14.9 | 29.9 | 37.7 | 41.6 | 7.3 | 13.6 | 19.0 | 21.4 |
| GreedyHash | 22.6 | 31.1 | 34.9 | 36.0 | 34.8 | 50.4 | 54.1 | 55.2 | 17.0 | 29.3 | 34.8 | 37.9 |
| DCH | 21.2 | 25.9 | 23.9 | 22.8 | 32.7 | 38.9 | 36.7 | 35.2 | 15.9 | 20.8 | 20.9 | 19.8 |
| DFH | 8.0 | 15.7 | 18.8 | 23.8 | 3.6 | 12.4 | 17.8 | 20.1 | 1.2 | 6.5 | 8.9 | 10.2 |
| DSHSD | 20.7 | 27.9 | 29.3 | 30.5 | 35.5 | 46.6 | 49.9 | 50.8 | 17.0 | 24.1 | 25.5 | 26.1 |
| IDHN | 15.7 | 21.8 | 27.5 | 30.6 | 13.8 | 19.6 | 25.1 | 29.1 | 3.0 | 10.1 | 13.5 | 15.7 |
| QSMIH | 19.7 | 23.2 | 25.6 | 27.1 | 19.9 | 26.7 | 28.2 | 31.3 | 9.4 | 14.5 | 16.0 | 17.1 |
| MSViT | 22.2 | 25.4 | 26.3 | 26.7 | 19.3 | 22.1 | 24.3 | 25.4 | 9.4 | 11.5 | 12.2 | 12.9 |
| HybridHash | 35.1 | 45.2 | 50.9 | 53.6 | 34.0 | 45.9 | 51.6 | 56.7 | 21.9 | 33.7 | 39.8 | 42.9 |
| PTLCH | 54.8 | 69.3 | 71.2 | 72.9 | 67.3 | 78.5 | **81.4** | **81.6** | 47.2 | 66.0 | 71.3 | 73.7 |
| FoodHash | **72.9** | **76.0** | **76.4** | **77.4** | **75.3** | **79.7** | 80.8 | 80.4 | **64.7** | **72.0** | **73.7** | **74.3** |

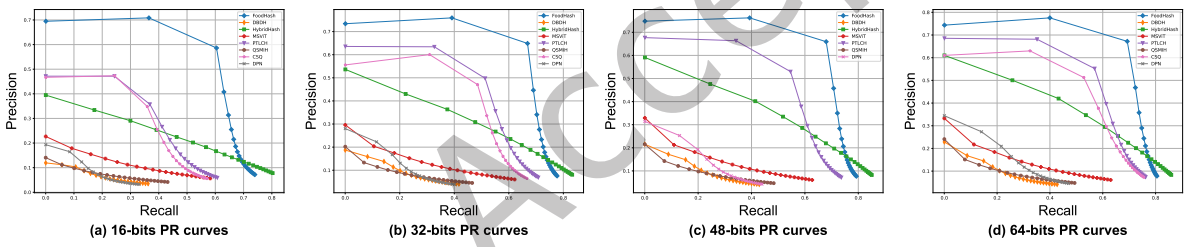


Fig. 4. PR curves on the ETH Food-101 dataset for hash codes lengths of 16-bits, 32-bits, 48-bits and 64-bits.

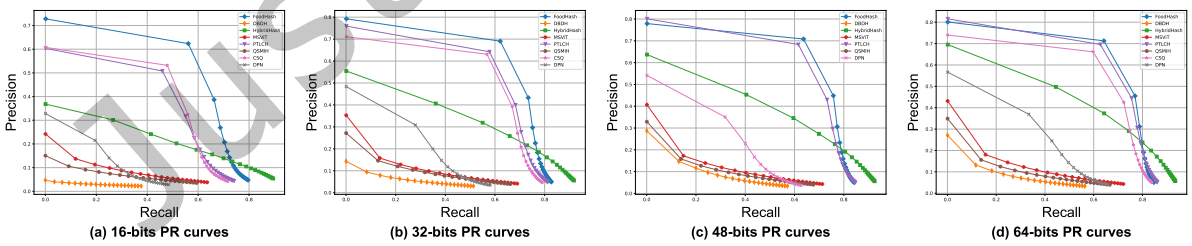


Fig. 5. PR curves on the Vireo Food-172 dataset for hash codes lengths of 16-bits, 32-bits, 48-bits and 64-bits.

To further verify the generalization performance of the proposed method on large-scale datasets, we conducted experiments on the ISIA Food-500 dataset using the same partitioning strategy as in ETH Food-101. The ISIA Food-500 dataset contains 500 food categories with approximately 500,000 images, making it one of the largest

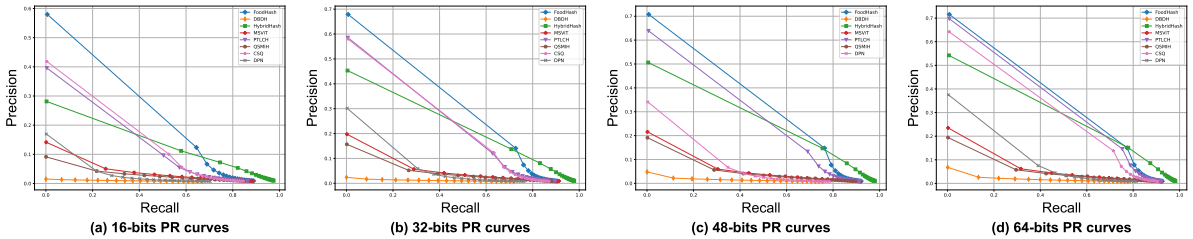


Fig. 6. PR curves on the UEC Food-256 dataset for hash codes lengths of 16-bits, 32-bits, 48-bits and 64-bits.

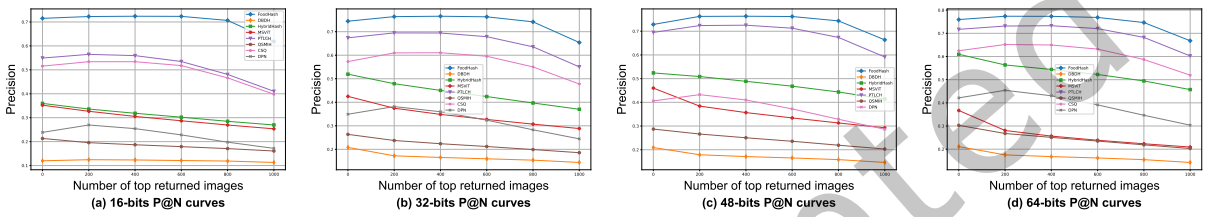


Fig. 7. P@N curves on the ETH Food-101 dataset for hash codes lengths of 16-bits, 32-bits, 48-bits and 64-bits.

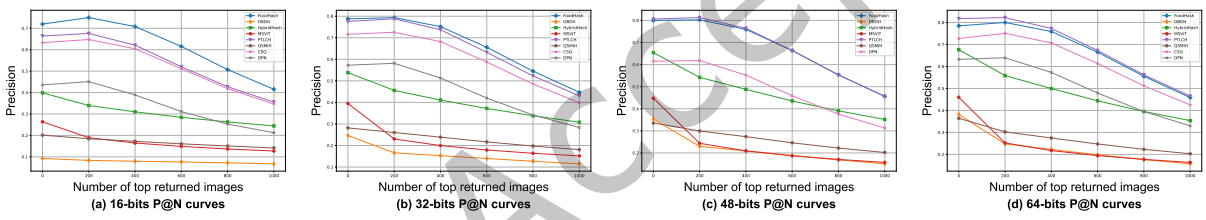


Fig. 8. P@N curves on the Vireo Food-172 dataset for hash codes lengths of 16-bits, 32-bits, 48-bits and 64-bits.

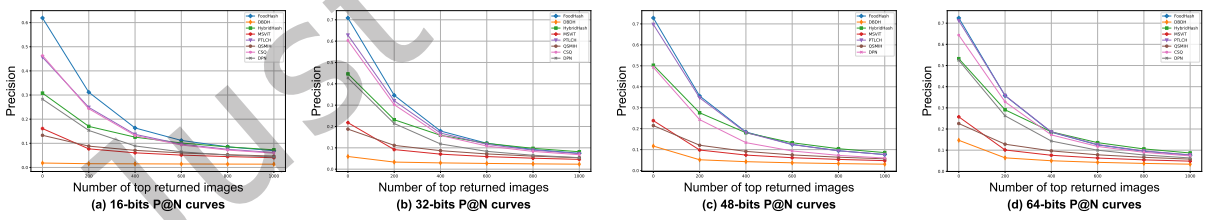


Fig. 9. P@N curves on the UEC Food-256 dataset for hash codes lengths of 16-bits, 32-bits, 48-bits and 64-bits.



and most challenging datasets in the field of food image retrieval. The experimental results are reported in Table 3. As shown in the table, FoodHash consistently achieves the best performance across different hash code lengths, significantly outperforming existing architectures. These results demonstrate that FoodHash not only effectively captures fine-grained details in food images but also fully models their semantic information, thereby exhibiting stronger discriminative ability and generalization capability in large-scale retrieval tasks.

Table 3. MAP scores (%) of different methods compared on the ISIA Food-500 dataset.

| Dataset | Methods | 16-bits | 32-bits | 48-bits | 64-bits |
|---|---|---|---|---|---|
| ISIA Food-500 | CSQ | 18.0 | 31.3 | - | 36.1 |
| | DBDH | 0.7 | 0.8 | 0.8 | 0.8 |
| | DPN | 8.8 | 17.5 | 22.0 | 24.8 |
| | QSMIH | 2.3 | 3.6 | 4.4 | 4.9 |
| | MSViT | 6.4 | 8.4 | 9.0 | 9.5 |
| | HybridHash | 12.4 | 18.9 | 23.2 | 25.2 |
| | PTLCH | 18.9 | 34.4 | 42.3 | 45.2 |
| | FoodHash | **33.1** | **42.2** | **45.0** | **45.5** |

## 4.4 Ablation Study

We use the DPN with ViT-B_16 as the feature extractor as the baseline and design the ablation experiments by modifying the network structure and adding a novel loss function, as shown in Table 4. The table shows that when the AIP module is added to the feature extraction network and fused using the CFM, compared to the baseline DPN, on the ETH Food-101 dataset, the mAP of the 16-bits, 32-bits, 48-bits and 64-bits hash codes of FoodHash increased by 0.6%, 0.9%, 0.6% and 0.5%, respectively. On the Vireo Food-172 dataset, the mAP of the 16-bits, 32-bits, 48-bits and 64-bits hash codes of FoodHash increased by 0.3%, 1.6%, 0.7% and 1.2%, respectively. On the UEC Food-256 dataset, the mAP of the 16-bits and 64-bits hash codes of FoodHash increased by 1.4% and 1.3%, respectively. The results indicate that the AIP module can extract different detailed features in food images, and the information exchange of different kinds of features can be enhanced by CFM, which enhances the fusion between different features so that the model can learn more features in the image and improve accuracy. With the addition of the novel loss function, compared to the baseline DPN, on the ETH Food-101 dataset, the mAP of the 16-bits, 32-bits, 48-bits and 64-bits hash codes of FoodHash increased by 1.3%, 1.2%, 0.8% and 1.2%, respectively. On the Vireo Food-172 dataset, the mAP of the 16-bits, 32-bits, 48-bits and 64-bits hash codes of FoodHash increased by 0.6%, 1.2%, 1.1% and 0.9%, respectively. On the UEC Food-256 dataset, the mAP of the 16-bits, 32-bits, 48-bits and 64-bits hash codes of FoodHash increased by 1.2%, 0.5%, 0.3% and 0.9%, respectively. This demonstrates that the novel loss function can optimize the network parameters during the hash learning process, allowing the network to learn more accurate hash codes.

Table 5 shows the experimental results on the ETH Food-101, Vireo Food-172 and UEC Food-256 datasets using different models as the feature extraction backbone network. The table compares the performance of AlexNet, ResNet, ViT-B_32, ViT-B_16 and the proposed FoodHash framework under different hash code lengths (16-bits, 32-bits, 48-bits and 64-bits). From the table, it can be observed that replacing AlexNet with ResNet or the ViT series significantly improves retrieval performance, which indicates that more powerful networks can learn the details and semantic features of food images more effectively. Among all the compared methods, FoodHash consistently achieves the best performance on the three datasets, and its performance advantage becomes more evident as the length of the hash codes increases, which indicates that FoodHash can fully utilize the details and semantic information of food images under various hash code lengths and dataset conditions and achieve better retrieval performance than traditional models and other Transformer-based architectures.

In this paper, the hyperparameter $\omega$ is used to adjust the importance of two types of feature interactions. Fig. 10 illustrates the trend of mAP with respect to the hyperparameter $\omega$ under different hash code lengths on the three food image datasets. It can be observed that as $\omega$ increases, the mAP values on all three datasets show an upward trend, with the most significant improvement achieved at 16-bits. The reason for the decrease is due to

Table 4. MAP scores (%) of our method on ablation experiment across three food datasets.

| Datasets | Ablation | 16-bits | 32-bits | 48-bits | 64-bits |
|---|---|---|---|---|---|
| ETH Food-101 | ViT-B_16 | 71.6 | 74.8 | 75.6 | 76.2 |
| | w/o $\mathcal{L}_S$ | 72.2 | 75.7 | 76.2 | 76.7 |
| | w/o AIP | 72.3 | 74.3 | 75.5 | 75.5 |
| | FoodHash | **72.9** | **76.0** | **76.4** | **77.4** |
| Vireo Food-172 | ViT-B_16 | 74.7 | 78.5 | 79.7 | 79.5 |
| | w/o $\mathcal{L}_S$ | 75.0 | **80.1** | 80.4 | **80.7** |
| | w/o AIP | 74.2 | 79.0 | 79.7 | 79.8 |
| | FoodHash | **75.3** | 79.7 | **80.8** | 80.4 |
| UEC Food-256 | ViT-B_16 | 63.5 | 71.5 | 73.4 | 73.4 |
| | w/o $\mathcal{L}_S$ | **64.9** | 71.6 | 73.4 | **74.7** |
| | w/o AIP | 63.6 | 71.8 | 72.9 | 73.8 |
| | FoodHash | 64.7 | **72.0** | **73.7** | 74.3 |

Table 5. MAP scores (%) of our method on ablation comparison experiments with different backbones across three food datasets.

| Datasets | Backbone | 16-bits | 32-bits | 48-bits | 64-bits |
|---|---|---|---|---|---|
| ETH Food-101 | AlexNet | 25.6 | 36.7 | 41.4 | 44.4 |
| | ResNet | 48.8 | 58.0 | 61.1 | 60.9 |
| | ViT-B_32 | 62.0 | 66.6 | 67.5 | 68.4 |
| | ViT-B_16 | 70.6 | 74.8 | 75.9 | 76.3 |
| | FoodHash | **72.9** | **76.0** | **76.4** | **77.4** |
| Vireo Food-172 | AlexNet | 44.8 | 58.4 | 61.6 | 63.6 |
| | ResNet | 64.4 | 71.6 | 73.4 | 74.2 |
| | ViT-B_32 | 65.9 | 72.0 | 72.6 | 73.1 |
| | ViT-B_16 | 74.6 | 78.7 | 79.5 | 79.6 |
| | FoodHash | **75.3** | **79.7** | **80.8** | **80.4** |
| UEC Food-256 | AlexNet | 30.4 | 43.8 | 49.2 | 52.8 |
| | ResNet | 48.0 | 60.7 | 65.1 | 65.9 |
| | ViT-B_32 | 57.0 | 65.5 | 67.8 | 68.9 |
| | ViT-B_16 | 63.7 | 71.5 | 72.3 | 72.8 |
| | FoodHash | **64.7** | **72.0** | **73.7** | **74.3** |



the shorter length of the 16-bits hash codes, which results in a limited encoding space and higher requirements for feature expression. Under these conditions, the imbalance in feature interaction weights is more likely to hinder effective communication between local and global features, making it challenging to capture the fine features in the images accurately and affecting the final retrieval performance. In contrast, when adopting longer hash code lengths (32-bits, 48-bits, and 64-bits), the model exhibits greater robustness to imbalanced feature weights, resulting in a relatively smaller increase in performance.

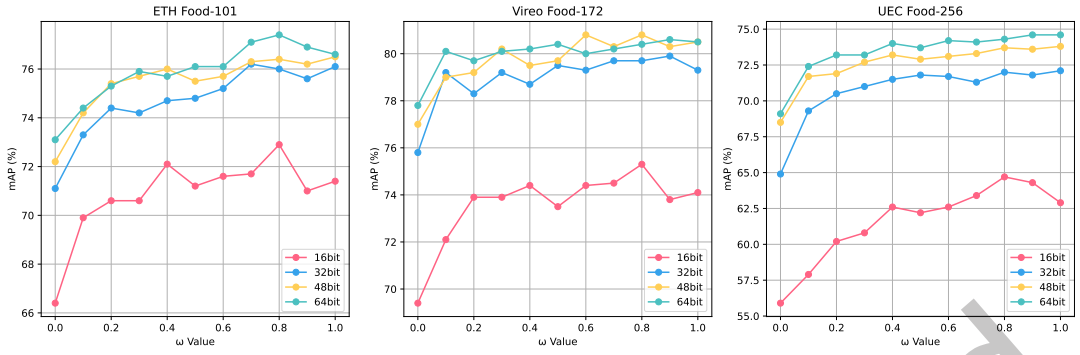


Fig. 10. The $\omega$-value curves across the three datasets.

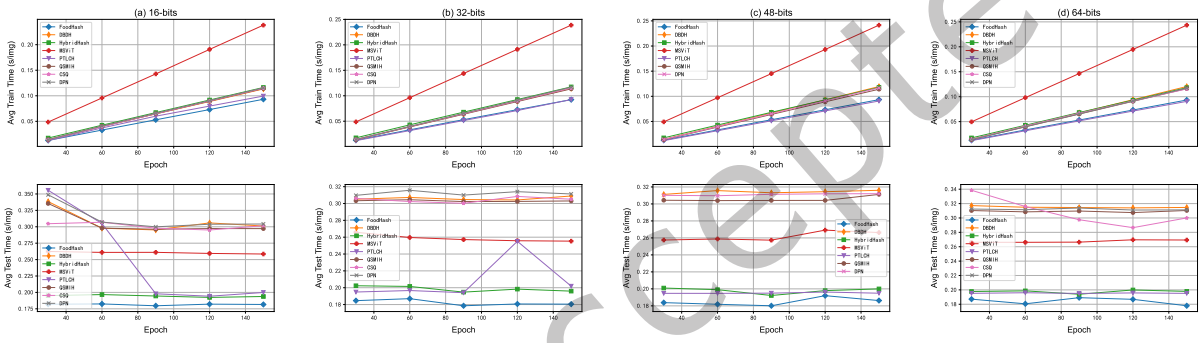


Fig. 11. The curves of average training time and average retrieval time on the ETH Food-101 dataset.

Fig. 11 and Fig. 12 illustrate FoodHash's time performance and memory usage across different hash code lengths on the ETH Food-101 dataset. In terms of time performance, FoodHash demonstrates significantly lower running times during both training and inference compared to most baseline models, indicating its high efficiency. Regarding memory usage, FoodHash consumes more memory, primarily due to its pure ViT-based architecture. The self-attention mechanism and large number of parameters in ViT result in higher memory requirements, placing FoodHash at a disadvantage compared to other models in this aspect. Future improvements aim to reduce memory usage through model compression and optimized attention mechanisms while maintaining high efficiency.

## 4.5 Visualization Results

To further validate the performance of our proposed method, we provide the t-SNE visualization of the hash code and the visualization of the top-8 retrieval results.

*4.5.1 t-SNE Visualization of Hash Codes.* Fig. 13 shows the t-SNE visualization results of our method and seven other deep hashing learning methods on the ETH Food-101 dataset with 64-bits hash codes. As can be seen from the figure, the hash codes generated by FoodHash exhibit a clearly distinguishable structure, with higher separation between hash codes of different categories and tighter clustering of hash codes within the same category, which better achieves the goal of large inter-class distances and small intra-class distances. The

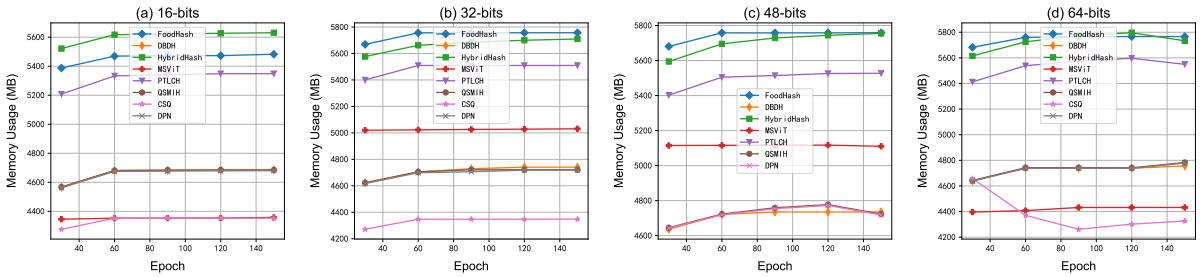


Fig. 12. The memory usage curves on the ETH Food-101 dataset.

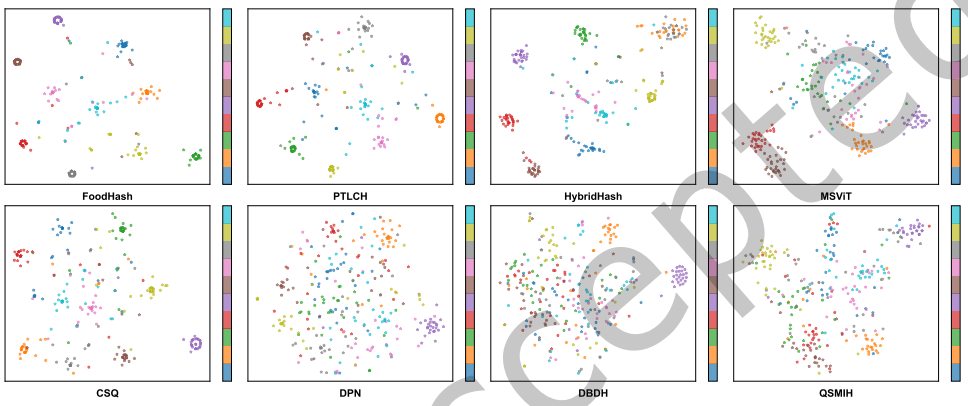


Fig. 13. Visualization of 64-bits hash codes generated by our method and seven other deep hashing learning methods, trained on the ETH Food-101 dataset using ten groups of data points (one group per class).



competitive PTLCH method also performs well, with points of 10 different colors forming 6 clusters. However, some noise points (points of different colors appearing within the same cluster) and outliers (points located outside all clusters) can still be observed in the visualization of PTLCH. The noisy points may result from the difficulty of separating data of different colors when mapping from high-dimensional space to low-dimensional space. Outliers may arise due to significant differences between specific high-dimensional data points and others, which makes them difficult to map into any of the clusters. In contrast, our method significantly reduces noise points and outliers within clusters and provides more explicit boundaries between clusters, which is highly advantageous for classification tasks. However, for application scenarios where a large number of retrieval results need to be returned, forcibly clustering these noise and outlier points may introduce bias in the visualization and retrieval ranking, exposing the limitations of our method in handling a small number of extreme samples.

*4.5.2 Visualization of the Top-8 Retrieval Results.* To further validate the effectiveness of the proposed method on food datasets, Fig. 14 presents the retrieval results using 64-bits hash codes on the ETH Food-101 dataset for eight randomly selected query images and their top-8 retrieved results. The first column in the figure represents the input query images, while the second to the ninth columns display the images returned from the retrieval. Blue checkmarks are used to mark correct retrieval results and red checkmarks are used to mark incorrect retrieval results. From the figure, it can be observed that the vast majority of the returned results in the top-8 retrieval results are consistent with the query image category, which reflects the effective modeling and recognition

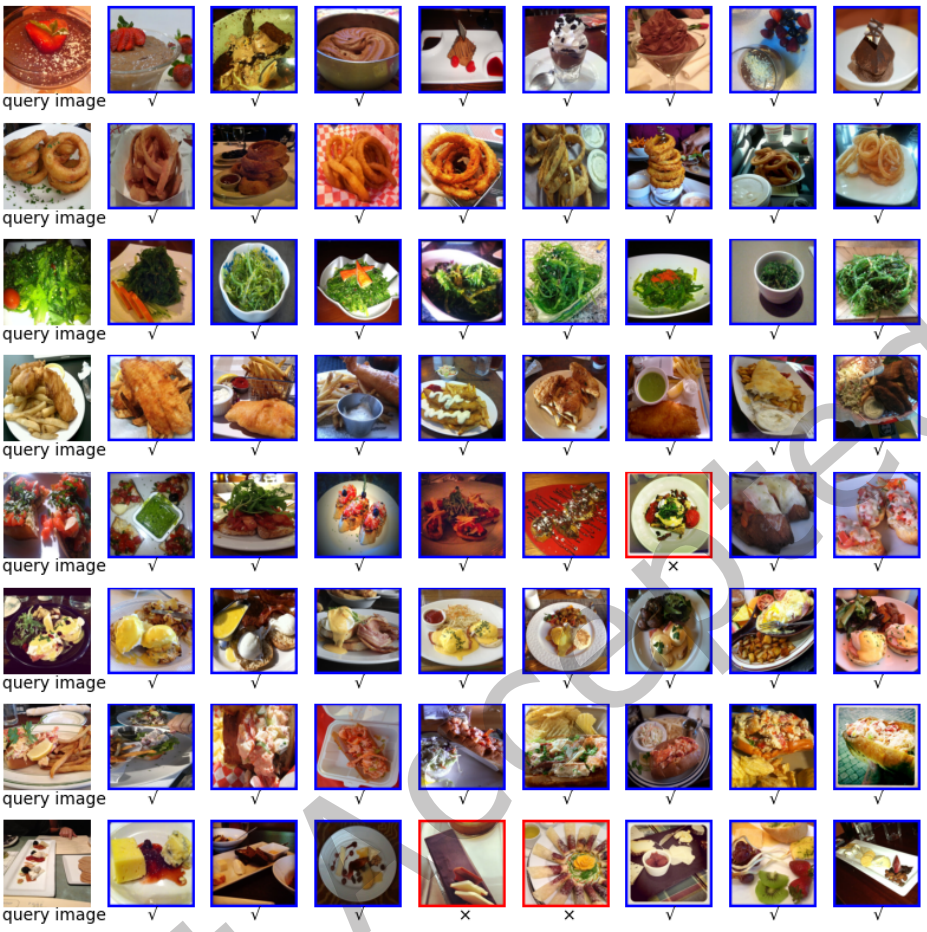


Fig. 14. Top-8 retrieval results from the ETH Food-101 dataset by FoodHash with 64-bits hash codes.



capability of our proposed method for the complex details of food images. Occasional retrieval errors (e.g., the red marks in the fifth and eighth rows) indicate that the model may experience difficulty in distinguishing between food images with similar appearances but distinct categories, potentially due to its inability to fully capture the unique features of these images. Additionally, visually complex or cluttered backgrounds in certain food images may interfere with the feature extraction process, leading to retrieval errors. While the proposed method demonstrates promising performance on the existing dataset, its scalability and ability to generalize across diverse, real-world datasets remain areas for further investigation. In summary, the proposed method yields accurate and stable results in food image retrieval tasks, highlighting its feasibility and potential advantages for practical applications. Nevertheless, addressing the identified limitations could further improve its robustness and adaptability.

## 5 Conclusion

This paper proposes a novel context-aware proxy interaction and fusion deep hashing method for food image retrieval, called FoodHash. The core of the method is the AIP module, which integrates three key designs: 1) Aggregation: local information from neighboring patch tokens is integrated to generate a set of proxy patch tokens, enhancing the ability of model to capture local features; 2) Interaction: global self-attention is introduced between proxy patch tokens; 3) Propagation: the global contextual information obtained by the proxy patch tokens is propagated to their corresponding sub-patch tokens to enable efficient fusion and recovery of multi-scale features. In addition, the proposed Cross-Fusion Module computes attention weights through scaled dot-product operations followed by weighted summation, thereby generating richer visual feature representations for the food image retrieval task. Building on this, the newly designed loss function better optimizes the intra-class compactness and inter-class separability of the hash codes. We conducted extensive experiments on three publicly available food datasets and compared our method with several retrieval methods based on CNNs and Transformers, validating the effectiveness of our approach.

Future food image retrieval research will face significant challenges and new development opportunities. Considering the multimodal nature of food information in real-world scenarios, such as images, textual descriptions, recipes, and nutritional content, exploring cross-modal retrieval tasks could enhance the system's practicality and generalization by integrating multimodal information. With the advancement of large-scale pre-trained models, incorporating these models into food image retrieval frameworks could further improve retrieval performance by leveraging their strengths in multimodal alignment and semantic feature extraction. Additionally, the knowledge transfer capabilities of large models can help improve recognition and retrieval performance for few-shot food categories under long-tail distributions. On the other hand, with the emergence of larger and more comprehensive food datasets (e.g., Food2K), exploring weakly supervised or semi-supervised learning methods will become a significant research direction. These approaches can reduce reliance on labeled data while fully leveraging the potential of massive amounts of unlabeled data, thereby enhancing the model's generalization capability and robustness. Advancing these research directions will enable food retrieval technologies to provide more substantial support for food computing and health management applications.



## References


[1] Amina Belalia, Kamel Belloulata, and Adil Redaoui. 2025. Enhanced Image Retrieval Using Multiscale Deep Feature Fusion in Supervised Hashing. *Journal of Imaging* 11, 1 (2025), 20.

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. 2014. Food-101–mining discriminative components with random forests. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*. Springer, 446–461.

[3] Yue Cao, Mingsheng Long, Bin Liu, and Jianmin Wang. 2018. Deep cauchy hashing for hamming space retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1229–1237.

[4] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Philip S Yu. 2017. Hashnet: Deep learning to hash by continuation. In *Proceedings of the IEEE international conference on computer vision*. 5608–5617.

[5] Zijian Chao, Shuli Cheng, and Yongming Li. 2023. Deep internally connected transformer hashing for image retrieval. *Knowledge-Based Systems* 279 (2023), 110953.

[6] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L Yuille, and Yuyin Zhou. 2021. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306* (2021).

[7] Jingjing Chen and Chong-Wah Ngo. 2016. Deep-based ingredient recognition for cooking recipe retrieval. In *Proceedings of the 24th ACM international conference on Multimedia*. 32–41.

[8] Yongbiao Chen, Sheng Zhang, Fangxin Liu, Zhigang Chang, Mang Ye, and Zhengwei Qi. 2022. Transhash: Transformer-based hamming hashing for efficient image retrieval. In *Proceedings of the 2022 international conference on multimedia retrieval*. 127–136.

[9] Alexey Dosovitskiy. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[10] Guoyong Duan, Jing Yang, and Yilong Yang. 2011. Content-based image retrieval research. *Physics Procedia* 22 (2011), 471–477.

[11] Shiv Ram Dubey, Satish Kumar Singh, and Wei-Ta Chu. 2022. Vision transformer hashing for image retrieval. In *2022 IEEE international conference on multimedia and expo (ICME)*. IEEE, 1–6.
[12] Lixin Fan, Kam Woh Ng, Ce Ju, Tianyu Zhang, and Chee Seng Chan. 2020. Deep Polarized Network for Supervised Learning of Accurate Binary Hashing Codes.. In *IJCAI*. 825–831.
[13] Giovanni Maria Farinella, Dario Allegra, Marco Moltisanti, Filippo Stanco, and Sebastiano Battiato. 2016. Retrieval and classification of food images. *Computers in biology and medicine* 77 (2016), 23–39.
[14] Jianyuan Guo, Kai Han, Han Wu, Yehui Tang, Xinghao Chen, Yunhe Wang, and Chang Xu. 2022. Cmt: Convolutional neural networks meet vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 12175–12185.
[15] Yuchen Guo, Guiguang Ding, Li Liu, Jungong Han, and Ling Shao. 2017. Learning to hash with optimized anchor embedding for scalable retrieval. *IEEE Transactions on Image Processing* 26, 3 (2017), 1344–1354.
[16] Dongchen Han, Xuran Pan, Yizeng Han, Shiji Song, and Gao Huang. 2023. Flatten transformer: Vision transformer using focused linear attention. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5961–5971.
[17] Ali Hatamizadeh, Greg Heinrich, Hongxu Yin, Andrew Tao, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. 2023. Fastervit: Fast vision transformers with hierarchical attention. *arXiv preprint arXiv:2306.06189* (2023).
[18] Chao He and Hongxi Wei. 2024. HybridHash: Hybrid Convolutional and Self-Attention Deep Hashing for Image Retrieval. In *Proceedings of the 2024 International Conference on Multimedia Retrieval*. 824–832.
[19] Akihisa Ishino, Yoko Yamakata, Hiroaki Karasawa, and Kiyoharu Aizawa. 2021. RecipeLog: Recipe Authoring App for Accurate Food Recording. In *Proceedings of the 29th ACM International Conference on Multimedia*. 2798–2800.
[20] Qingyuan Jiang and Wujun Li. 2018. Asymmetric deep supervised hashing. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.
[21] Lu Jin, Zechao Li, Yonghua Pan, and Jinhui Tang. 2023. Relational Consistency Induced Self-Supervised Hashing for Image Retrieval. *IEEE Transactions on Neural Networks and Learning Systems* (2023).
[22] Rong Kang, Yue Cao, Mingsheng Long, Jianmin Wang, and Philip S Yu. 2019. Maximum-margin hamming hashing. In *Proceedings of the IEEE/CVF international conference on computer vision*. 8252–8261.
[23] Yoshiyuki Kawano and Keiji Yanai. 2015. Automatic expansion of a food image dataset leveraging existing categories with domain adaptation. In *Computer Vision-ECCV 2014 Workshops: Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part III 13*. Springer, 3–17.
[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2017. ImageNet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (2017), 84–90.
[25] Tao Li, Zheng Zhang, Lishen Pei, and Yan Gan. 2022. HashFormer: Vision transformer based deep hashing for image retrieval. *IEEE Signal Processing Letters* 29 (2022), 827–831.
[26] Wujun Li, Sheng Wang, and Wangcheng Kang. 2015. Feature learning based deep supervised hashing with pairwise labels. *arXiv preprint arXiv:1511.03855* (2015).
[27] Xue Li, Jiong Yu, Shaochen Jiang, Hongchun Lu, and Ziyang Li. 2023. Msvit: training multiscale vision transformers for image retrieval. *IEEE Transactions on Multimedia* (2023).
[28] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. 2022. Exploring plain vision transformer backbones for object detection. In *European conference on computer vision*. Springer, 280–296.
[29] Yunqiang Li, Wenjie Pei, Jan van Gemert, et al. 2019. Push for quantization: Deep fisher hashing. *arXiv preprint arXiv:1909.00206* (2019).
[30] Yunqiang Li and Jan van Gemert. 2021. Deep unsupervised image hashing by maximizing bit entropy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 2002–2010.
[31] Kevin Lin, Huei-Fang Yang, Jen-Hao Hsiao, and Chu-Song Chen. 2015. Deep learning of binary hash codes for fast image retrieval. In *2015 IEEE conference on computer vision and pattern recognition workshops (CVPRW)*. IEEE, 27–35.
[32] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. 2016. Deep supervised hashing for fast image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2064–2072.
[33] Huawen Liu, Wenhua Zhou, Hong Zhang, Gang Li, Shichao Zhang, and Xuelong Li. 2023. Bit reduction for locality-sensitive hashing. *IEEE Transactions on Neural Networks and Learning Systems* (2023).
[34] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang. 2012. Supervised hashing with kernels. In *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2074–2081.
[35] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*. 10012–10022.
[36] Xiao Luo, Haixin Wang, Daqing Wu, Chong Chen, Minghua Deng, Jianqiang Huang, and Xian-Sheng Hua. 2023. A survey on deep hashing methods. *ACM Transactions on Knowledge Discovery from Data* 17, 1 (2023), 1–50.
[37] Omid Nejati Manzari, Hamid Ahmadabadi, Hossein Kashiani, Shahriar B Shokouhi, and Ahmad Ayatollahi. 2023. MedViT: a robust vision transformer for generalized medical image classification. *Computers in Biology and Medicine* 157 (2023), 106791.

[38] Weiqing Min, Shuqiang Jiang, Linhu Liu, Yong Rui, and Ramesh Jain. 2019. A survey on food computing. *ACM Computing Surveys (CSUR)* 52, 5 (2019), 1–36.
[39] Weiqing Min, Linhu Liu, Zhiling Wang, Zhengdong Luo, Xiaoming Wei, Xiaolin Wei, and Shuqiang Jiang. 2020. ISIA Food-500: A Dataset for Large-Scale Food Recognition via Stacked Global-Local Attention Network. *Proceedings of the 28th ACM International Conference on Multimedia* (2020), 393–401.
[40] Weiqing Min, Zhiling Wang, Yuxin Liu, Mengjiang Luo, Liping Kang, Xiaoming Wei, Xiaolin Wei, and Shuqiang Jiang. 2023. Large scale visual food recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 8 (2023), 9932–9949.
[41] Junting Pan, Adrian Bulat, Fuwen Tan, Xiatian Zhu, Lukasz Dudziak, Hongsheng Li, Georgios Tzimiropoulos, and Brais Martinez. 2022. Edgevits: Competing light-weight cnns on mobile devices with vision transformers. In *European Conference on Computer Vision*. Springer, 294–311.
[42] Nikolaos Passalis and Anastasios Tefas. 2021. Deep supervised hashing using quadratic spherical mutual information for efficient image retrieval. *Signal Processing: Image Communication* 93 (2021), 116146.
[43] Adil Redaoui and Kamel Belloulata. 2023. Deep Feature Pyramid Hashing for Efficient Image Retrieval. *Information* 14, 1 (2023), 6.
[44] Huan Ren, Jiangtao Guo, Shuli Cheng, and Yongming Li. 2024. Pooling-based Visual Transformer with low complexity attention hashing for image retrieval. *Expert Systems with Applications* 241 (2024), 122745.
[45] Xiuxiu Ren, Xiangwei Zheng, Huiyu Zhou, Weilong Liu, and Xiao Dong. 2022. Contrastive hashing with vision transformer for image retrieval. *International Journal of Intelligent Systems* 37, 12 (2022), 12192–12211.
[46] Ali Rostami, Nitish Nagesh, Amir Rahmani, and Ramesh Jain. 2022. World food atlas for food navigation. In *Proceedings of the 7th International Workshop on Multimedia Assisted Dietary Management*. 39–47.
[47] Avantika Singh and Shaifu Gupta. 2022. Learning to hash: a comprehensive survey of deep learning-based hashing methods. *Knowledge and Information Systems* 64, 10 (2022), 2565–2597.
[48] Jiajun Song, Zhuo Li, Weiqing Min, and Shuqiang Jiang. 2023. Towards food image retrieval via generalization-oriented sampling and loss function design. *ACM Transactions on Multimedia Computing, Communications and Applications* 20, 1 (2023), 1–19.
[49] Jiajun Song, Weiqing Min, Yuxin Liu, Zhuo Li, Shuqiang Jiang, and Yong Rui. 2022. A noise-robust locality transformer for fine-grained food image retrieval. In *2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 348–353.
[50] Shupeng Su, Chao Zhang, Kai Han, and Yonghong Tian. 2018. Greedy hash: Towards fast optimization for accurate hash coding in cnn. *Advances in neural information processing systems* 31 (2018).
[51] A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* (2017).
[52] Yair Weiss, Antonio Torralba, and Rob Fergus. 2008. Spectral hashing. *Advances in neural information processing systems* 21 (2008).
[53] Lei Wu, Hefei Ling, Ping Li, Jiazhong Chen, Yang Fang, and Fuhao Zhou. 2019. Deep supervised hashing based on stable distribution. *IEEE Access* 7 (2019), 36489–36499.
[54] Chengyin Xu, Zenghao Chai, Zhengzhuo Xu, Hongjia Li, Qiruyi Zuo, Lingyu Yang, and Chun Yuan. 2022. HHF: Hashing-guided hinge function for deep hashing retrieval. *IEEE Transactions on Multimedia* 25 (2022), 7428–7440.
[55] Yoko Yamakata, Akihisa Ishino, Akiko Sunto, Sosuke Amano, and Kiyoharu Aizawa. 2022. Recipe-oriented food logging for nutritional management. In *Proceedings of the 30th ACM International Conference on Multimedia*. 6898–6904.
[56] Wenjing Yang, Liejun Wang, and Shuli Cheng. 2022. Deep parameter-free attention hashing for image retrieval. *Scientific Reports* 12, 1 (2022), 7082.
[57] Tao Yao, Gang Wang, Lianshan Yan, Xiangwei Kong, Qingtang Su, Caiming Zhang, and Qi Tian. 2019. Online latent semantic hashing for cross-media retrieval. *Pattern Recognition* 89 (2019), 1–11.
[58] Li Yuan, Tao Wang, Xiaopeng Zhang, Francis EH Tay, Zequn Jie, Wei Liu, and Jiashi Feng. 2020. Central similarity quantization for efficient image and video retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 3083–3092.
[59] Dell Zhang, Jun Wang, Deng Cai, and Jinsong Lu. 2010. Self-taught hashing for fast similarity search. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. 18–25.
[60] Zheng Zhang, Qin Zou, Yuewei Lin, Long Chen, and Song Wang. 2019. Improved deep hashing with soft pairwise similarity for multi-label image retrieval. *IEEE Transactions on Multimedia* 22, 2 (2019), 540–553.
[61] Xiangtao Zheng, Yichao Zhang, and Xiaoqiang Lu. 2020. Deep balanced discrete hashing for image retrieval. *Neurocomputing* 403 (2020), 224–236.
[62] Lei Zhu, Xinjiang Wang, Zhanghan Ke, Wayne Zhang, and Rynson WH Lau. 2023. Biformer: Vision transformer with bi-level routing attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10323–10333.

Received 20 February 2025; revised 16 September 2025; accepted 8 December 2025