International Journal of Computer Vision (2024) 132:2765–2797
https://doi.org/10.1007/s11263-024-02009-7

Hugs Bring Double Benefits: Unsupervised Cross-Modal Hashing with Multi-granularity Aligned Transformers

Jinpeng Wang[1,3] · Ziyun Zeng[1,3] · Bin Chen[2,3] · Yuting Wang[1,3] · Dongliang Liao[4] · Gongfu Li[4] · Yiru Wang[4] · Shu-Tao Xia[1,3]

Received: 13 April 2023 / Accepted: 16 January 2024 / Published online: 18 February 2024

Abstract

Unsupervised cross-modal hashing (UCMH) has been commonly explored to support large-scale cross-modal retrieval of unlabeled data. Despite promising progress, most existing approaches are developed on convolutional neural network and multilayer perceptron architectures, sacrificing the quality of hash codes due to limited capacity for excavating multi-modal semantics. To pursue better content understanding, we break this convention for UCMH and delve into a transformer-based paradigm. Unlike naïve adaptations via backbone substitution that overlook the heterogeneous semantics from transformers, we propose a multi-granularity learning framework called **hugging** to bridge the modality gap. Specifically, we first construct a fine-grained semantic space composed of a series of aggregated local embeddings that capture implicit attribute-level semantics. In the hash learning stage, we innovatively incorporate fine-grained alignment with these local embeddings to enhance global hash code alignment. Notably, this fine-grained alignment only facilitates robust cross-modal learning without complicating global hash code generation at test time, thus fully maintaining the high efficiency of hash-based retrieval. To make the most of fine-grained information, we further propose a differentiable optimized quantization algorithm and extend our framework to **hugging**$^+$. This variant neatly integrates quantization learning into the fine-grained alignment during training, producing quantization codes of local embeddings as a gift at test time, which can augment the retrieval performance through an efficient reranking stage. We instantiate simple baselines with contrastive learning objectives for hugging and hugging$^+$, namely HUGGINGHASH and HUGGINGHASH$^+$. Extensive experiments on 4 text-image retrieval and 2 text-video retrieval benchmark datasets show the competitive performance of HUGGINGHASH and HUGGINGHASH$^+$ against state-of-the-art baselines. More encouragingly, we also validate that hugging and hugging$^+$ are flexible and effective across various baselines, suggesting their universal applicability in the realm of UCMH.

Keywords Unsupervised cross-modal hashing (UCMH) · Transformers · Image retrieval · Video retrieval · Vector of locally aggregated descriptors (VLAD) · Optimized product quantization (OPQ)

Communicated by Guang Yang.

✉ Bin Chen
chenbin2021@hit.edu.cn

Jinpeng Wang
wjp20@mails.tsinghua.edu.cn

Ziyun Zeng
zengzy21@mails.tsinghua.edu.cn

Yuting Wang
huangmozhi9527@gmail.com

Dongliang Liao
brightliao@tencent.com

Gongfu Li
gongfuli@tencent.com

Yiru Wang
dorisyrwang@tencent.com

Shu-Tao Xia
xiast@sz.tsinghua.edu.cn

[1] Shenzhen International Graduate School, Tsinghua University, Shenzhen, China

[2] Harbin Institute of Technology, Shenzhen, China

[3] Research Center of Artificial Intelligence, Peng Cheng Laboratory, Shenzhen, China

[4] Wechat Group, Tencent Inc., Shenzhen, China

1 Introduction

Cross-modal retrieval aims to retrieve relevant items from one modality using queries from another modality, e.g. to retrieve images by text, which plays an essential role in multimedia search and recommendation. Ever-increasing multi-modal data raises the concern about search efficiency. Hashing has become a popular indexing solution because Hamming descriptors can significantly reduce index storage and accelerate distance computation with fast XOR operations (Li et al., 2020; Song et al., 2020; Liu et al., 2019; Zhang et al., 2023b). We can roughly categorize existing efforts on cross-modal hashing (CMH) into unsupervised and supervised methods. Unsupervised methods (Yu et al., 2021a; Su et al., 2019; Hu et al., 2020; Tu et al., 2023) exploit natural multi-modal co-occurrence for hashing. In contrast, supervised methods (Chen et al., 2021; Cao et al., 2018; Jiang and Li, 2017; Zhang et al., 2023a) can leverage full supervision, e.g. ground-truth labels, to better preserve semantic information in hash codes. While supervised methods usually show impressive performance, they are less favored in real-world applications due to the expensive annotation cost. Therefore, we pay attention to the unsupervised methodology alternatively.

The performance of unsupervised cross-modal hashing (UCMH) is inherently subject to multimedia understanding. Although deep neural networks have made remarkable progress in hashing, the advances have yet to be fully exploited. State-of-the-art approaches (Wang et al., 2020b; Hu et al., 2020; Su et al., 2019; Yu et al., 2021a; Zhu et al., 2023) mainly adopted classic convolutional neural networks (CNNs), e.g. VGGNet (Simonyan and Zisserman, 2015) and AlexNet (Krizhevsky et al., 2012), to extract visual features and used multilayer perceptrons (MLPs) to encode text information. These designs are sub-optimal to capture semantic information from visual and language modalities, and they also suffer from limited generalizability. To improve UCMH and keep pace with the development of deep learning, one promising direction is to take advantage of transformers (Vaswani et al., 2017).

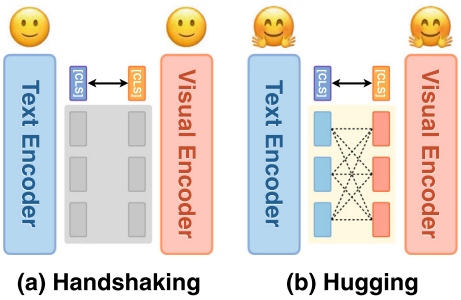

Fig. 1 Semantic alignment paradigms for transformer-based unsupervised cross-modal hashing. **a** *Handshaking* means aligning global representations of class tokens (i.e., `[CLS]`). **b** *Hugging* further exploits fine-grained alignment using content tokens. Fine-grained alignment serves as an auxiliary task to bridge the modality gap and can effectively improve global hash code generation without extra test-time overhead. Additionally, we further extend hugging to produce local quantization codes for efficient reranking, which is a bonus from fine-grained alignment

In the past few years, transformers have made inspiring successes in natural language processing (Devlin et al., 2019; Liu et al., 2019c; Yang et al., 2019) and computer vision (Dosovitskiy et al., 2021; Carion et al., 2020; Liu et al., 2021b), sparking explorations toward better content understanding and semantic extraction. Pre-trained on large-scale corpora (Krizhevsky et al., 2012; Zhu et al., 2015), transformers can serve as versatile experts that are effective and generalizable for various downstream tasks (Shin et al., 2022). We have learned about recent progress in transformer-based image (Li et al., 2022; Lu et al., 2021; Dubey et al., 2022) and video hashing (Li et al., 2021b; He et al., 2021; Wang et al., 2023) that can verify the efficacy of transformers in hashing. Nevertheless, we find transformers-based cross-modal hashing remains under-explored. Although lately (Yu et al., 2022) have proposed a CLIP-based (Radford et al., 2021) approach with impressive results, its success was mainly attributed to the off-the-shelf well-aligned transformers. Differently, in this paper, we pursue a general solution with *unaligned* transformers.

Pre-trained transformers provide solid semantic extraction for each modality, but UCMH still remains non-trivial. The main challenge is to bridge heterogeneous modalities so that the hash codes can be well aligned. Analogous to existing CNN-MLP-based UCMH models, we can train transformer-based models to produce hash codes via the global representation tokens (i.e., `[CLS]`). A simple way to learn is to align the global tokens using the objectives in existing UCMH methods. We liken this global alignment strategy to *handshaking*, as shown in Fig. 1a. In practice, handshaking is effective as expected but can be improved to reduce the modality gap. Note that transformer is a sequential architecture that arranges inputs as sequences. It naturally provides a set of content tokens (e.g. words of a text or patches of an image) with fine-grained and structural semantics, which can capture heterogeneous modality knowledge but was usually overlooked.

To enhance transformer-based UCMH learning, we present a multi-granularity alignment framework dubbed *hugging*, as illustrated in Fig. 1b. Besides global alignment using hashing representations from `[CLS]` tokens, we further develop fine-grained alignment based on the content tokens. Particularly, we construct another shared latent space with semantic structure via a GhostVLAD (Zhong et al., 2018) module. Content tokens in this space are softly aggregated into a series of parameterized clusters, each representing a latent topic or semantic concept. Locally (i.e., cluster-wise) con-

trastive (Chen et al., 2020a; He et al., 2020) alignment serves as an auxiliary objective for model training without requiring external supervision or hints (e.g. object regions in a picture or parsed components in a text). In contrast with handshaking, hugging enables synergy between global and fine-grained alignment in training, providing effective regularization to enhance the cross-modal consistency of hash codes. Interestingly, we find it improves retrieval performance and transferability, even though fine-grained parts do *not* engage in the forward process of the global part in inference. In other words, hugging will *not* bring extra inference overhead to global hash code generation. We will provide in-depth experimental analyses on the effectiveness of fine-grained alignment in Sect. 4.3.2.

Moreover, we extend the hugging framework to make the best of fine-grained representations. Note that in the above version of hugging, fine-grained alignment was discarded in inference to maintain the efficiency of hash-based retrieval. It leads to a major limitation on the usage of fine-grained representations. We left the solution as future work in our preliminary work (Wang et al., 2022b). Fortunately, we now find an effective quantization-based solution to turn fine-grained representations into profit while reconciling efficient retrieval. To be specific, for each locally aggregated cluster, we incorporate a learnable optimized quantization module to compress continuous embeddings while retaining maximal semantics. We adopt asymmetric-quantized contrastive learning to align quantized representations, achieving fine-grained alignment and quantization learning by one objective. We call this quantization-based adaptation by *hugging*$^+$ to avoid name confusion. Despite involving lossy compression with fine-grained semantics, hugging$^+$ still keeps the benefit of enhancing global hash codes. Additionally, it allows for generating fine-grained quantization codes during inference, which can be taken as another bonus to improve retrieval performance. In particular, we adopt the common practice of appending a reranking stage (ZhongZhong et al., 2017; Ye et al., 2022) to the retrieval pipeline, in which we first retrieve a moderate subset of relevant items using global hash codes and then rerank the subset according to the fine-grained similarity computed with quantized representations. We highlight two merits of hugging$^+$ regarding practical two-stage retrieval systems: (i) It enables more efficient reranking because quantization largely reduces the storage overhead for fine-grained representations and also accelerates similarity computation. (ii) It learns a holistic model to produce multi-granularity representations for different stages via just one pass, saving the effort of independent development for multiple retrieval stages. Besides keeping consistent with common practice, we choose to rerank using fine-grained representations rather than fusing multi-granularity similarity in one-stage retrieval for two reasons: (i) Decoupling global and fine-grained ranking improves retrieval efficiency as it iterates fine-grained similarity computation within a small relevant subset rather than the entire database. (ii) Two-stage retrieval is more robust than one-stage retrieval with the fusion-based strategy. Empirically, we find it sensitive to determine the fusion weights of global and fine-grained similarities for one-stage retrieval. Luckily, this weighting issue is naturally avoided in two-stage retrieval.

We instantiate hugging and hugging$^+$ strategies by building HUGGINGHASH and HUGGINGHASH$^+$ models, respectively, with a simple contrastive learning objective for global alignment. Experiments on text-image and text-video retrieval datasets show that HUGGINGHASH and HUGGINGHASH$^+$ can outperform state-of-the-art UCMH methods integrating transformers in handshaking style. Besides, we adapt several state-of-the-art approaches with hugging and hugging$^+$, demonstrating their flexibility and general effectiveness on UCMH when utilizing transformers.

Our contributions are summarized as follows.

- We highlight the significance and provide a detailed study on transformer-based UCMH, which can serve as a research basis for this promising new direction.
- Unlike straightforward ideas that only align global hash codes (i.e., *handshaking* in Fig. 1), we propose *hugging* with multi-granularity alignment for transformer-based UCMH. It shows a positive synergy for cross-modal alignment during training and improves retrieval performance and transferability of global hash codes without extra inference overhead.
- We further extend hugging to *hugging*$^+$ that combines a novel optimized quantization with fine-grained alignment. It not only keeps the benefit of improving global hash codes but also enables fine-grained quantization code generation as a gift. By reranking with fine-grained quantization codes, the inference benefits more from the hugging$^+$ training strategy while still enjoying high retrieval efficiency.
- We conduct extensive experiments on text-image and text-video retrieval datasets, showing that hugging and hugging$^+$ help to outperform state-of-the-art baselines with handshaking. Moreover, we demonstrate the efficacy and flexibility of hugging and hugging$^+$ to exiting UCMH methods when utilizing transformers.

Extension Notes Compared with our preliminary work (Wang et al., 2022b), this paper has been improved and further studied from three aspects. (i) *Technical extension.* First, we further investigate how fine-grained alignment promotes global hash code learning. The details can be found in Sect. 4.3.2, including the comparison of handshaking and hugging in three aspects: training dynamics, visualization and clustering analysis of the fine-grained latent space, and attention visualization. We believe they can provide readers

with a deeper understanding of hugging's efficacy. Second, hugging drops fine-grained parts in inference and hence falls short of leveraging fine-grained representations. This limitation was unresolved in our preliminary paper. In this paper, we fulfill an effective solution by extending hugging to hugging$^+$, where first we integrate quantization learning with fine-grained alignment in training and then take fine-grained quantization codes as a bonus for efficient reranking in inference. Our new solution obtains a better tradeoff between retrieval efficacy and efficiency. Third, we contribute a new approach to fine-grained quantization by transforming the classical solution of the orthogonal matrix in Optimized Product Quantization (OPQ) (Ge et al., 2013) into a series of learnable Householder matrices. This integration seamlessly incorporates OPQ into the deep learning pipeline, enhancing fine-grained semantic preservation and quantization quality. This contribution can benefit various existing quantization approaches. (ii) *Experimental extension.* We include new results on one text-image retrieval (Chua et al., 2009) and two text-video retrieval datasets (Xu et al., 2016; Chen and Dolan, 2011). We join two state-of-the-art and open-sourced approaches (Zhu et al., 2023; Tu et al., 2023) in the investigation of transformer-based UCMH. We also conduct qualitative analysis and extensive ablation studies to justify the efficacy of our design. (iii) *Survey extension.* We update the survey of related work and add more discussion on the relation and differences to the latest papers.

2 Related Work

2.1 Hashing for Fast Visual Search

Hashing aims to transform high-dimensional data into compact binary codes while preserving semantic information, which has been extensively studied in visual search due to fast retrieval speed and low storage cost (Wang et al., 2016, 2018). According to different strategies on distance (or similarity) computation, it can be subdivided into *binary hashing* and *quantization*. Specifically, binary hashing (Datar et al., 2004)[1] transforms continuous embeddings into the Hamming space such that distances can be quickly computed with bitwise operators. Quantization (Jégou et al., 2011) divides embedding space into disjoint clusters and approximates each data point by the nearest centroid. By pre-computing the inter-centroid distances in a lookup table, the search speed can be greatly accelerated.

To preserve semantic information, traditional binary hashing (Datar et al., 2004; Weiss et al., 2008; Heo et al., 2012; Gong et al., 2013) and quantization (Ge et al., 2013; Babenko and Lempitsky, 2014; Zhang et al., 2014; Kalantidis and Avrithis, 2014; Martinez et al., 2016) approaches adopted various heuristic strategies that are sensitive to the statistical property of application data and inflexible to adapt to new data. More effectively, deep binary hashing (Liong et al., 2015; Zhu et al., 2016; Liu et al., 2019) and quantization (Cao et al., 2017; Yu et al., 2020) approaches jointly optimized deep feature extraction and hashing (or quantization) in an end-to-end fashion, which have been shown to outperform traditional approaches in image retrieval (Shen et al., 2019; Zhang et al., 2019; Wang et al., 2021; Cui et al., 2021; Sun et al., 2022), video retrieval (Li et al., 2021b, 2022a; Zeng et al., 2022; Wang et al., 2023), multi-modal retrieval (Zheng et al., 2020; Tan et al., 2022), and cross-modal retrieval (Chen et al., 2018; Sun et al., 2019; Shen et al., 2021; Li et al., 2021; Tu et al., 2022).

From a practical perspective, we present a hybrid hashing framework that includes binary hashing and quantization for two-stage efficient retrieval. Here we discuss the relations to existing work from different aspects and highlight the insights in our design. (i) (Shi and Chung, 2021) proposed a similar two-stage inference pipeline to our hugging$^+$ that learned binary hashing and quantization at once for preliminary ranking and further reranking, respectively. Nevertheless, this approach was not designed for transformer-based hashing. Besides, both binary hashing and quantization in it are built on the same global features, which tends to homogenize the two stages and thereby limits the performance gain. Differently, hugging$^+$ develops global hashing and fine-grained quantization, which leverages heterogeneous, coarse-to-fine semantics to maximize the gain. In addition, both global and fine-grained components can be mutually promoted in hugging-style training, leading to a win-win situation in inference. (ii) The fine-grained quantization in hugging$^+$ follows the common practice in deep quantization (Klein and Wolf, 2019; Yu et al., 2020; Jang and Cho, 2021; Wang et al., 2022a) that uses a differentiable trick (Chen et al., 2020b) to relax the optimization of product quantization (PQ) (Jégou et al., 2011), making the whole model end-to-end back-propagable. Furthermore, we revisit optimized product quantization (OPQ) (Ge et al., 2013) and introduce an isometric matrix to minimize quantization distortions. Different from OPQ, which solved the matrix by intractable SVD decomposition, we propose a differentiable solution based on the Householder transformation, enabling end-to-end deep learning.

[1] Binary hashing is sometimes called *hashing* for short when quantization is regarded as a *parallel* concept to hashing. But to be precise, quantization should be regarded as a *subordinate* concept to hashing as it essentially produces a hash function.

2.2 Unsupervised Cross-Modal Hashing (UCMH)

Traditional UCMH approaches learned to transform hard-craft features (Lowe, 2004) into binary codes by solving linear problems, e.g. matrix factorization (Zhou et al., 2014; Ding et al., 2016; Hu et al., 2019) and spectral decomposition (Kumar and Udupa, 2011; Song et al., 2013). The shallow features and linear solutions limited the performance and scalability. In contrast, by leveraging deep neural networks (DNNs), deep UCMH approaches can capture richer semantic information and generate better hash codes. Early deep approaches (Hu et al., 2019; Wu et al., 2018; Hoang et al., 2020) replaced the hand-crafted features with the deep features and applied linear solutions as in some shallow approaches (Kumar and Udupa, 2011; Zhu et al., 2013; Song et al., 2013; Hu et al., 2019). To better estimate pairwise similarity to guide hash learning during training, later deep approaches (Su et al., 2019; Liu et al., 2020; Wang et al., 2020b; Yang et al., 2020; Zhang et al., 2022; Yu et al., 2021a; Zhang et al., 2021; Zhu et al., 2023; Tu et al., 2023) delved into fusing multi-modal affinities to estimate a precise similarity matrix. Besides, some recent deep approaches tried to narrow the modality gap by adversarial learning (Zhang et al., 2018; Li et al., 2019; Zhang and Peng, 2020), knowledge distillation (Hu et al., 2020; Li and Wang, 2021), or self-supervised learning (Wang et al., 2020, 2021; Hoang et al., 2023; Mikriukov et al., 2022), showing promising results.

Note that existing deep approaches mainly used classic AlexNet (Krizhevsky et al., 2012) or VGGNets (Simonyan and Zisserman, 2015) to extract visual features and used MLPs to encode text information, which fell short of semantic extraction and limited hash representations. Instead, our paper highlights the significance of transformers to UCMH and conducts a detailed study on transformer-based UCMH. We examine the efficacy of transformers to UCMH with five representative and open-sourced approaches (Su et al., 2019; Yang et al., 2020; Yu et al., 2021a; Zhu et al., 2023; Tu et al., 2023). More importantly, we also propose effective multi-granularity hash learning strategies that provide new insights into the transformer-based UCMH.

2.3 Transformers for Multimedia Retrieval

Recently, transformers (Vaswani et al., 2017) have made remarkable progress in CV (Dosovitskiy et al., 2021; Carion et al., 2020; Liu et al., 2021b) and NLP (Devlin et al., 2019; Liu et al., 2019c; Yang et al., 2019) tasks, triggering the surge toward better multimedia understanding. In cross-modal retrieval, the potential of transformers has been widely explored (Shin et al., 2022). Single-stream retrieval methods (Lu et al., 2019; Song and Soleymani, 2019; Li et al., 2020a; Gao et al., 2020; Yu et al., 2021b; Bao et al., 2022b; Radenovic et al., 2023) designed unified models with fully cross-modal interaction. Albeit the superior performance, the quadratic complexity of pairwise interaction limited their retrieval efficiency. Dual-stream methods developed separate encoders for modality-specific representations, in which the core is to align the representations between modalities. Most dual-stream methods (Liu et al., 2019b; Gabeur et al., 2020; Radford et al., 2021; Liu et al., 2021a; Yang et al., 2021; Patrick et al., 2021; Bain et al., 2021; Luo et al., 2022; An et al., 2023; Liu et al., 2023; Lin et al., 2023) globally aligned the aggregation tokens (e.g. `[CLS]`) with metric learning (Liu, 2009) or contrastive learning (Chen et al., 2020a), where negative sampling strategies have been extensively investigated to enhance representation learning. To reduce the modality gap, many recent works further excavated fine-grained semantics from content tokens (e.g. text words or image patches) and designed various fine-grained interaction strategies, augmenting cross-modal learning and relevance estimation. For instance, Messina et al. (2021); Yao et al. (2022); Wu et al. (2023) measured cross-modal token-wise relevance and aggregated the fine-grained scores by pooling strategies. Wang et al. (2021b) developed a global–local fusion approach that combines global and local similarities for alignment and retrieval. Fang et al. (2023) proposed an uncertainty-adaptive approach that models cross-modal interaction as a distribution matching procedure. Jin et al. (2023b) designed a text-frame attention module to enhance cross-modal interaction and proposed a novel generative retrieval approach based on diffusion mechanism. Zala et al. (2023); Jin et al. (2023a) adopted a hierarchical perspective to model cross-modal interaction at different levels, such as entity, action, and event, where Jin et al. (2023a) also introduced a novel Banzhaf Interaction mechanism to refine cross-modal correspondence. In general, these fine-grained approaches help to achieve better performance but sacrifice retrieval efficiency, because their test-time efficacy essentially relies on complex similarity computation.

In the specific field of hashing-based retrieval, recent advances in image (Lu et al., 2021; Li et al., 2022; Dubey et al., 2022; Chen et al., 2022) and video hashing (Li et al., 2021b; He et al., 2021; Zeng et al., 2022; Wang et al., 2023) have also revealed the effectiveness of uni-modal transformer hashing. In the cross-modal scenario, CLIP-based (Radford et al., 2021) hashing approaches have emerged recently for text-image and text-video retrieval. SACH (Yu et al., 2022) and DCMHT (Tu et al., 2022) are two latest approaches for supervised and unsupervised text-image hashing. Despite impressive results, their successes mainly relied on off-the-shelf well-aligned transformers. Besides, Tu et al. (2022) further leveraged ground-truth labels to guide hash learning. CLIP4Hashing (Zhuo et al., 2022) adapted CLIP for text-video UCMH, where the text and video encoders were only aligned via global hash representations. In contrast, we proposed effective and general solutions for UCMH

with *unaligned* transformers. By adopting multi-granularity alignment in training, we effectively enhance the cross-modal consistency, producing better hash representations than CLIP4Hashing.

3 Methodology

3.1 Problem Formulation and Model Overview

Given an unlabeled training set $\mathcal{D}$ of $N_{\mathcal{D}}$ naturally coexisted dual-modal (e.g. text-image or text-video) pairs, our goal is to learn a pair of modality-specific hash encoders that encode texts and images (or videos) as L-bit semantic-preserving binary codes for cross-modal retrieval.

To this end, we construct a HUGGINGHASH model using the *hugging* framework, as illustrated in Fig. 2. Specifically, given a training pair, we first preprocess the text and the image (or video) as the input tokens for transformers. Then, we extract features with transformers and get the output embeddings from the `[CLS]` and content tokens (Sect. 3.2). Next, we forward the `[CLS]` tokens to the hash modules and produce text and image (or video) hash code vectors. Meanwhile, we project the embeddings of content tokens to a cross-modal latent space. Finally, we conduct multi-granularity alignment, including global alignment based on the hash codes (Sect. 3.3.1) and fine-grained alignment based on latent local representations (Sect. 3.3.2), to bridge text and visual modalities. We further design learnable optimized quantization collaborating with the fine-grained alignment to make the best of fine-grained representations (Sect. 3.3.3). In inference, we obtain hash codes through the global branch and enable the generation of fine-grained quantization codes (Sect. 3.4.1). We take global hash codes for preliminary ranking (Sect. 3.4.2) and also provide optional reranking with fine-grained quantization codes (Sect. 3.4.3).

3.2 Base Encoders

3.2.1 Text Encoder

For each text sample, we tokenize it into word pieces and construct content tokens. Then we append a `[CLS]` token and form the text input. Denote the token sequence of the ith text in a mini-batch $\mathcal{B}$ by $\mathcal{T}_i = \{t_{i,[\mathrm{CLS}]}, t_{i,1}, t_{i,2}, \cdots, t_{i,K_i^{\mathrm{t}}}\}$, where K_i^{t} is the number of content tokens[2] for the ith text. We pad the sequence to a fixed length, add position embeddings and forward it to the BERT (Devlin et al., 2019) encoder $f^{\mathrm{t}}(\cdot;\boldsymbol{\theta}_f^{\mathrm{t}})$ to compute the token embeddings, namely $\boldsymbol{x}_{i,[\mathrm{CLS}]}^{\mathrm{t}} \in \mathbb{R}^{D^{\mathrm{t}}}$ and $\{\boldsymbol{x}_{i,k}^{\mathrm{t}}\}_{k=1}^{K_i^{\mathrm{t}}} \subset \mathbb{R}^{D^{\mathrm{t}}}$. We can formulate the whole process by

$$\boldsymbol{x}_{i,k}^{\mathrm{t}} = f^{\mathrm{t}}(\mathcal{T}_i;\boldsymbol{\theta}_f^{\mathrm{t}})_k,\ k = [\mathrm{CLS}], 1, 2, \cdots, K_i^{\mathrm{t}}. \tag{1}$$

3.2.2 Image Encoder

For each image sample, we use the ViT (Dosovitskiy et al., 2021) pre-processor to patchify it into a fixed number (e.g. 196) of content tokens and add a `[CLS]` token to form the image input. We denote the token sequence of the ith image in a mini-batch $\mathcal{B}$ by $\mathcal{V}_i = \{v_{i,[\mathrm{CLS}]}, v_{i,1}, v_{i,2}, \cdots, v_{i,K^{\mathrm{v}}}\}$, where K^{v} is the number of content tokens. We then add position embeddings and forward them to the ViT $f^{\mathrm{v}}(\cdot;\boldsymbol{\theta}_f^{\mathrm{v}})$ to compute embeddings, namely $\boldsymbol{x}_{i,[\mathrm{CLS}]}^{\mathrm{v}} \in \mathbb{R}^{D^{\mathrm{v}}}$ and $\{\boldsymbol{x}_{i,k}^{\mathrm{v}}\}_{k=1}^{K^{\mathrm{v}}} \subset \mathbb{R}^{D^{\mathrm{v}}}$. Analogous to the text side, we summarize the image feature extraction process by

$$\boldsymbol{x}_{i,k}^{\mathrm{v}} = f^{\mathrm{v}}(\mathcal{V}_i;\boldsymbol{\theta}_f^{\mathrm{v}})_k,\ k = [\mathrm{CLS}], 1, 2, \cdots, K^{\mathrm{v}}. \tag{2}$$

3.2.3 Video Encoder

In addition to text-image retrieval, HUGGINGHASH also supports hash-based text-video retrieval tasks. To be concise, we reuse the notations in Sect. 3.2.2 and assign analogous definitions to them. Given the ith video in a mini-batch $\mathcal{B}$, we first adopt ViT as the spatial encoder to extract frame-level features $\boldsymbol{v}_{i,1}, \boldsymbol{v}_{i,2}, \cdots, \boldsymbol{v}_{i,K^{\mathrm{v}}}$, where K^{v} is the number of sampled frames. We append a temporal `[CLS]` token to frame features, add temporal positional embeddings, forming the input token sequence $\mathcal{V}_i = \{\boldsymbol{v}_{i,[\mathrm{CLS}]}, \boldsymbol{v}_{i,1}, \boldsymbol{v}_{i,2}, \cdots, \boldsymbol{v}_{i,K^{\mathrm{v}}}\}$. Then, we build a lightweight self-attention layer upon $\mathcal{V}_i$ to capture temporal semantics, producing output embeddings by $\boldsymbol{x}_{i,[\mathrm{CLS}]}^{\mathrm{v}} \in \mathbb{R}^{D^{\mathrm{v}}}$ and $\{\boldsymbol{x}_{i,k}^{\mathrm{v}}\}_{k=1}^{K^{\mathrm{v}}} \subset \mathbb{R}^{D^{\mathrm{v}}}$. We denote the whole video encoder by $f^{\mathrm{v}}(\cdot;\boldsymbol{\theta}_f^{\mathrm{v}})$, and formulate the encoding process by

$$\boldsymbol{x}_{i,k}^{\mathrm{v}} = f^{\mathrm{v}}(\mathcal{V}_i;\boldsymbol{\theta}_f^{\mathrm{v}})_k,\ k = [\mathrm{CLS}], 1, 2, \cdots, K^{\mathrm{v}}. \tag{3}$$

3.3 *Hugging*: Multi-granularity Alignment with Transformers for Hash Learning

Transformers provide better multimedia understanding to facilitate hash learning, but aligning heterogeneous knowledge between different modalities is challenging. We present *hugging*, a multi-granularity alignment framework to tackle the challenge. In addition to the global alignment for hash codes, we design fine-grained alignment using GhostVLAD

[2] We define content tokens as those non-special tokens. Any special tokens, including classification (`[CLS]`), 'end-of-sentence' (`[EOS]`), padding (`[PAD]`), mask (`[MASK]`), and 'out-of-vocabulary' (`[UNK]`), will be removed from transformer's output token sequence.

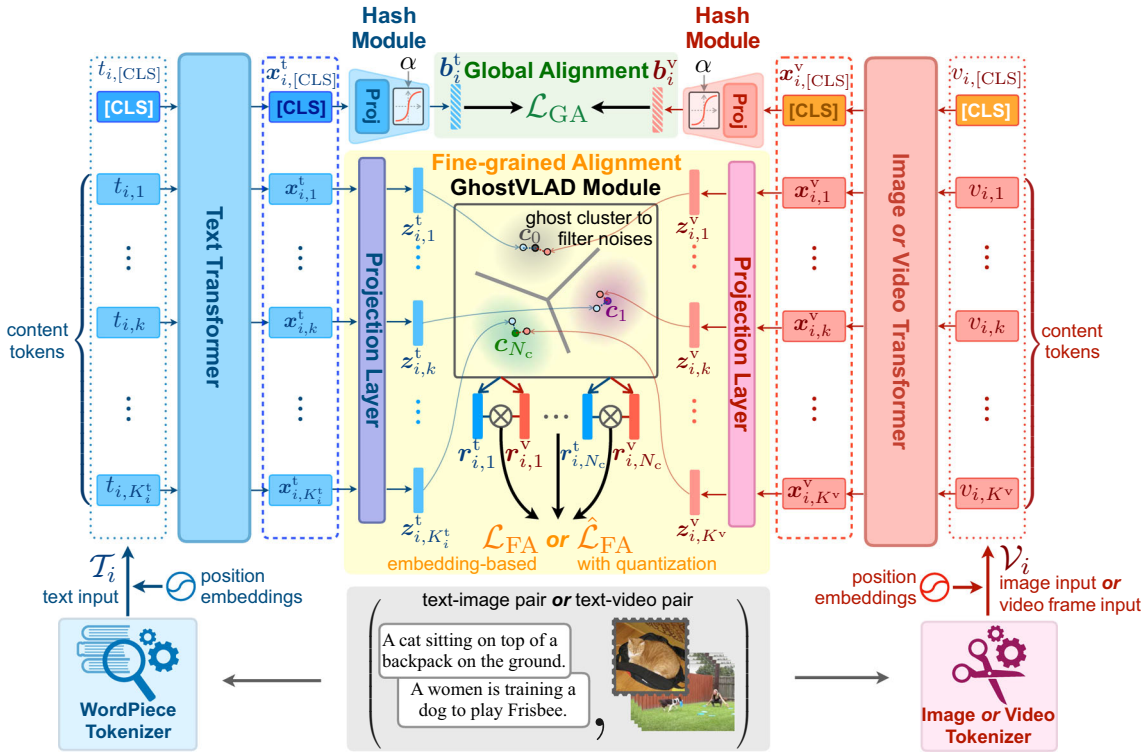

Fig. 2 The pipeline HUGGINGHASH. During training, it integrates global alignment based on hash codes and fine-grained alignment with a GhostVLAD (Zhong et al., 2018) module. We design embedding-based (Sect. 3.3.2) and quantization-based (Sect. 3.3.3) objectives as alternatives for fine-grained alignment. The synergy of global and fine-grained alignment shows a positive effect on both components during learning. In inference, we obtain hash representations for retrieval via the global branch. Quantization-based fine-grained alignment further enables fine-grained quantization code generation as a bonus, which can be exploited to refine retrieval results by efficient reranking. *Best viewed in color*

(Zhong et al., 2018) with content tokens. The global alignment provides direct guidance on hash code learning, while the fine-grained alignment supplies effective regularization to reduce the modality gap.

3.3.1 Global Alignment

We apply global alignment to the hash codes. First, we project and convert the output embeddings of the aggregation tokens (i.e., `[CLS]`) into binary hash codes:

$$\boldsymbol{h}_i^{\mathrm{t}} = \tanh\left(\alpha \cdot \phi^{\mathrm{t}}(\boldsymbol{x}_{i,\texttt{[CLS]}}^{\mathrm{t}})\right) \in [-1,+1]^L, \tag{4}$$

$$\boldsymbol{h}_i^{\mathrm{v}} = \tanh\left(\alpha \cdot \phi^{\mathrm{v}}(\boldsymbol{x}_{i,\texttt{[CLS]}}^{\mathrm{v}})\right) \in [-1,+1]^L, \tag{5}$$

$$\boldsymbol{b}_i^{\mathrm{t}} = \boldsymbol{h}_i^{\mathrm{t}} - \mathrm{sg}\left(\boldsymbol{h}_i^{\mathrm{t}} - \mathrm{sgn}(\boldsymbol{h}_i^{\mathrm{t}})\right) \in \{-1,+1\}^L, \tag{6}$$

$$\boldsymbol{b}_i^{\mathrm{v}} = \boldsymbol{h}_i^{\mathrm{v}} - \mathrm{sg}\left(\boldsymbol{h}_i^{\mathrm{v}} - \mathrm{sgn}(\boldsymbol{h}_i^{\mathrm{v}})\right) \in \{-1,+1\}^L, \tag{7}$$

where ϕ^{t} and ϕ^{v} are modality-specific projections that transform $\mathbb{R}^{D^{\mathrm{t}}}$-dimensional text features and $\mathbb{R}^{D^{\mathrm{v}}}$-dimensional visual features into an L-dimensional shared latent space, respectively. $\alpha > 0$ is a factor in controlling the smoothness of the tanh outputs. $\boldsymbol{h}_i^{\mathrm{t}}$ and $\boldsymbol{h}_i^{\mathrm{v}}$ are smoothed hash codes. $\mathrm{sgn}(\cdot)$ is the sign function that outputs $+1$ for positive input and -1 otherwise on each element. $\mathrm{sg}(\cdot)$ is the *stop gradient* operator that is the identity function in the forward pass but drops gradient for variables inside it during the backward pass. Equations (6) and (7) allow us to directly pass the gradient straight through (Bengio et al., 2013) the binary hash codes, i.e., $\boldsymbol{b}_i^{\mathrm{t}}$ and $\boldsymbol{b}_i^{\mathrm{v}}$. In HUGGINGHASH, we adopt the contrastive learning loss (van den Oord et al., 2018; He et al., 2020; Chen et al., 2020a) for global alignment, namely

$$\ell_{\mathrm{GA},i}^{\vec{\mathrm{tv}}} = -\log \frac{\exp(M_{ii}/\tau)}{\sum_{j=1}^{|\mathcal{B}|}\exp(M_{ij}/\tau)}, \tag{8}$$

$$\ell_{\mathrm{GA},i}^{\vec{\mathrm{vt}}} = -\log \frac{\exp(M_{ii}/\tau)}{\sum_{j=1}^{|\mathcal{B}|}\exp(M_{ji}/\tau)}, \tag{9}$$

$$\mathcal{L}_{\mathrm{GA}} = \frac{1}{2|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\left(\ell_{\mathrm{GA},i}^{\vec{\mathrm{tv}}} + \ell_{\mathrm{GA},i}^{\vec{\mathrm{vt}}}\right), \tag{10}$$

where $\mathcal{B}$ denotes a mini-batch. $M_{ij} = \cos(\boldsymbol{b}_i^{\mathrm{t}}, \boldsymbol{b}_j^{\mathrm{v}})$. $\tau > 0$ is the temperature hyper-parameter.

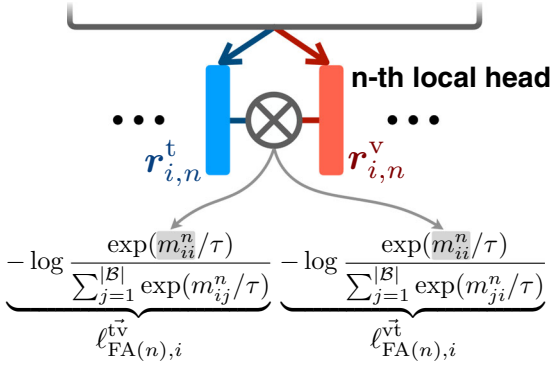

Fig. 3 Embedding-based fine-grained alignment

3.3.2 Fine-Grained Alignment with Locally Aggregated Descriptors

We present a clustering-based strategy with GhostVLAD (Zhong et al., 2018) for fine-grained alignment. Our basic idea is to exploit concept-aware semantics and enable concept-aware alignment in the latent space. Specifically, we first project the output embeddings of the content tokens into a shared latent space, namely,

$$z^{t}_{i,k} = \psi^{t}(x^{t}_{i,k}), \quad k = 1, 2, \cdots, K^{t}_{i}, \tag{11}$$
$$z^{v}_{i,k} = \psi^{v}(x^{v}_{i,k}), \quad k = 1, 2, \cdots, K^{v}, \tag{12}$$

where ψ^{t} and ψ^{v} are the projections that transform $\mathbb{R}^{D^{t}}$-dimensional text features and $\mathbb{R}^{D^{v}}$-dimensional visual features into an D-dimensional fine-grained shared latent space, respectively. We denote the collections of latent embeddings for text and visual content tokens as $\mathcal{Z}^{t}_{i} = \{z^{t}_{i,k}\}_{k=1}^{K^{t}_{i}}$ and $\mathcal{Z}^{v}_{i} = \{z^{v}_{i,k}\}_{k=1}^{K^{v}}$, respectively.

Then, we use a GhostVLAD module to learn N_c+1 cluster centroids, $\{c_0, c_1, c_2, \cdots, c_{N_c}\}$. In particular, we designate c_0 as the "ghost" centroid to filter noise, e.g. uninformative words in a sentence and background features for an image or a video. Each other centroid is expected to represent a latent concept or attribute, such as color, scene, etc., contributing to a partial description of the object. We forward $\mathcal{Z}^{t}_{i}$ and $\mathcal{Z}^{v}_{j}$ to the GhostVLAD, where each token is viewed as composites of different concepts and will be softly assigned to all clusters. We use assignment scores to estimate the relevance of a token to all concepts or attributes. For instance, the assignment score of the text token $z^{t}_{i,k}$ *w.r.t.* the nth cluster is computed by

$$a^{t}_{i,k,n} = \frac{\exp(\text{BatchNorm}(w_n^{\top} z^{t}_{i,k}))}{\sum_{n'=0}^{N_c}\exp(\text{BatchNorm}(w_{n'}^{\top} z^{t}_{i,k}))}, \tag{13}$$

where BatchNorm$(\cdot)$ is batch normalization (Ioffe and Szegedy, 2015) and $W = [w_0, w_1, \cdots, w_{N_c}]$ is a trainable parameter matrix. Suppose the nth cluster is associated with a latent concept of 'animal', then $a^{t}_{i,k,n}$ indicates the relevance of the text token $z^{t}_{i,k}$ to 'animal', e.g. 70%.

After clustering, we aggregate modality-wise residual embeddings at each cluster except the "ghost" cluster. For the nth cluster, we aggregate residual embeddings of $\mathcal{Z}^{t}_{i}$ and $\mathcal{Z}^{v}_{j}$ respectively by

$$r^{t}_{i,n} = \sum_{k=1}^{K^{t}_{i}} a^{t}_{i,k,n} \cdot (z^{t}_{i,k} - c_n), \tag{14}$$
$$r^{v}_{j,n} = \sum_{k=1}^{K^{v}} a^{v}_{j,k,n} \cdot (z^{v}_{j,k} - c_n). \tag{15}$$

In Eqs. (14) and (15), each residual embedding represents the token semantics *conditioned* by the concept associated with the cluster. For instance, $(z^{t}_{i,k} - c_n)$ may represent that *given the concept of 'animal'*, the token is about the 'cat'. Notably, assignment scores and residual embeddings are designed to capture different information, i.e., concept relevance and concept-aware semantics, respectively. So we decouple them by using independent parameters for the assigner and centroids. Tokens with high relevance scores do not necessarily approach the centroid. Then the aggregated embedding can depict the total semantics of the whole token sequence *w.r.t.* the concept associated with the cluster. For example, $r^{t}_{i,n}$ may indicate how the text input $\mathcal{Z}^{t}_{i}$ is relevant to 'animal' and what 'animal' it tells.

Finally, we define fine-grained alignment as aligning aggregated local representations from text and vision cluster by cluster, and introduce the cluster-wise contrastive learning loss as

$$\ell^{\vec{tv}}_{FA(n),i} = -\log \frac{\exp(m^n_{ii}/\tau)}{\sum_{j=1}^{|\mathcal{B}|}\exp(m^n_{ij}/\tau)}, \tag{16}$$
$$\ell^{\vec{vt}}_{FA(n),i} = -\log \frac{\exp(m^n_{ii}/\tau)}{\sum_{j=1}^{|\mathcal{B}|}\exp(m^n_{ji}/\tau)}, \tag{17}$$
$$\mathcal{L}_{FA(n)} = \frac{1}{2|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\left(\ell^{\vec{tv}}_{FA(n),i} + \ell^{\vec{vt}}_{FA(n),i}\right), \tag{18}$$
$$\mathcal{L}_{FA} = \frac{1}{N_c}\sum_{n=1}^{N_c}\mathcal{L}_{FA(n)}, \tag{19}$$

where $m^n_{ij} = \cos(r^{t}_{i,n}, r^{v}_{j,n})$ is the fine-grained similarity of $\mathcal{Z}^{t}_{i}$ and $\mathcal{Z}^{v}_{j}$ *w.r.t.* the nth cluster. We illustrate this embedding-based alignment in Fig. 3. As each aggregated representation is expected to capture both concept relevance (distribution statistics of assignment) and concept-aware semantics (residual embedding), cluster-wise alignment serves as a strong

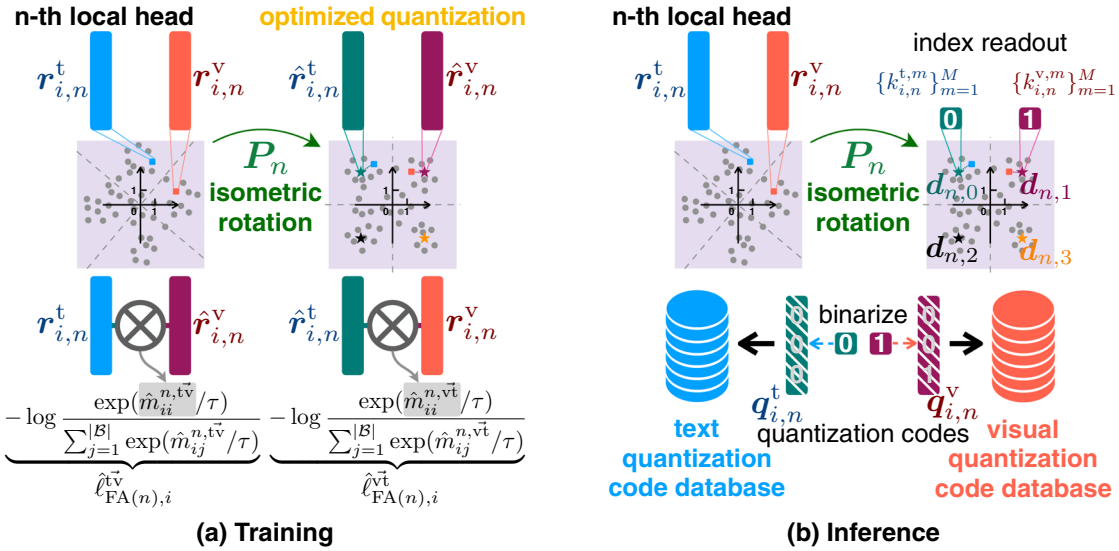

Fig. 4 The training and inference processes of quantization-based fine-grained alignment

signal to regularize cross-modal consistency concept by concept, thus fulfilling fine-grained alignment.

Remark: Criteria for choosing clustering algorithm The choice of clustering algorithm is crucial for the fine-grained alignment. Our selection criteria include the following: (i) End-to-end optimization capability is mandatory, as we fundamentally train neural networks to achieve cross-modal alignment. Unfortunately, we find that most classic clustering algorithms like DBSCAN (Ester et al., 1996), do not meet this requirement. Other preferences, in decreasing order, include: (ii) Interpretability in the latent space, such as latent topics for text and visual concepts for images. It is preferable to capture the correlation between samples and latent information. (iii) Adequate robustness to noise, since we often lack detailed fine-grained annotations, such as correspondences between text words and image regions, which necessitates noise reduction and key information extraction. (iv) Simplicity and efficiency, as slow or complex algorithms will hinder training efficiency. (v) The output of the clustering algorithm should also support concise and efficient fine-grained cross-modal alignment.

We opt for GhostVLAD because it meets all these criteria. Specifically, (i) GhostVLAD is derived from NetVLAD (Arandjelovic et al., 2016), itself a trainable neural network module successfully applied to tasks like place recognition (Arandjelovic et al., 2016), image retrieval (Humenberger et al., 2022), and face recognition (Zhong et al., 2018). (ii) GhostVLAD produces interpretable results. It learns a fixed number of cluster prototypes in the latent space, serving as indicators of fine-grained concepts. By adaptively aggregating residuals of each content token *w.r.t.* each cluster centroid, it provides representations *w.r.t.* different latent concepts that aid comprehensive sample descriptions. (iii) GhostVLAD employs an information aggregation process that reduces sensitivity to noise introduced by individual tokens. Besides, compared to NetVLAD, it introduces ghost (i.e., idle) clusters, enhancing noise filtering through end-to-end training. (iv) The design of GhostVLAD is GPU-friendly, consisting of common deep learning operators and only requiring a single pass for clustering, as opposed to multiple iterations in k-means. (v) Utilizing GhostVLAD's output for cross-modal alignment is simple and efficient. The output comprises fixed-size representations *w.r.t.* the number of local clusters, irrespective of the number of content tokens. Alignment is achieved through one-to-one matching *w.r.t.* each local cluster, avoiding exhaustive cross-interaction between two embedding sets and thus promoting conciseness and efficiency.

3.3.3 Optimized Quantization Learning for Fine-Grained Representations

When the training is equipped with embedding-based fine-grained alignment, the fine-grained parts have to be discarded in inference to maintain the efficiency of hash-based retrieval. It falls short of leveraging fine-grained representations and has yet to be resolved in our preliminary work (Wang et al., 2022b). Here we provide a quantization-based solution to this problem, which aims to align fine-grained cross-modal correspondence and learn quantized representations jointly.

Suppose at the nth local head of fine-grained alignment, we have a D-dimensional continuous-value embedding $\boldsymbol{r}_n$ to be quantized with M sub-codebooks, namely $\boldsymbol{D}_n^1, \boldsymbol{D}_n^2, \cdots, \boldsymbol{D}_n^M$. The mth sub-codebook $\boldsymbol{D}_n^m$ consists of K sub-codewords $\boldsymbol{d}_{n,0}^m, \boldsymbol{d}_{n,1}^m, \boldsymbol{d}_{n,K-1}^m \in \mathbb{R}^d$. The problem of

product quantization (Jégou et al., 2011) is given by

$$\hat{\boldsymbol{r}}_n = \underset{\boldsymbol{d}_n \in \boldsymbol{D}_n^1 \times \boldsymbol{D}_n^2 \times \cdots \times \boldsymbol{D}_n^M}{\arg\min} \|\boldsymbol{r}_n - \boldsymbol{d}_n\|_2^2 . \tag{20}$$

Let $\mathbb{R}^D \equiv \mathbb{R}^{Md}$, where M and d are both positive integers. We divide $\boldsymbol{r}_n$ into M equal-length d-dimensional segments, i.e., $\boldsymbol{r}_n \equiv [\boldsymbol{r}_n^1, \boldsymbol{r}_n^2, \cdots, \boldsymbol{r}_n^M]$. Then, the original problem in Eq. (20) can be re-formulated into M independent sub-problems. For example, the mth sub-problem is defined as

$$\hat{\boldsymbol{r}}_n^m = \underset{\boldsymbol{d}_{n,k}^m \in \boldsymbol{D}_n^m}{\arg\min} \left\|\boldsymbol{r}_n^m - \boldsymbol{d}_{n,k}^m\right\|_2^2 . \tag{21}$$

By imposing $\left\|\boldsymbol{r}_n^m\right\|_2 = \left\|\boldsymbol{d}_n^m\right\|_2$ (e.g. L2 normalization), Eq. (21) can be re-written as

$$\hat{\boldsymbol{r}}_n^m = \underset{\boldsymbol{d}_{n,k}^m \in \boldsymbol{D}_n^m}{\arg\max} \left\langle \boldsymbol{r}_n^m, \boldsymbol{d}_{n,k}^m \right\rangle, \tag{22}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product operator.

To enable end-to-end deep learning, we follow the common practice of deep quantization approaches (Klein and Wolf, 2019; Yu et al., 2020) that relax the Eq. (22) by softmax trick, producing

$$\hat{\boldsymbol{r}}_n^m = \sum_{k=0}^{K-1} p_{n,k}^m \boldsymbol{d}_{n,k}^m, \tag{23}$$

$$p_{n,i}^m = \frac{\exp(\beta \cdot \langle \boldsymbol{r}_n^m, \boldsymbol{d}_{n,k'}^m \rangle)}{\sum_{k'=0}^{K-1} \exp(\beta \cdot \langle \boldsymbol{r}_n^m, \boldsymbol{d}_{n,k'}^m \rangle)}. \tag{24}$$

$p_{n,k}^m \in [0,1]^K$ is the codeword selection probability *w.r.t.* $\boldsymbol{D}_n^m$. $\beta > 0$ is a scaling factor such that Eq. (23) approximates Eq. (22) when $\beta \to +\infty$.

As pointed out by a classic approach, optimized product quantization (OPQ) (Ge et al., 2013), the way to divide $\boldsymbol{r}_n$ into $\boldsymbol{r}_n^1, \boldsymbol{r}_n^2, \cdots, \boldsymbol{r}_n^M$ (i.e., sub-space partition) is important to the quantization quality. However, existing deep quantization practices have not considered this aspect, which may result in large quantization distortions. Inspired by OPQ, we further design ***optimized quantization learning*** that introduces an isometric rotation matrix $\boldsymbol{P}_n \in \mathbb{R}^{D \times D}$ to optimize the sub-space partition in Eq. (20), leading to

$$\hat{\boldsymbol{r}}_n = \underset{\boldsymbol{d}_n \in \boldsymbol{D}_n^1 \times \boldsymbol{D}_n^2 \times \cdots \times \boldsymbol{D}_n^M}{\arg\min} \|\boldsymbol{P}_n \boldsymbol{r}_n - \boldsymbol{d}_n\|_2^2 , \quad \text{s.t.} \quad \boldsymbol{P}_n^\top \boldsymbol{P}_n = \boldsymbol{I}. \tag{25}$$

$\boldsymbol{I}$ denotes the identity matrix.

Note that in OPQ, the problem of Eq. (25) is solved by SVD decomposition, which is intractable for the end-to-end deep learning pipeline. Differently, we design a back-propagatable solution to Eq. (25). First, we set a series of trainable parameters $\boldsymbol{u}_{n,1}, \boldsymbol{u}_{n,2}, \cdots, \boldsymbol{u}_{n,N_\mathrm{h}}$ and transform them into orthogonal Householder matrices, namely

$$\boldsymbol{H}_{n,h} = \boldsymbol{I} - 2\frac{\boldsymbol{u}_{n,h}\boldsymbol{u}_{n,h}^\top}{\left\|\boldsymbol{u}_{n,h}\right\|_2^2}, \ 1 \le h \le N_\mathrm{h} \le D. \tag{26}$$

Then, we parameterize $\boldsymbol{P}_n$ by a product of N_h Householder matrices, giving

$$\boldsymbol{P}_n = \prod_{h=1}^{N_\mathrm{h}} \boldsymbol{H}_{n,h}. \tag{27}$$

In practice, we implement $\boldsymbol{P}_n \boldsymbol{r}_n$ in Eq. (25) by applying N_h iterations:

$$\boldsymbol{r}_{n,0} := \boldsymbol{r}_n, \tag{28}$$

$$\boldsymbol{r}_{n,h} := \boldsymbol{r}_{n,h-1} - \frac{2\boldsymbol{u}_{n,h}}{\left\|\boldsymbol{u}_{n,h}\right\|_2^2} \cdot \boldsymbol{u}_{n,h}^\top \boldsymbol{r}_{n,h-1}, \quad 1 \le h \le N_\mathrm{h} \le D, \tag{29}$$

$$\boldsymbol{P}_n \boldsymbol{r}_n := \boldsymbol{r}_{n,N_\mathrm{h}}. \tag{30}$$

After the isometric rotation, we apply Eqs. (23) and (24) as trainable quantization.

Analogous to embedding-based fine-grained alignment, here we introduce how to align fine-grained representations and train the optimized quantization module simultaneously. We denote the quantized representations of $\boldsymbol{r}_{i,n}^\mathrm{t}$ and $\boldsymbol{r}_{i,n}^\mathrm{v}$ by $\hat{\boldsymbol{r}}_{i,n}^\mathrm{t}$ and $\hat{\boldsymbol{r}}_{i,n}^\mathrm{v}$, respectively. Besides, we define the asymmetric-quantized similarity by

$$\hat{m}_{ij}^{n,\vec{\mathrm{tv}}} = \cos(\boldsymbol{r}_{i,n}^\mathrm{t}, \hat{\boldsymbol{r}}_{j,n}^\mathrm{v}), \tag{31}$$

$$\hat{m}_{ij}^{n,\vec{\mathrm{vt}}} = \cos(\boldsymbol{r}_{i,n}^\mathrm{v}, \hat{\boldsymbol{r}}_{j,n}^\mathrm{t}). \tag{32}$$

Finally, as illustrated in Fig. 4a, we define the asymmetric-quantized contrastive learning loss for fine-grained alignment as

$$\hat{\ell}_{\mathrm{FA}(n),i}^{\vec{\mathrm{tv}}} = -\log \frac{\exp(\hat{m}_{ii}^{n,\vec{\mathrm{tv}}}/\tau)}{\sum_{j=1}^{|\mathcal{B}|} \exp(\hat{m}_{ij}^{n,\vec{\mathrm{tv}}}/\tau)}, \tag{33}$$

$$\hat{\ell}_{\mathrm{FA}(n),i}^{\vec{\mathrm{vt}}} = -\log \frac{\exp(\hat{m}_{ii}^{n,\vec{\mathrm{vt}}}/\tau)}{\sum_{j=1}^{|\mathcal{B}|} \exp(\hat{m}_{ij}^{n,\vec{\mathrm{vt}}}/\tau)}, \tag{34}$$

$$\hat{\mathcal{L}}_{\mathrm{FA}(n)} = \frac{1}{2|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \left(\hat{\ell}_{\mathrm{FA}(n),i}^{\vec{\mathrm{tv}}} + \hat{\ell}_{\mathrm{FA}(n),i}^{\vec{\mathrm{vt}}} \right), \tag{35}$$

$$\hat{\mathcal{L}}_{\mathrm{FA}} = \frac{1}{N_c} \sum_{n=1}^{N_c} \hat{\mathcal{L}}_{\mathrm{FA}(n)}. \tag{36}$$

The reason we choose the asymmetric-quantized loss (Jang and Cho, 2021) rather than the symmetric one (Wang et al., 2022a) is to optimize the AQS-based (i.e., based on asymmetric quantization similarity, as shown in Fig. 5) retrieval directly. To distinguish the quantization-based fine-grained alignment from the embedding-based counterpart, we call the hugging with optimized quantization learning by *hugging*$^+$. Accordingly, we instantiation of hugging$^+$ is dubbed HUGGINGHASH$^+$.

3.3.4 Learning Objectives

Here we summarize the learning objectives of HUGGINGHASH and HUGGINGHASH$^+$ as follows:

$$\mathcal{L}_{\text{HUGGINGHASH}} = \mathcal{L}_{\text{GA}} + \lambda\mathcal{L}_{\text{FA}} + \gamma\mathcal{R}_{\text{quant}}, \tag{37}$$

$$\mathcal{L}_{\text{HUGGINGHASH}^+} = \mathcal{L}_{\text{GA}} + \lambda\hat{\mathcal{L}}_{\text{FA}} + \gamma\mathcal{R}_{\text{quant}},$$

$$\mathcal{R}_{\text{quant}} = \frac{1}{2|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\left(\left\|\boldsymbol{b}_i^{\text{t}} - \boldsymbol{h}_i^{\text{t}}\right\|_2^2 + \left\|\boldsymbol{b}_i^{\text{v}} - \boldsymbol{h}_i^{\text{v}}\right\|_2^2\right). \tag{38}$$

$\mathcal{R}_{\text{quant}}$ is the quantization loss of hash codes. $\lambda, \gamma > 0$ are the hyper-parameters to balance different loss terms. The hugging and hugging$^+$ frameworks are flexible and compatible. By replacing $\mathcal{L}_{\text{GA}}$ (Eq. (10)) with other hashing objectives, we can easily extend them to other UCMH methods.

3.4 Indexing and Retrieval

3.4.1 Encoding Global and Fine-Grained Indices

Without loss of generality, we take text-to-image retrieval as an example to describe how HUGGINGHASH and HUGGINGHASH$^+$ produce indices (i.e., hash-based representations) in inference.

We encode database images with the image hash encoder, which comprises a patchifier, a ViT, and the image hash module. We denote the global hash codes of the ith image by $\boldsymbol{b}_i^{\text{v}} \in \{-1, +1\}^L$, which can be taken as the global index.

Meanwhile, if the hugging$^+$ training framework is adopted, we can take fine-grained representations as a gift from multi-granularity alignment. In addition to the forward process of global index generation, there is no need for another pass to get fine-grained indices of the same instance. Instead, in the same forward pass, we retain output embeddings of local heads and compress them with corresponding quantization modules. As illustrated in Fig. 4b, at the nth local head, we first apply the isometric rotation matrix $\boldsymbol{P}_n$ to the local image embedding $\boldsymbol{r}_{i,n}^{\text{v}}$, which can be efficiently implemented by Eqs. (28) to (30), and obtain the rotated embedding $\tilde{\boldsymbol{r}}_{i,n}^{\text{v}}$ We divide $\tilde{\boldsymbol{r}}_{i,n}^{\text{v}}$ into M segments, namely $\tilde{\boldsymbol{r}}_{i,n}^{\text{v},1}, \tilde{\boldsymbol{r}}_{i,n}^{\text{v},2}, \cdots, \tilde{\boldsymbol{r}}_{i,n}^{\text{v},M}$. Then, we find the sub-codeword index of each segment. Take the mth segment as an example:

$$k_{i,n}^{\text{v},m} = \arg\max_{0\le k<K} \langle \tilde{\boldsymbol{r}}_{i,n}^{\text{v},m}, \boldsymbol{d}_{n,k}^m \rangle. \tag{39}$$

Next, we collect the indices $\{k_{i,n}^{\text{v},m}\}_{m=1}^M$ and convert them into a binary code vector $\boldsymbol{q}_{i,n}^{\text{v}}$. We take it as quantization codes *w.r.t.* the nth local head.

3.4.2 Ranking in the Hamming Space

Given a text query, we forward it to the text hash encoder, which comprises a tokenizer, a text transformer, and the text hash module. We denote the query hash codes as $\boldsymbol{b}_{\text{q}}^{\text{t}} \in \{-1, +1\}^L$. Hamming distance between $\boldsymbol{b}_{\text{q}}^{\text{t}}$ and $\boldsymbol{b}_i^{\text{v}}$ is defined by

$$d_{\mathbb{H}}(\boldsymbol{b}_{\text{q}}^{\text{t}}, \boldsymbol{b}_i^{\text{v}}) = \frac{1}{2}\left(L - \boldsymbol{b}_{\text{q}}^{\text{t}\top}\boldsymbol{b}_i^{\text{v}}\right). \tag{40}$$

By leveraging bit-wise operators (i.e., XOR), retrieval efficiency can be largely improved.

We rank the database images according to the Hamming distance. The smaller the distance, the higher the ranking. As shown in Fig. 5, if we only consider single-stage retrieval or adopt the vanilla hugging strategy, we directly return the top-ranked IDs *w.r.t.* Hamming distance. Otherwise, we reserve a portion of the top-ranked images in the whole database, $N_{\mathcal{D}'}$ in total, for a fine-grained reranking stage.

3.4.3 Reranking with Fine-Grained Quantization Codes

Suppose the text query $\mathcal{T}_{\text{q}}$ produces an embedding $\boldsymbol{r}_{\text{q},n}^{\text{t}}$ at the nth local head. We first apply the isometric rotation matrix $\boldsymbol{P}_n$ to $\boldsymbol{r}_{\text{q},n}^{\text{t}}$ by Eqs. (28) to (30), and obtain the rotated embedding $\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t}}$. Then, we divide $\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t}}$ into M segments, namely $\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},1}, \tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},2}, \cdots, \tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},M}$. Next, we adopt Asymmetric Quantization Similarity (AQS) (Jégou et al., 2011) as the metric, which computes the similarity between $\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t}}$ and the quantized representation of the ith database item, $\hat{\boldsymbol{r}}_{i,n}^{\text{v}}$, by

$$\text{AQS}(\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t}}, \hat{\boldsymbol{r}}_{i,n}^{\text{v}}) = \sum_{m=1}^{M} \frac{\langle \tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},m}, \hat{\boldsymbol{r}}_{i,n}^{\text{v},m} \rangle}{\left\|\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},m}\right\|_2} \tag{41}$$

$$= \sum_{m=1}^{M} \frac{\langle \tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},m}, \boldsymbol{d}_{n,k_{i,n}^{\text{v},m}}^m \rangle}{\left\|\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},m}\right\|_2}, \tag{42}$$

where $k_{i,n}^{\text{v},m}$ is the sub-codeword index of $\hat{\boldsymbol{r}}_{i,n}^{\text{v},m}$ in the mth sub-codebook, obtained by Eq. (39). We can set up a lookup table $\boldsymbol{T}_{\text{q},n} \in \mathbb{R}^{M\times K}$ *w.r.t.* each $\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t}}$, which stores the precomputed similarities between the segments of $\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t}}$ and all sub-codewords. Specifically, $T_{q,n,k}^m = \langle \tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},m}, \boldsymbol{d}_{n,k}^m \rangle / \|\tilde{\boldsymbol{r}}_{\text{q},n}^{\text{t},m}\|_2$.

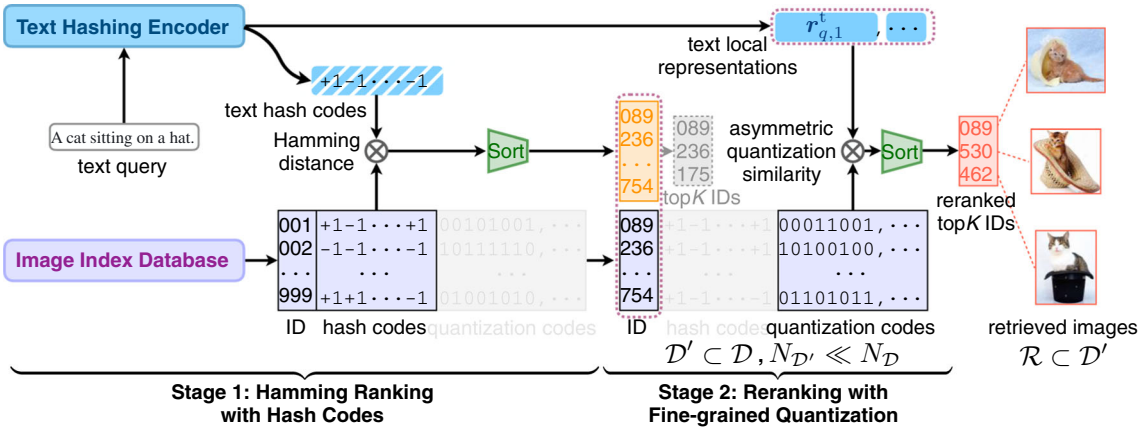

Fig. 5 The hash-based retrieval pipeline. Given a query text, we first use global hash codes to rank the database. We can directly take the top-ranked results or enable a further reranking stage with fine-grained quantization codes

Table 1 Dataset information and settings

Dataset	Text style	Setting reference	#Train	#Query	#Database	Evaluation metric
Text-Image Retrieval						
Flickr25K (Huiskes and Lew, 2008)	Hashtags	Li et al. (2019)	5000	2000	18,015	MAP@All
NUSWIDE (Chua et al., 2009)	Hashtags	Yu et al. (2021a)	5000	2000	184,577	MAP@All
MSCOCO (Lin et al., 2014)	Sentence	Wang et al. (2020b)	122,558	1000	122,558	MAP@All
Wiki (Rasiwasia et al., 2010)	Article	Wang et al. (2020b)	2173	693	2173	MAP@50
Text-Video Retrieval						
MSRVTT (Xu et al., 2016)	Sentence	Zhuo et al. (2022)	9000	1000	1000	R@{1,5,10}, MdR
MSVD (Chen and Dolan, 2011)	Sentence	Zhuo et al. (2022)	1,200	670	670	R@{1,5,10}, MdR

Hence, AQS can be efficiently computed by summing some items from lookup table according to the indices $\{k_{i,n}^{\mathrm{v},m}\}_{m=1}^{M}$ converted from quantization codes $\boldsymbol{q}_{i,n}^{\mathrm{v}}$, i.e.,

$$\mathrm{AQS}(\tilde{\boldsymbol{r}}_{\mathrm{q},n}^{\mathrm{t}}, \hat{\boldsymbol{r}}_{i,n}^{\mathrm{v}}) = \sum_{m=1}^{M} T_{q,n,k_{i,n}^{\mathrm{v},m}}^{m}. \tag{43}$$

The fine-grained similarity between the text query $\mathcal{T}_{\mathrm{q}}$ and fine-grained quantization codes of the ith database images, $\{\boldsymbol{q}_{i,n}^{\mathrm{v}}\}_{n=1}^{N_{\mathrm{c}}}$, is computed by summing up N_{c} local similarity scores, namely

$$s_{\mathbb{Q}}(\mathcal{T}_{\mathrm{q}}, \{\boldsymbol{q}_{i,n}^{\mathrm{v}}\}_{n=1}^{N_{\mathrm{c}}}) = \sum_{n=1}^{N_{\mathrm{c}}} \mathrm{AQS}(\tilde{\boldsymbol{r}}_{\mathrm{q},n}^{\mathrm{t}}, \hat{\boldsymbol{r}}_{i,n}^{\mathrm{v}}). \tag{44}$$

As shown in Fig. 5, if the hugging^{+} strategy is adopted, we can further rerank the filtered subset $\mathcal{D}'$ obtained from Sect. 3.4.2 according to Eq. (44). The higher the score, the higher the ranking. Finally, we return the top-ranked images in the reranked list as retrieval results. Since $N_{\mathcal{D}'} \ll N_{\mathcal{D}}$ and the reranking is quantization-based, it keeps high efficiency of hash-based retrieval.

4 Experiments

4.1 Experimental Setup

4.1.1 Datasets

We conduct experiments on four commonly used text-image datasets in cross-modal hashing:

Flickr25K (Huiskes and Lew, 2008) It contains 25,000 image-text pairs with 24 annotated labels. Each pair contains an image and the associated textual tags. We filter out data without labels and use 20,015 pairs in our experiment. The tag information for each image is represented as a 1,386-dimensional bag-of-words vector.

NUSWIDE (Chua et al., 2009) It provides 186,577 image-tag pairs from the top-10 concepts. The tag information is represented as a 1,000-dimensional bag-of-words vector.

MSCOCO (Lin et al., 2014) It consists of 123,558 image-sentence pairs from 80 object categories. Each image is associated with 4 sentences describing its content. Each text is represented as a 2,000-dimensional bag-of-words vector.

Wiki (Rasiwasia et al., 2010) It composes of 2,866 documents from 10 categories. Each document contains an image

and a text with at least 70 words. An 128-dimensional SIFT feature vector is provided for each image, and each text is represented as a 10-dimensional topic vector.

In addition, following (Zhuo et al., 2022), we conduct experiments on two text-video datasets:

MSRVTT (Xu et al., 2016) It consists of 10,000 video clips, each is annotated with 20 captions. The average video duration is 15 s and the frame rate is 30FPS. We follow the setting of Zhuo et al. (2022) and report the results on the 1K-A test set (Yu et al., 2018).

MSVD (Chen and Dolan, 2011) It contains 1,970 video clips and each clip is associated with about 40 captions. The dataset is split into train, validation, and test sets with 1,200, 100, and 670 clips, respectively. In testing, we follow Zhuo et al. (2022) to select the fifth caption for each clip, resulting in bidirectional one-to-one retrieval.

The data splits are described in Table 1.

4.1.2 Implementation Details

Our implementation is based on PyTorch (Paszke et al., 2019) with 4 NVIDIA GTX 3080Ti (12GB) GPUs. We adopt the standard metric, mean average precision (**MAP**@N), to evaluate text-image retrieval tasks. For text-video retrieval tasks, use recall (**R**@N) and median rank (**MdR**) of the ground-truth items as the evaluation metrics. For comparison, shallow approaches take bag-of-words features and hand-crafted visual descriptors (e.g. SIFT (Lowe, 2004)) as text and image inputs, respectively. Deep approaches use CNN (e.g. AlexNet (Krizhevsky et al., 2012)) features as image inputs. For transformer-based approaches, we use pre-trained BERT (Devlin et al., 2019) ('bert-base-uncased') and ViT (Dosovitskiy et al., 2021) ('vit-base-patch16-224') as default transformers on text-image retrieval tasks. For text-video retrieval tasks, we follow Zhuo et al. (2022); Luo et al. (2022) that use the pre-trained CLIP (Radford et al., 2021) ('clip-vit-base-patch32') to initialize the text and frame encoders. The dimensions of token embeddings are $D^{\mathrm{t}} = D^{\mathrm{v}} = 768$. Maximum numbers of text tokens are set to 128 for text-image retrieval tasks and 32 for text-video retrieval tasks. The dimension of fine-grained alignment space, D, is set to 128 for text-image retrieval tasks and 512 for text-video retrieval tasks. The batch size is set to 32. Following the practice of Luo et al. (2022), we take Adam (Kingma and Ba, 2015) as the optimizer with a learning rate of 1e-7 for pretrained text and image transformers and a learning rate of 1e-4 for other modules. For text-video retrieval tasks, we uniformly select 1 frame per second from a raw video and randomly sample 12 frames from the selected ones as the video input. Other default settings are as follows: (i) The loss weights in Eqs. (37) and (38) are $\lambda = 0.2$ and $\gamma = 1$. (ii) The smoothness factor in Eqs. (4) and (5) is $\alpha = 0.5$. (iii) The scaling factor in Eqs. (23) and (24) is $\beta = 1$. (iv) The temperature factor in contrastive learning objectives is $\tau = 0.2$. (v) The number of active clusters in GhostVLAD is $N_{\mathrm{c}} = 7$. (vi) For the quantization module, we set the codeword number of each sub-codebook, $K = 256$, such that each local embedding is encoded by $M \log_2 K = 8M$ bits (i.e., M bytes). (vii) For text-image retrieval tasks, we set $M = 4$ sub-codebooks in each quantization module, producing 32-bit quantization codes at each local head. For text-video retrieval tasks, we set $M = 32$ sub-codebooks in each quantization module, leading to 256-bit quantization codes at each local head. (viii) The number of Householder transformations in Eq. (27) is $N_{\mathrm{h}} = D/8 = 16$. (ix) In the two-stage retrieval pipeline, we set the reranking size $N_{\mathcal{D}'} = 0.1 N_{\mathcal{D}}$.

4.2 Comparison with State-of-the-art Approaches

4.2.1 On Text-Image Retrieval

The comparison is with 15 UCMH baselines: (i) 5 shallow methods: CVH (Kumar and Udupa, 2011), IMH (Song et al., 2013), CMFH (Ding et al., 2016), FSH (Liu et al., 2017), ACQ (Irie et al., 2015). (ii) 10 SOTA deep methods: DBRC (Hu et al., 2019), UGACH (Zhang et al., 2018), UCH (Li et al., 2019), DJSRH (Su et al., 2019), UKD-SS (Hu et al., 2020), SRCH (Wang et al., 2020b), DSAH (Yang et al., 2020), DGCPN (Yu et al., 2021a), CIRH (Zhu et al., 2023), UCHSTM (Tu et al., 2023). To explore the impact of transformers on UCMH, we adapt the open-sourced implementation of 5 representative baselines, i.e., DJSRH, DSAH, DGCPN, CIRH, and UCHSTM, by using the same backbones as HuggingHash and HuggingHash$^+$. There are two differences between HuggingHash and HuggingHash$^+$: (i) In training, HuggingHash adopts embedding-based fine-grained alignment objective (i.e., Eq. (19)) while HuggingHash$^+$ adopts quantization-based objective (i.e., Eq. (36)). (ii) In inference, HuggingHash only produces global hash codes, while HuggingHash$^+$ further supplies fine-grained quantization codes. HuggingHash executes retrieval by ranking with hash codes only, while HuggingHash$^+$ further enable an efficient reranking stage (as shown in Fig. 5) to refine the retrieval results.

Performance Table 2 reports the MAP results under different numbers of hash bits. The same methods in the 'Transformers' block outperform those in the 'CNN + MLP' block by considerable margins. It verifies that pre-trained transformers provide better modality understandings than CNNs and MLPs, thus contributing to high-quality hash codes. Besides, on all settings, HuggingHash outperforms transformer-based baselines that only consider global alignment. In terms of global alignment for hash codes, although Hugging-

Table 2 Text-image retrieval mean average precision (MAP) results for different numbers of bits on the three datasets

				Dataset (Metric)											
				Flickr25K (MAP@All,↑)			NUSWIDE (MAP@All,↑)			MSCOCO (MAP@All,↑)			Wiki (MAP@50,↑)		
Method	Venue	Type	Hug?	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
Text-to-Image Retrieval															
CVH	2011	S	✗	0.607	0.591	0.581	0.378	0.378	0.379	0.507	0.479	0.446	0.252	0.235	0.171
IMH	2013	S	✗	0.586	0.593	0.589	0.350	0.356	0.372	0.413	0.435	0.443	0.467	0.478	0.453
ACQ	2015	S	✗	–	–	–	0.445	0.420	0.399	0.565	0.561	0.520	–	–	–
CMFH	2016	S	✗	0.611	0.606	0.575	0.487	0.488	0.493	0.453	0.435	0.499	0.595	0.601	0.616
FSH	2017	S	✗	0.589	0.595	0.595	0.395	0.408	0.417	–	–	–	–	–	–
DBRC	2019	D_{CNN}	✗	0.591	0.596	0.598	0.444	0.454	0.459	0.562	0.566	0.573	0.574	0.588	0.598
UGACH	2018	D_{CNN}	✗	0.676	0.692	0.703	0.557	0.562	0.580	0.566	0.595	0.607	0.544	0.555	0.572
UCH	2019	D_{CNN}	✗	0.661	0.667	0.668	–	–	–	0.446	0.469	0.488	–	–	–
DJSRH	2019	D_{CNN}	✗	0.683	0.694	0.717	0.546	0.568	0.599	0.573	0.578	0.584	0.611	0.635	0.646
UKD-SS	2020	D_{CNN}	✗	0.704	0.705	0.714	0.587	0.599	0.599	0.580	0.594	0.603	0.556	0.565	0.578
SRCH	2020b	D_{CNN}	✗	–	–	–	0.553	0.567	0.575	0.600	0.606	0.623	–	–	–
DSAH	2020	D_{CNN}	✗	0.707	0.713	0.728	0.596	0.607	0.620	0.606	0.599	0.620	0.644	0.650	0.660
DGCPN	2021a	D_{CNN}	✗	0.729	0.741	0.749	0.599	0.609	0.614	0.594	0.603	0.616	0.629	0.638	0.641
CIRH	2023	D_{CNN}	✗	0.705	0.709	0.724	0.607	0.619	0.618	0.587	0.592	0.606	0.631	0.639	0.637
UCHSTM	2023	D_{CNN}	✗	0.704	0.721	0.725	0.601	0.615	0.622	0.595	0.601	0.611	0.622	0.630	0.642
DJSRH	2019	D_{Trf}	✗	0.716	0.724	0.725	0.620	0.628	0.633	0.623	0.621	0.627	0.640	0.649	0.652
DSAH	2020	D_{Trf}	✗	0.726	0.729	0.729	0.616	0.627	0.641	0.641	0.636	0.642	0.655	0.661	0.667
DGCPN	2021a	D_{Trf}	✗	0.743	0.756	0.751	0.635	0.645	0.651	0.638	0.649	0.650	0.651	0.658	0.662
CIRH	2023	D_{Trf}	✗	0.730	0.725	0.740	0.616	0.629	0.629	0.632	0.636	0.645	0.653	0.658	0.658
UCHSTM	2023	D_{Trf}	✗	0.733	0.745	0.745	0.611	0.624	0.635	0.636	0.642	0.648	0.646	0.647	0.660
HUGGINGHASH	Ours	D_{Trf}	✓	0.745	0.760	0.766	0.648	0.661	0.659	0.652	0.661	0.663	0.659	0.669	0.675
HUGGINGHASH^{+}	Ours	D_{Trf}	✓	**0.761**	**0.768**	**0.776**	**0.661**	**0.665**	**0.660**	**0.658**	**0.664**	**0.669**	**0.686**	**0.697**	**0.703**

Table 2 continued

Method	Venue	Type	Hug?	Dataset (Metric)											
				Flickr25K (MAP@All,↑)			NUSWIDE (MAP@All,↑)			MSCOCO (MAP@All,↑)			Wiki (MAP@50,↑)		
				16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
Image-to-Text Retrieval															
CVH	2011	S	✗	0.602	0.587	0.578	0.379	0.378	0.377	0.499	0.471	0.443	0.179	0.162	0.153
IMH	2013	S	✗	0.588	0.581	0.585	0.350	0.357	0.371	0.416	0.435	0.443	0.201	0.203	0.204
ACQ	2015	S	✗	–	–	–	0.440	0.416	0.396	0.559	0.553	0.515	–	–	–
CMFH	2016	S	✗	0.659	0.660	0.663	0.483	0.488	0.486	0.442	0.423	0.492	0.251	0.253	0.259
FSH	2017	S	✗	0.590	0.597	0.597	0.400	0.414	0.424	–	–	–	–	–	–
DBRC	2019	D_{CNN}	✗	0.596	0.600	0.602	0.440	0.447	0.456	0.555	0.561	0.564	0.253	0.265	0.269
UGACH	2018	D_{CNN}	✗	0.676	0.693	0.702	0.568	0.583	0.589	0.550	0.584	0.599	0.388	0.392	0.403
UCH	2019	D_{CNN}	✗	0.654	0.669	0.679	–	–	–	0.447	0.471	0.485	–	–	–
DJSRH	2019	D_{CNN}	✗	0.666	0.678	0.699	0.513	0.535	0.566	0.572	0.575	0.579	0.388	0.403	0.412
UKD-SS	2020	D_{CNN}	✗	0.700	0.706	0.709	0.584	0.578	0.586	0.564	0.592	0.601	0.403	0.411	0.416
SRCH	2020b	D_{CNN}	✗	–	–	–	0.544	0.557	0.567	0.598	0.605	0.623	–	–	–
DSAH	2020	D_{CNN}	✗	0.701	0.712	0.722	0.578	0.588	0.607	0.598	0.589	0.609	0.416	0.430	0.438
DGCPN	2021a	D_{CNN}	✗	0.732	0.742	0.751	0.597	0.606	0.615	0.587	0.594	0.612	0.422	0.440	0.446
CIRH	2023	D_{CNN}	✗	0.707	0.719	0.726	0.590	0.601	0.612	0.596	0.603	0.610	0.429	0.433	0.444
UCHSTM	2023	D_{CNN}	✗	0.724	0.732	0.740	0.598	0.608	0.617	0.589	0.600	0.615	0.432	0.437	0.449
DJSRH	2019	D_{Trf}	✗	0.712	0.718	0.723	0.609	0.614	0.617	0.619	0.624	0.627	0.496	0.502	0.511
DSAH	2020	D_{Trf}	✗	0.723	0.727	0.734	0.614	0.624	0.631	0.637	0.639	0.648	0.491	0.489	0.501
DGCPN	2021a	D_{Trf}	✗	0.745	0.750	0.755	0.631	0.637	0.645	0.633	0.641	0.643	0.489	0.498	0.503
CIRH	2023	D_{Trf}	✗	0.730	0.735	0.747	0.612	0.628	0.627	0.629	0.633	0.641	0.473	0.492	0.501
UCHSTM	2023	D_{Trf}	✗	0.737	0.748	0.754	0.624	0.632	0.637	0.631	0.639	0.648	0.482	0.486	0.505
HUGGINGHASH	Ours	D_{Trf}	✓	0.752	0.758	0.764	0.641	0.648	0.651	0.646	0.653	0.662	0.522	0.520	0.526
HUGGINGHASH^{+}	Ours	D_{Trf}	✓	**0.760**	**0.764**	**0.773**	**0.646**	**0.652**	**0.653**	**0.655**	**0.660**	**0.668**	**0.537**	**0.535**	**0.540**

The 'S', 'D_{CNN}', 'D_{Trf} in the 'Type' column represent the shallow, the 'CNN+MLP' and the transformer-based methods, respectively. The ✗ and ✓ in 'Hug?' column mark whether the method adopts fine-grained alignment. Best values are bolded

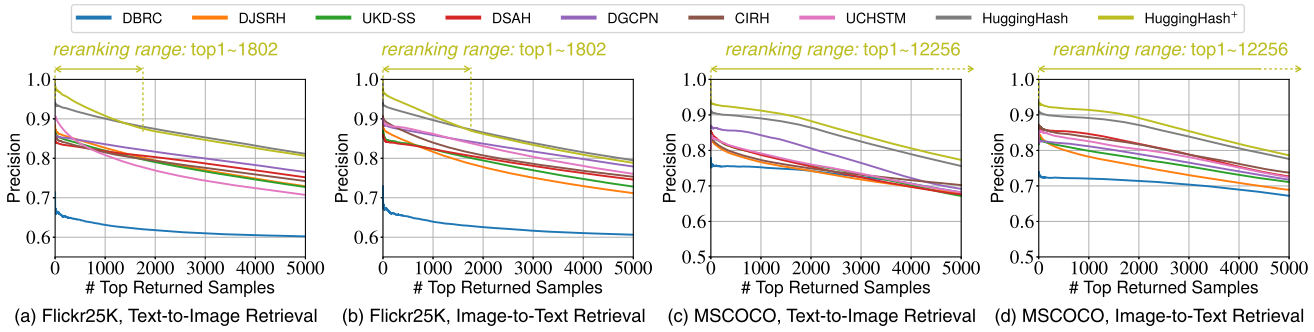

Fig. 6 Top-N precision curves of 64-bit hashing methods

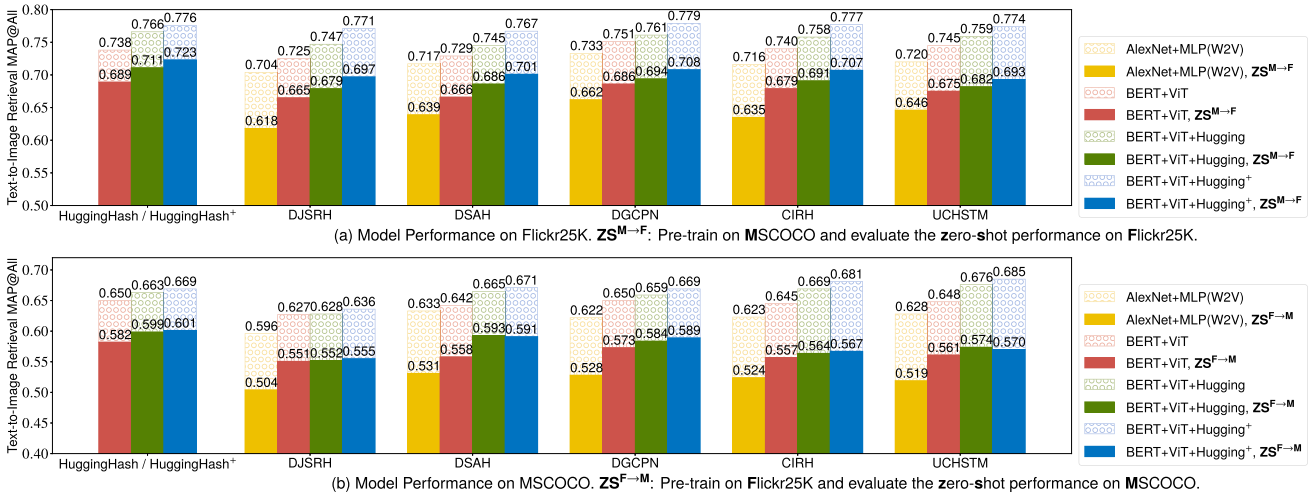

Fig. 7 Multi-dataset (Flickr25K-MSCOCO) evaluation with different 64-bit UCMH methods. Transformers and the *hugging* improve generalizability and robustness. Besides, HUGGINGHASH shows the best zero-shot results

HASH adopts a simple contrastive learning objective (i.e., Eq. (10)) that is much simpler than the baselines, the superior results it shows indicate the effectiveness of exploring multi-granularity alignment with transformers beyond global alignment itself. Moreover, by making better use of fine-grained quantization codes, HUGGINGHASH$^+$ can further surpass HUGGINGHASH, especially at those top positions, as we can learn from the MAP@50 results on the Wiki dataset. It suggests the value of leveraging fine-grained semantics in transformers. We also illustrate the precision curves of different approaches to give a more intuitive understanding. As shown in Fig. 6, HUGGINGHASH reaches higher precision than other baselines significantly. Within the range of reranking, HUGGINGHASH$^+$ refines the retrieval results such that the matched items are ranked higher precisely. Therefore, we can see a further gain within the reranking range. Note that HUGGINGHASH$^+$ provides both coarse-grained and fine-grained representations by learning one unified model. This property is beneficial to practical search systems (Asadi and Lin, 2013) where ranking is multi-stage and consists of a series of independent models. On the other hand, as reranking will consume more memory (or storage) and computation compared to one-stage hash-based retrieval, the efficacy-efficiency tradeoff should depend on the application scenario.

Transferability Transferability is an important but often ignored target in practice, reflecting the domain generalizability from offline training to online serving. Here we conduct a multi-dataset (Flickr25K-MSCOCO) evaluation with five representative baselines and the proposed models. We compare *standard* (i.e., train and test on the same dataset) and *zero-shot* performance (i.e., train and test on different datasets). We also investigate different backbones. To deal with different vocabularies between two datasets, we replace the bag-of-words features with the word2vec (Le and Mikolov, 2014) features as the text inputs. "BERT+ViT" indicates transformer-based variants using global alignment. "BERT+ViT +Hugging" and "BERT+ViT +Hugging$^+$" indicate the variants with hugging or hugging$^+$, respectively.

The results are shown in Fig. 7. We can learn that transformers not only boost in-domain performance but also improve generalizability. The proposed hugging and hugging$^+$ strategies can further improve the zero-shot performance in most cases. Besides, although combining *hugging* with SOTA baselines yields competitive results, we notice

that HUGGINGHASH and HUGGINGHASH$^+$ show better zero-shot performance. We conjecture that contrastive learning objectives help to produce more transferable hash codes.

4.2.2 On Text-Video Retrieval

The comparison is with 14 baselines: (i) 7 representative embedding-based text-video retrieval methods: CE (Liu et al., 2019b), MMT (Gabeur et al., 2020), Support-Set (Patrick et al., 2021), HiT (Liu et al., 2021a), CLIP (Radford et al., 2021), T2VLAD (Wang et al., 2021b), and CLIP4Clip (Luo et al., 2022). (ii) 1 hash-based text-video retrieval method with non-transformer backbone: S^{2}Bin (Qi et al., 2021). (iii) 5 UCMH methods originally designed for text-image retrieval: DJSRH (Su et al., 2019), DSAH (Yang et al., 2020), DGCPN (Yu et al., 2021a), CIRH (Zhu et al., 2023), and UCHSTM (Tu et al., 2023). (iv) 1 state-of-the-art transformer-based UCMH method tailored for text-video retrieval: CLIP4Hashing (Zhuo et al., 2022). In particular, we follow Zhuo et al. (2022) to adopt pre-trained CLIP (Radford et al., 2021) as the text and video frame encoder, which is also a popular practice in text-video retrieval literature (Luo et al., 2022). Although pretrained CLIP is well-aligned for text-image tasks, we argue that it is not aligned for text-video retrieval. It needs to be improved to understand the correspondence between language and temporal dynamics, e.g. whether an object is moving from left to right or otherwise. In HUGGINGHASH and HUGGINGHASH$^+$, we design a temporal self-attention layer based on the image-level frame embeddings to enhance this aspect. We also adapt the UCMH baselines in (iii) by using the same backbones as HUGGINGHASH and HUGGINGHASH$^+$.

Performance Table 3 presents the results under different numbers of hash bits. Some findings are as below. (i) CLIP provides useful cross-modal knowledge to bridge the modality gap but is not well-aligned for text-video retrieval. Though vanilla CLIP alone can outperform CE and SupportSet, it is insufficient to understand text-video correspondence and shows inferior results than HiT on most metrics. Through temporal-aware training, CLIP4CLIP shows higher recall than other embedding-based approaches; 2048-bit CLIP4Hashing outperforms vanilla CLIP. (ii) The text-image hashing objectives are inadequate for one-to-one text-video retrieval due to the unsatisfied basic assumption. Despite the same backbone, we can see that all the compared baselines from text-image retrieval achieve significantly lower recall than HUGGINGHASH. One of the reasons is that most text-image hashing objectives are based on the assumption that one query is associated with multiple matched items given the same labels. Differently, in text-video retrieval, one query is expected to match only one item, which leads to an extremely sparse similarity matrix, hence limiting the performance of text-image hashing

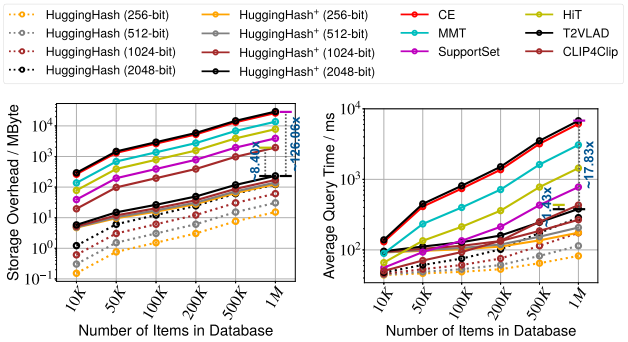

Fig. 8 Storage (memory) overhead and average query time of different text-to-video retrieval models

approaches. (iii) Hugging is more effective than handshaking. The strong baseline, CLIP4Hashing, can be regarded as a handshaking approach as it only designs global alignment. Our HUGGINGHASH outperforms it by considerable margins. Besides, T2VLAD develops multi-granularity similarity in training and inference, which can also be regarded as a practice of hugging. Without large-scale cross-modal pretraining, T2VLAD even outperforms vanilla CLIP. The success highlights the efficacy of multi-granularity similarity. Nevertheless, we note that it depends on high-dimensional continuous embeddings of global and local semantics, which incurs much more computation and storage overhead. The efficiency is a major weakness of T2VLAD, as we will show below. (iv) Hugging$^+$ can further improve retrieval performance beyond hugging. Similar to the observation on the text-image retrieval, we can find that HUGGINGHASH$^+$ shows relatively better results than HUGGINGHASH in most cases. The advantage of reranking is more pronounced in the case of shorter code length, e.g. 256 and 512, while fading for longer bit length, e.g. 2048. Two main factors contribute to the observed results. First, longer hash code lengths enhance semantic information capacity, making fine-grained reranking less necessary for global retrieval. Besides, as hash codes become longer, the impact of complementing them with fine-grained quantization codes diminishes, particularly evident with 2048-bit hash codes.

Comparison with T2VLAD Using the Same Encoders Since HUGGINGHASH, HUGGINGHASH$^+$ and T2VLAD share similar network modules, comparing them using the same text encoder and visual encoder can be worthy. We report the results in Table 4. As the source code for T2VLAD is currently unavailable, we replicated the model based on our HUGGINGHASH. Specifically, since we only utilized CLIP features for video, we set the global alignment to have one expert. Notably, the original T2VLAD paper employed a bidirectional max-margin ranking loss with a margin of 0.02 for training. To align our experiments, we also present results using the contrastive learning loss for training, denoted as T2VLAD*.

Table 3 Text-video retrieval recall and median rank results for different numbers of bits on the three datasets

			Dataset→	MSRVTT								MSVD							
				Text-to-Video Retrieval				Video-to-Text Retrieval				Text-to-Video Retrieval				Video-to-Text Retrieval			
Method	Year	Hugs?	#bits	R@1,↑	R@5,↑	R@10,↑	MdR,↓	R@1,↑	R@5,↑	R@10,↑	MdR,↓	R@1,↑	R@5,↑	R@10,↑	MdR,↓	R@1,↑	R@5,↑	R@10,↑	MdR,↓
CE	2019b	✗	–	0.209	0.488	0.624	6	0.206	0.503	0.640	5	0.198	0.490	0.638	6	-	-	-	-
MMT	2020	✗	–	0.244	0.560	0.678	4	0.246	0.540	0.671	4	–	–	–	–	–	–	–	–
Support-Set	2021	✗	–	0.220	0.468	0.599	6	–	–	–	–	0.230	0.528	0.658	5	0.273	0.507	0.608	5
HiT	2021a	✗	–	0.266	0.551	0.675	3	0.274	0.563	0.677	3	–	–	–	–	–	–	–	–
CLIP	2021	✗	–	0.307	0.531	0.626	5	0.260	0.517	0.624	5	0.439	0.721	0.813	2	0.435	0.754	0.850	2
T2VLAD	2021b	✓	–	0.295	0.590	0.701	4	0.318	0.600	0.711	3	–	–	–	–	–	–	–	–
CLIP4Clip	2022	✗	–	0.431	0.704	0.808	2	0.431	0.705	0.812	2	0.536	0.819	0.884	1	0.542	0.851	0.907	1
S^2Bin	2021	✗	256	0.079	0.225	0.323	31	–	–	–	–	–	–	–	–	–	–	–	–
DJSRH	2019	✗	256	0.069	0.202	0.237	66	0.066	0.189	0.226	65	0.082	0.181	0.249	37	0.135	0.211	0.256	28
			512	0.089	0.239	0.273	47	0.084	0.221	0.266	47	0.141	0.284	0.378	17	0.177	0.294	0.346	17
			1024	0.138	0.344	0.368	27	0.135	0.350	0.378	28	0.194	0.372	0.484	12	0.249	0.357	0.413	14
			2048	0.192	0.424	0.451	22	0.177	0.345	0.419	21	0.280	0.479	0.571	7	0.247	0.420	0.501	8
DSAH	2020	✗	256	0.077	0.219	0.250	64	0.075	0.210	0.241	63	0.105	0.200	0.262	32	0.151	0.220	0.261	26
			512	0.100	0.262	0.289	43	0.095	0.245	0.283	43	0.181	0.314	0.397	15	0.195	0.306	0.353	16
			1024	0.153	0.373	0.387	25	0.155	0.390	0.405	25	0.249	0.411	0.509	10	0.278	0.374	0.423	13
			2048	0.212	0.460	0.474	20	0.201	0.373	0.441	18	0.365	0.532	0.602	6	0.275	0.439	0.512	8
DGCPN	2021a	✗	256	0.050	0.171	0.246	50	0.049	0.163	0.216	56	0.149	0.314	0.394	16	0.257	0.374	0.440	11
			512	0.083	0.234	0.284	51	0.097	0.261	0.308	40	0.223	0.419	0.511	10	0.405	0.552	0.598	6
			1024	0.133	0.386	0.449	19	0.144	0.361	0.413	21	0.283	0.489	0.567	7	0.491	0.627	0.658	4
			2048	0.202	0.468	0.538	14	0.180	0.384	0.459	13	0.326	0.545	0.610	6	0.580	0.660	0.704	3

Table 3 continued

			Dataset→	MSRVTT								MSVD							
				Text-to-Video Retrieval				Video-to-Text Retrieval				Text-to-Video Retrieval				Video-to-Text Retrieval			
Method	Year	Hugs?	#bits	R@1,↑	R@5,↑	R@10,↑	MdR,↓	R@1,↑	R@5,↑	R@10,↑	MdR,↓	R@1,↑	R@5,↑	R@10,↑	MdR,↓	R@1,↑	R@5,↑	R@10,↑	MdR,↓
CIRH	2023	✗	256	0.051	0.172	0.247	50	0.052	0.169	0.221	55	0.172	0.334	0.406	15	0.298	0.397	0.453	10
			512	0.085	0.237	0.286	50	0.101	0.269	0.314	38	0.259	0.445	0.527	9	0.467	0.584	0.615	5
			1024	0.135	0.390	0.452	19	0.150	0.373	0.421	20	0.327	0.518	0.583	7	0.565	0.663	0.676	4
			2048	0.204	0.471	0.540	13	0.187	0.393	0.466	13	0.378	0.579	0.629	5	0.672	0.700	0.725	3
UCHSTM	2023	✗	256	0.047	0.161	0.237	52	0.046	0.154	0.209	58	0.173	0.334	0.406	15	0.265	0.379	0.442	11
			512	0.078	0.222	0.275	53	0.090	0.245	0.296	42	0.259	0.445	0.526	9	0.415	0.558	0.601	6
			1024	0.125	0.366	0.434	20	0.135	0.343	0.400	22	0.329	0.519	0.584	7	0.503	0.633	0.661	4
			2048	0.188	0.442	0.519	14	0.168	0.368	0.446	14	0.374	0.576	0.627	5	0.596	0.667	0.708	3
CLIP4Hashing	2022	✗	256	0.149	0.274	0.335	34	0.147	0.274	0.337	33	0.344	0.524	0.619	5	0.352	0.539	0.629	5
			512	0.225	0.397	0.478	13	0.227	0.374	0.473	13	0.455	0.662	0.770	**2**	0.495	0.734	0.820	**2**
			1024	0.323	0.521	0.613	5	0.329	0.515	0.597	5	0.524	0.735	0.829	**1**	0.527	0.772	0.839	**1**
			2048	0.376	0.569	0.663	3	0.370	0.583	0.660	3	0.545	0.782	0.873	**1**	0.545	0.788	0.871	**1**
			256	0.198	0.433	0.532	10	0.198	0.433	0.535	9	0.376	0.588	0.701	**4**	0.416	0.610	0.709	4
			512	0.365	0.595	0.675	**4**	0.370	0.600	0.675	**4**	0.459	0.715	0.818	**2**	0.503	0.745	0.822	**2**
			1024	0.439	0.684	0.778	**2**	0.440	0.682	0.776	**2**	0.531	0.766	0.860	**1**	0.539	0.808	0.868	**1**
HUGGINGHASH	Ours	✓	2048	**0.457**	**0.722**	0.799	**2**	**0.460**	0.723	**0.800**	**2**	**0.545**	0.819	0.884	**1**	**0.557**	0.845	0.895	**1**
			256	**0.207**	**0.444**	**0.544**	**9**	**0.212**	**0.448**	**0.554**	**8**	**0.381**	**0.595**	**0.711**	**4**	**0.426**	**0.623**	**0.726**	**3**
			512	**0.374**	**0.604**	**0.685**	**4**	**0.377**	**0.609**	**0.685**	**4**	**0.465**	**0.724**	**0.830**	**2**	**0.511**	**0.756**	**0.836**	**2**
			1024	**0.441**	**0.687**	**0.781**	**2**	**0.442**	**0.685**	**0.779**	**2**	**0.532**	**0.768**	**0.864**	**1**	**0.540**	**0.811**	**0.872**	**1**
HUGGINGHASH$^+$	Ours	✓	2048	0.456	**0.722**	**0.802**	**2**	0.458	**0.724**	**0.800**	**2**	0.544	**0.820**	**0.885**	**1**	**0.557**	**0.847**	**0.897**	**1**

The ✗ and ✓ in 'Hug?' column mark whether the method adopts fine-grained alignment. Best values are bolded

Table 4 Comparison of HUGGINGHASH, HUGGINGHASH$^+$, and T2VLAD (Wang et al., 2021b) on the MSRVTT dataset

Model	Loss Type	Similarity	Index Size / Byte	#Time / Sim (Retrieval)	Text-to-Video Retrieval				Video-to-Text Retrieval			
					$D = 2^8$	$D = 2^9$	$D = 2^{10}$	$D = 2^{11}$	$D = 2^8$	$D = 2^9$	$D = 2^{10}$	$D = 2^{11}$
HUGGINGHASH	Contrastive	Global Only	$D/8$	$D \cdot T_{\text{xor}}$	0.357	0.527	0.616	0.641	0.358	0.531	0.615	0.643
HUGGINGHASH$^+$	Contrastive	Global/Local	$D/8 + 224$	$D \cdot T_{\text{xor}}$	0.368	0.537	0.619	0.642	0.375	0.540	0.618	0.643
T2VLAD	Ranking	Global+Local	$4D + 14336$	$(D + 3584) \cdot T_{\text{mul}}$	0.543	0.586	0.608	0.627	0.565	0.590	0.604	0.626
T2VLAD*	Contrastive	Global+Local	$4D + 14336$	$(D + 3584) \cdot T_{\text{mul}}$	0.564	0.595	0.622	0.644	0.578	0.602	0.625	0.644

All models utilize the same network and parameter settings, where there are 7 active clusters in the 512-dimensional (fixed) local alignment module and the bit length of quantization codes is fixed to 256. The term 'Similarity' denotes the representation used for computing correlation scores during inference. 'Index Size / Byte' indicates the storage overhead per each item in the database, measured in bytes, i.e., units of 8 bits. Here, D represents the length of **global** hash codes or the dimensionality of **global** embeddings. We suppose each floating-point number occupies 4 bytes. '#Time / Sim (Retrieval)' represents the time required per similarity computation during global (stage 1) retrieval. T_{xor} and T_{mul} represent the time taken for one bit-wise XOR operation and one floating-point multiplication, respectively. *: replacing the bidirectional max-margin ranking loss in the original T2VLAD model with a contrastive learning objective. Evaluation metric for video-text retrieval: R@{1,5,10}

We can learn from Table 4 that T2VLAD outperforms HUGGINGHASH (or HUGGINGHASH$^+$) notably for lower values of D. This outcome reflects the higher capacity of T2VLAD's representation compared to HUGGINGHASH (or HUGGINGHASH$^+$). For instance, at $D = 256$, T2VLAD uses higher-dimensional embeddings, including 256-dimensional floating-point and 3584-dimensional local embeddings. In contrast, HUGGINGHASH (or HUGGINGHASH$^+$) relies on 256-bit binary global hash codes. As D increases, HUGGINGHASH and HUGGINGHASH$^+$ gradually approach or even surpass T2VLAD variants. For instance, at $D = 2048$, HUGGINGHASH and HUGGINGHASH$^+$ surpass T2VLAD, and T2VLAD* slightly outperforms them. This can be attributed to the risk of overfitting in T2VLAD's high-dimensional feature space and the potential impact of global–local similarity balance on T2VLAD's performance. In contrast, HUGGINGHASH and HUGGINGHASH$^+$ separate these aspects, avoiding competition.

In terms of global retrieval, HUGGINGHASH (or HUGGINGHASH$^+$) employs D bits per item and XOR operations over D bits for similarity calculation. In contrast, T2VLAD utilizes a D-dimensional global embedding and 7 512-dimensional local embeddings, incurring over 32 times the storage and memory overhead of hash-based methods. T2VLAD also involves more time-consuming floating-point multiplications for similarity calculations. Consequently, T2VLAD exhibits significantly longer retrieval times. For real-world large-scale retrieval scenarios, efficiency is crucial alongside performance. Notably, HUGGINGHASH$^+$ reduces overhead by using lightweight quantized codes instead of embeddings, and reorders only a small portion (e.g. top 10%) of the globally hashed list based on quantized representations, resulting in faster retrieval.

Although HUGGINGHASH and T2VLAD share similar network modules and aligned model settings from table 4, the results serve as references rather than fair comparisons. Fair comparison remains challenging due to differing design considerations and operational mechanisms. HUGGINGHASH and HUGGINGHASH$^+$ prioritize efficient retrieval, focusing on representational size constraints and semantic preservation within limited resources. Our fine-grained alignment enhances global alignment for improved cross-modal semantics in hash codes without altering generation and retrieval. The quantization option enhances reranking for performance refinement while maintaining efficiency. T2VLAD prioritizes performance optimization, resulting in disparities in representation and efficiency. Its straightforward global–local fusion employs fine-grained alignment, capturing multi-granularity semantics at the cost of efficiency.

Efficiency of hash-based retrieval Efficiency is an important target for retrieval since it highly relates to the *scalability* of the search system. Here we show results about this

Table 5 Ablation study on MSCOCO and MSRVTT datasets.

	Dataset (Metrics) →		MSCOCO (MAP@All)						MSRVTT (Geometric Mean of R@{1,5,10})							
			Text-to-Image Retrieval			Image-to-Text Retrieval			Text-to-Video Retrieval				Video-to-Text Retrieval			
ID	Type	Variant Name	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	256 bits	512 bits	1024bits	2048bits	256 bits	512 bits	1024bits	2048bits
1	**A**	HUGGINGHASH	0.652	0.661	0.663	0.646	0.653	0.662	0.357	0.527	0.616	0.641	0.358	0.531	0.615	0.643
2	**A**	$\text{HUGGINGHASH}_{\setminus \mathcal{L}_{\text{FA}}}$	0.633	0.644	0.650	0.631	0.636	0.644	0.333	0.507	0.598	0.628	0.328	0.501	0.591	0.621
3	**D**	$\text{HUGGINGHASH}_{\cup \text{Emb\_ReR}}$	0.663	0.673	0.677	0.655	0.666	0.674	0.371	0.539	0.629	0.643	0.372	0.543	0.629	0.643
4	**C**	$\text{HUGGINGHASH}_{\cup \text{PQ\_ReR}}$	0.655	0.662	0.665	0.649	0.658	0.664	0.365	0.526	0.616	0.643	0.369	0.530	0.616	0.645
5	**C**	HUGGINGHASH^{+}	0.658	0.664	0.669	0.655	0.660	0.668	0.368	0.537	0.619	0.642	0.375	0.540	0.618	0.643
6	**B**	$\text{HUGGINGHASH}^{+}_{\setminus \mathcal{L}_{\text{GA}}}$	0.657			0.655			0.619				0.622			
7	**B**	$\text{HUGGINGHASH}^{+}_{\text{FG\_R}}$	0.679*	0.678*	0.676*	0.670*	0.674*	0.675*	0.636*	0.628*	0.622*	0.623*	0.633*	0.630*	0.625*	0.624*
8	**A**	$\text{HUGGINGHASH}^{+}_{\setminus \text{ReR}}$	0.656	0.661	0.667	0.651	0.656	0.666	0.353	0.529	0.612	0.638	0.360	0.527	0.616	0.644
9	**C**	$\text{HUGGINGHASH}^{+}_{\setminus \mathcal{P}_n}$	0.654	0.662	0.665	0.652	0.656	0.663	0.362	0.533	0.616	0.640	0.370	0.535	0.616	0.641
10	**E**	$\text{HUGGINGHASH}^{+}_{\text{Fuse\_R}}$	0.662	0.659	0.665	0.652	0.664	0.662	0.403	0.530	0.615	0.627	0.414	0.536	0.619	0.630
(allocate all bits for handshaking)			240bits	256bits	288bits	240bits	256bits	288bits	2048bits	2304bits	2816bits	3840bits	2048bits	2304bits	2816bits	3840bits
2*	**A**	$\text{HUGGINGHASH}_{\setminus \mathcal{L}_{\text{FA}}}$	0.672	0.677	0.680	0.664	0.672	0.675	0.628	0.630	0.636	0.644	0.621	0.627	0.635	0.642

We number the model variants for easy reference in the main text. We categorize variants according to the retrieval process and record them in the "Type" column. List of abbreviations: FA: Fine-grained Alignment; GA: Global Alignment; R: Ranking; ReR: Reranking. '\' and '∪' denotes removing and adding a module, respectively; *: The bit length refers to the hash code length of global alignment in training, whereas global hash codes are ignored in inference, and retrieval is based on fixed-length (i.e., 224-bit on MSCOCO and 1792-bit on MSRVTT) fine-grained quantization codes. Note that $\text{HUGGINGHASH}^{+}_{\setminus \mathcal{L}_{\text{GA}}}$ removes global alignment in training and does retrieval only with fine-grained quantization codes; therefore, it is independent of the length of global hash codes

rarely investigated aspect in text-video retrieval. The comparison is with 6 embedding-based text-to-video retrieval: CE, MMT, SupportSet, HiT, T2VLAD, and CLIP4Clip, on the MSRVTT dataset. The manners of models are as follows: (i) CE represents an instance by 9 768-dimensional embeddings and a 9-dimensional weighting vector; (ii) MMT represents an instance by 7 512-dimensional embeddings and a 7-dimensional weighting vector; (iii) SupportSet represents an instance by a 1024-dimensional embedding; (iv) HiT represents an instance by a 2048-dimensional embedding; (v) T2VLAD represents an instance by 9 768-dimensional local embeddings and a 768-dimensional global embedding; (vi) CLIP4Clip represents an instance by a 512-dimensional embedding; (vii) Hash-based approaches represent an instance by a fixed-length hash code vector; we take HuggingHash as a showcase. A 256-bit code can be encoded by 32 bytes for compact storage; (viii) HuggingHash$^+$ represents an instance by a fixed-length global hash code and 7 256-bit local quantization codes. Since MSRVTT only contains 10k videos, we duplicate the videos to simulate a large database. We evaluate: (i) the average query time, including the time for text encoding time on GPU and nearest neighbor search on CPU; and (ii) the storage overhead for offline-computed video representations. We experiment with an NVIDIA GTX 3080 Ti (12GB) and Intel® Xeon® Platinum 8269CY CPU @ 2.50GHz (104 cores). The results are shown in Fig. 8. We can see that hash-based approaches reduce storage (or memory) overhead and accelerate retrieval, especially on large-scale data. For example, on a 1 M-size database, HUGGINGHASH (2048-bit) consumes $\sim$241$\times$ and $\sim$16$\times$ less of storage and accelerate $\sim$23.6$\times$ and $\sim$1.5$\times$ of query speed, compared with T2VLAD and CLIP4Clip, respectively. Compared with vanilla hash-based retrieval, though HUGGINGHASH$^+$ consumes more computation and storage due to the fine-grained quantization-based reranking, overall, it still achieves a good trade-off between efficacy and efficiency.

4.3 Model Analyses

4.3.1 Ablation Study

Recall that HUGGINGHASH is trained with multi-granularity alignment objectives (Eq. (37)) and uses global hash codes to rank and retrieve items. HUGGINGHASH$^+$ extends HUGGINGHASH by replacing embedding-based fine-grained alignment loss with quantization-based loss (Eq. (38)) and further reranks top-10% hash-ranked results with fine-grained quantization codes (Fig. 5).

To understand the contributions of different modules in HUGGINGHASH and HUGGINGHASH$^+$, we construct 8 variants for analysis. The modifications of these variants are listed below. 3 variants of HUGGINGHASH: (i) HUGGINGHASH$_{\setminus\mathcal{L}_{\mathrm{FA}}}$ removes the embedding-based fine-grained alignment loss (Eq. (19)), producing a handshaking model. (ii) HUGGINGHASH$_{\cup\mathrm{Emb_ReR}}$ enables top-10% reranking in retrieval by leveraging fine-grained embeddings. (iii) HUGGINGHASH$_{\cup\mathrm{PQ_ReR}}$ enables top-10% reranking in retrieval by leveraging fine-grained embeddings and OPQ (Ge et al., 2013). We adopt the FAISS (Johnson et al., 2021) implementation of OPQ as a post-compression on the extracted embeddings. 5 variants of HUGGINGHASH$^+$: (i) HUGGINGHASH$^+_{\setminus\mathcal{L}_{\mathrm{GA}}}$ removes the global hash alignment loss (Eq. (10)) and only trains a fine-grained quantization model. Since it does not produce global hash codes, in inference, we use the quantization codes to rank the whole database. (ii) HUGGINGHASH$^+_{\mathrm{FG_R}}$ follows the hugging$^+$ training strategy but changes the retrieval process. Instead of the two-stage ranking, it leverages fine-grained quantization codes to rank the whole database. (iii) HUGGINGHASH$^+_{\setminus\mathrm{ReR}}$ follows the hugging$^+$ training strategy but disables the top-10% reranking in inference. It slightly differs from standard HUGGINGHASH on the fine-grained alignment of training. Concretely, it uses a quantization-based loss (Eq. (36)) while HUGGINGHASH uses an embedding-based loss (Eq. (19)). (iv) HUGGINGHASH$^+_{\setminus\mathcal{P}_n}$ removes the rotation matrix, $\mathcal{P}_n$ in Eq. (25), of each local quantization module. (v) HUGGINGHASH$^+_{\mathrm{Fuse_R}}$ only changes the retrieval process. Instead of the two-stage ranking, it directly fuses multi-granularity similarity scores with the (IV) strategy in fig. 10 to rank the whole database. (vi) To further investigate the global–local bit allocation strategy, we attempt to reallocate all the bits from the fine-grained quantization in the hugging$^+$ mechanism to global hashing, thereby deriving a long-bit handshaking model. We designate the new model ID as 2*. We also categorize different variants into 5 types according to the retrieval process: (**A**) Rank the whole database with global hash codes. (**B**) Rank the whole database with fine-grained quantization codes. (**C**) Rank the whole database with global hash codes, and then rerank top-10% results with fine-grained quantization codes. (**D**) Rank the whole database with global hash codes, and then rerank the top-10% results with fine-grained embeddings. (**E**) Rank the whole database by fusing multi-granularity similarities from global hash codes and fine-grained quantization codes.

Performance results on MSCOCO (text-image) and MSRVTT (text-video) are reported in table 5. We number the variants with IDs for easy reference. Similar to Fig. 8, we also measure the storage and computational efficiency of different variants to give a more comprehensive comparison, as shown in Fig. 9. We can obtain several conclusions from the ablation results. (i) Global alignment and fine-grained alignment promote each other; both of them are indispensable for achieving a positive synergy in hugging and hugging$^+$. Comparing Models 1 and 2, we can learn that the fine-grained alignment

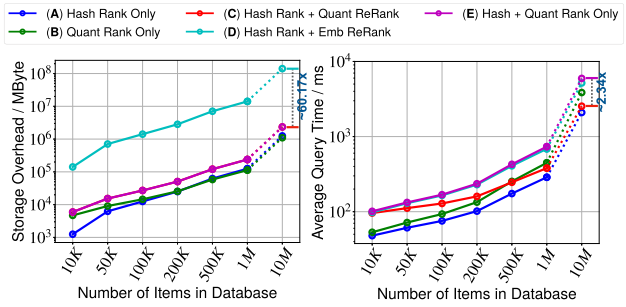

Fig. 9 Storage (memory) overhead and average query time of different types of HUGGINGHASH (or HUGGINGHASH$^+$) variants. Showcased task: text-to-video retrieval. The lengths of hash codes and quantization codes are 2048-bit and 7×256-bit, respectively. The results under 10 million items are extrapolated

loss is significant in improving the quality of hash codes. Note that Models 1 and 2 share the same retrieval type and are consistent in retrieval efficiency; in other words, hugging will *not* bring extra inference overhead to global hash-based retrieval. In Sect. 4.3.2, we will conduct more empirical analyses to understand this implicit efficacy of hugging. Besides, Models 6 and 7 share the same type that ranks the whole database with 224-bit (7×32) fine-grained quantization codes on MSCOCO and with 1792-bit (7×256) codes on MSRVTT, whereas Model 6 learns by fine-grained alignment only and Model 7 learn by hugging$^+$. As Model 7 outperforms Model 6 by considerable margins, we learn that global alignment can improve fine-grained alignment as well. (ii) Quantization in hugging$^+$ facilities the efficient usage of fine-grained semantics. In contrast to Model 1, Model 5 leverages fine-grained semantics and improves retrieval performance effectively. Although it relatively underperforms Model 3, the major benefit of high efficiency by integrating quantization should not be overlooked. Take text-to-video reranking as a showcase, as shown in Fig. 9, Model 5 consumes only $\sim 1/60$ the storage (memory) overhead and half of retrieval time the Model 3 requires in a database of 10 million. (iii) Jointly learning quantization and fine-grained alignment in hugging$^+$ helps to achieve a better trade-off between efficacy and efficiency. By learning fine-grained embeddings and using OPQ to post-compress the learned embeddings, Model 4 shows slight performance gains on 1 in most scenarios, but is still inferior to 5 that jointly learns optimized quantization and fine-grained alignment. The underlying reason is that end-to-end learning of quantized representations can reduce semantic distortion, thus producing better fine-grained quantization codes while maintaining high efficiency. (iv) In some scenarios, quantization can serve as another regularization to enhance the synergy between global and fine-grained alignment. For example, Model 8 with quantization-based fine-grained alignment loss shows superior results than Model 1 on MSCOCO, but this phenomenon does not hold true on MSRVTT. (v) Shorter bit-length retrieval benefits more from the fine-grained reranking. Comparing Models 5 and 8, we learn that reranking improves retrieval performance in most cases. In particular, on MSRVTT, we observe that reranking shows more performance gains on shorter bit-lengths, e.g. 256- and 512-bit, while its efficacy fades on longer ones, e.g. 2048-bit video-to-text retrieval. The reason is that we fix the bit-length of fine-grained quantization codes by default, e.g. 7×256-bit on MSRVTT, which is less effective in providing more complementary semantics beyond long and sufficient global hash codes. (vi) Two-stage pipeline is more efficient and robust than multi-granularity fusion-based retrieval. To exploit multi-granularity semantics for retrieval, another natural design is Model 10, which fuses global and fine-grained similarity scores for more precise estimation. However, we can see from Fig. 9 that fusion-based retrieval requires more than twice the time of two-stage retrieval, which becomes expensive in large-scale scenarios. Moreover, the retrieval performance is also sensitive *w.r.t.* the adopted fusion strategy. As shown in Fig. 10, the efficacy of different fusion strategies varies in different cases, which can increase the labor of careful choosing. Under the default fusion strategy (i.e., (IV) in Fig. 10), the gains from fusion are still unsure. For example, compared with a handshaking Model 8, Model 10 does not gain performance in half of the retrieval tasks on MSCOCO. On MSRVTT, while it brings remarkable improvements under 256 bits thanks to the fine-grained similarity scores, the advantages of Model 10 over Model 8 shrink rapidly as bit-length raises and even turn into clear disadvantages under 2048 bits due to biased similarity estimation. In contrast, Model 5 with the two-stage strategy shows more stable improvement over Model 8, thus preferable to the fusion-based strategy. (vii) Hugging and hugging$^+$ help semantic preservation within limited bit lengths. Model 2* demonstrates significantly better performance across most metrics compared to Models 1 and 5. The superior performance is attributed to the longer global hash code used by Model 2*, enabling enhanced semantic information storage. Although Model 5 benefits from reranking, it only optimizes the top portion of a sorted list, not the entire list, limiting its performance improvement. The comparison is illustrative, considering the inherent differences. Interestingly, Model 2*'s dominance diminishes when hash code lengths of Models 1 and 5 are increased to 2048 bits on the MSRVTT dataset, indicating diminishing returns in bit length extension and the necessity for more intelligent learning objectives. (viii) Introducing an optimized rotation matrix can improve fine-grained quantization. In contrast to 5, Model 9 without $\boldsymbol{P}_n$ in each local quantization module shows slight but observable performance decays, indicating that optimizing subspace partition for learnable quantization can lead to better quantization codes.

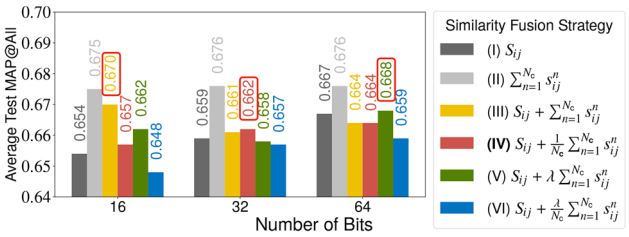

Fig. 10 Average MAP@All of text-to-image and image-to-text retrieval *w.r.t.* various bit-lengths and fusion strategies. S_{ij} denotes the global similarity and s_{ij}^n denotes the nth local similarity. Under each bit-length, the result of the best fusion strategy except (I) and (II) is highlighted in a red box. Model 10 in table 5 adopts (IV) by default

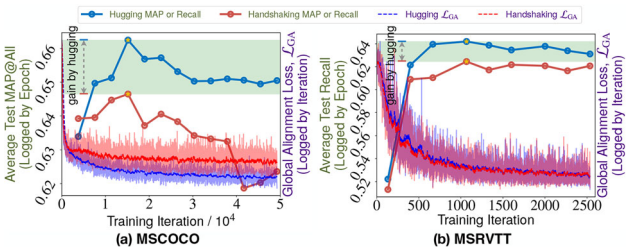

Fig. 11 Dynamics of the performance on the test set and global hash alignment loss on the train set. The recall on MSRVTT is computed by geometric mean of R@{1,5,10}. We compare two strategies with HUGGINGHASH

4.3.2 Understanding the Effectiveness of Fine-grained Alignment

Until now, we have learned that hugging can effectively enhance global hash code learning and boost retrieval performance, but how fine-grained alignment helps to learn better global representation still remains unclear. In this section, we delve into this phenomenon and aim to understand the underlying mechanism more comprehensively.

Without loss of generality, we take HUGGINGHASH as the analysis object in this section. First, we investigate the training dynamics to acquire the differences between hugging and handshaking during training. Then, from the macro view, we visualize fine-grained latent space to show what each cluster looks like. Finally, from the micro view, we analyze visual attention maps to facilitate an intuitive understanding of how local clusters contribute to global hash learning.

Analysis of training dynamics We study the dynamics of training loss and testing performance of HUGGINGHASH and a handshaking baseline without fine-grained alignment. The results on MSCOCO and MSRVTT are illustrated in Fig. 11. On MSCOCO, we can observe that fine-grained alignment helps to reduce the modality gap of global representation. Hence hugging reaches a better optimum than handshaking. Although both hugging and handshaking tend to overfit and degrade after reaching the optimum, hugging can still retain better performance than handshaking. On MSRVTT, the pre-

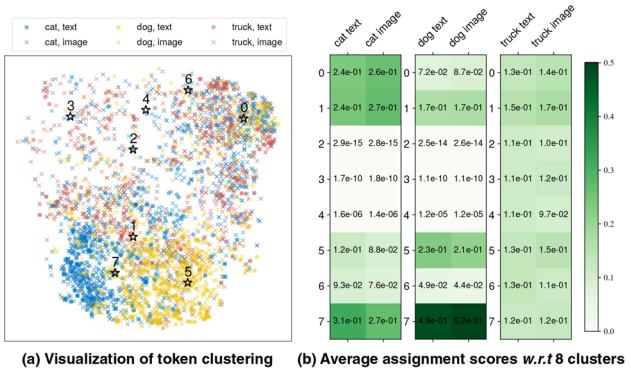

Fig. 12 Visualization of fine-grained latent space and fine-grained cross-modal correspondence in hugging

trained CLIP provides an easy start for aligning the texts and videos. Thus, we can see hugging and handshaking have similar training loss dynamics. Nevertheless, handshaking still underperforms hugging on the test set, which suggests the significance of fine-grained alignment to model generalizability.

Basically, our intuition is that global alignment provides direct guidance to the cross-modal correspondence, while fine-grained alignment constructs a structural space with (latent) concept-level correspondence to regularize cross-modal learning. The synergy of global and fine-grained turns out to reach a better optimum.

Analysis of fine-grained latent space As a case study, we pick up some text-image pairs associated with 3 single labels: cat, dog and truck, from the MSCOCO dataset. Each text or image is represented by multiple content tokens in the fine-grained latent space. Figure 12a visualizes the token distribution by different labels and modalities, where the black stars mark the centroids of clusters. For quantitative analysis, we compute the class-specific and modality-specific average assignment scores *w.r.t.* all clusters and present them in Fig. 12b. The general consistency between the text and the image assignments *w.r.t.* each class implies the effectiveness of cross-modal fine-grained alignment. Besides, we observe that the assignment pattern can vary among different labels. While truck exhibits a uniform pattern, cat and dog concentrate on clusters 1, 5 and 7. In particular, we hypothesize a connection between the cluster 7 and the concept of *animals*.

Analysis of visual attention Apart from Fig. 12, which provides a macro view of the cross-modal alignment in the fine-grained latent space, here we present a more intuitive visualization of the text-visual semantic correspondence and how it contributes to the global representations.

We use GradCAM (Selvaraju et al., 2017) to visualize the attention map *w.r.t.* the `[CLS]` token, and the attention maps *w.r.t.* different clusters in hugging. For cluster-level attention, we aggregate the attention maps of the associated content tokens weighted by their assignment scores. The

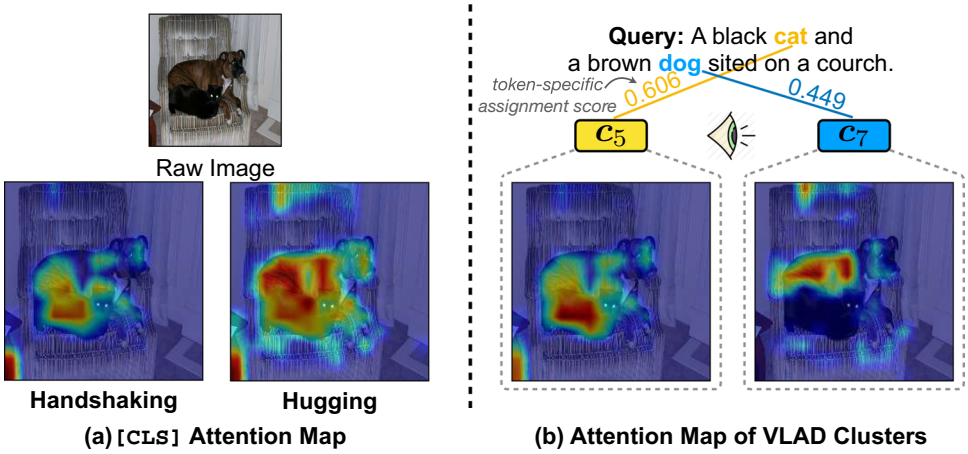

Fig. 13 Attention map visualization on MSCOCO. Left: attention maps *w.r.t.* [CLS] in handshaking and hugging. Hugging helps to focus on more comprehensive semantic regions. Right: attention maps *w.r.t.* different clusters. We can see the text-visual semantic correspondence

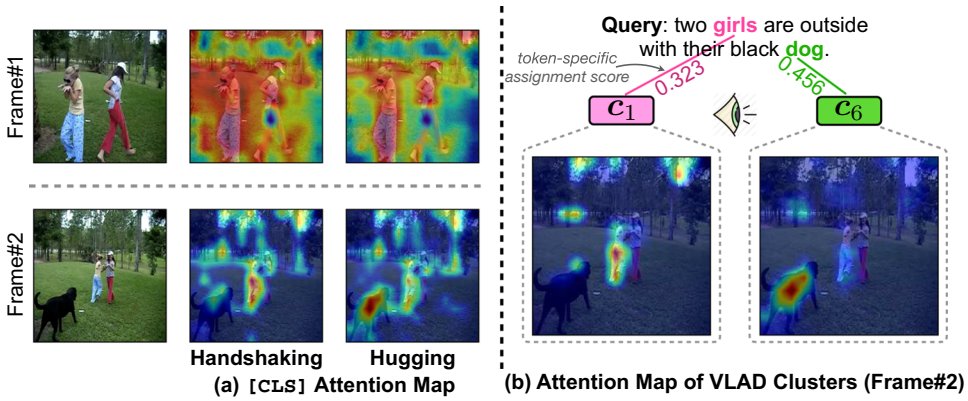

Fig. 14 Attention map visualization on MSRVTT. Left: attention maps on different frames *w.r.t.* the temporal [CLS] token in handshaking and hugging. Hugging helps to focus on more precise semantic regions (e.g. the people) in frame#1 and more comprehensive semantic regions in frame#2 (e.g. the dog). Right: attention maps *w.r.t.* different VLAD clusters. We can see some correspondence between text and visual semantics

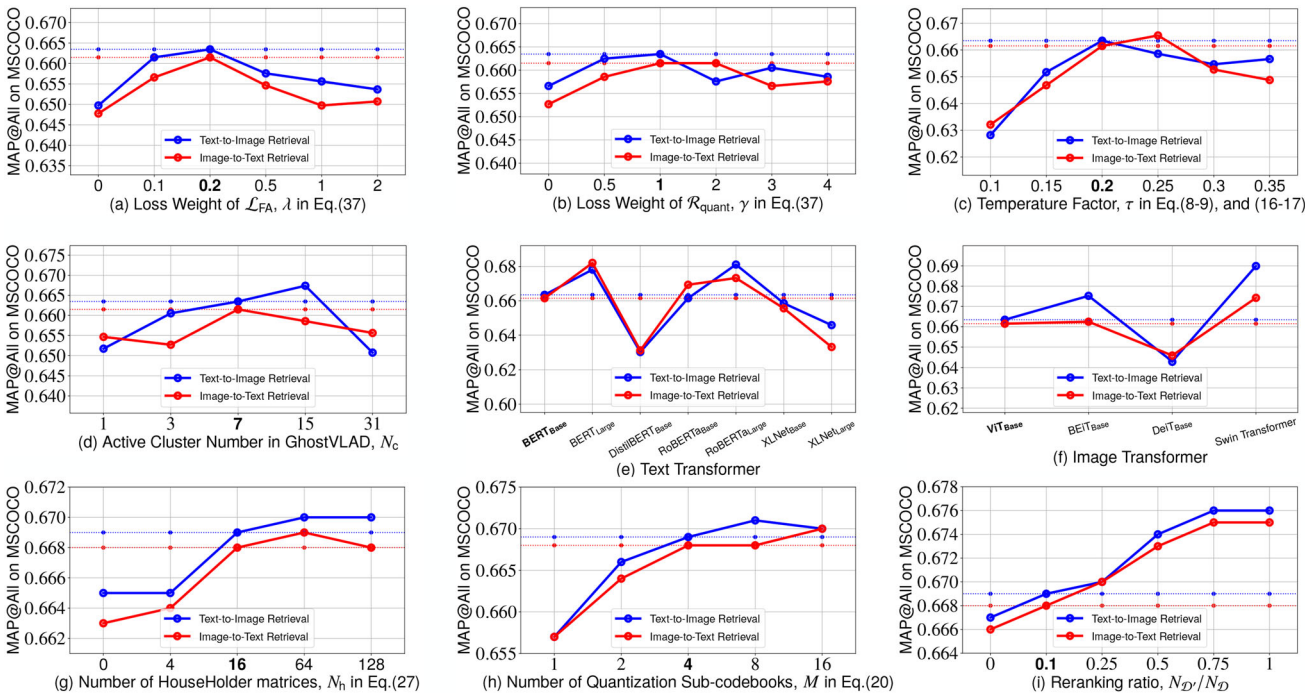

Fig. 15 Hyper-parameter Sensitivities on MSCOCO dataset. Without loss of generality, the results in (**a**)–(**f**) are obtained with a 64-bit HUGGINGHASH. The quantization- and reranking-related results in (**g**)–(**i**) are obtained with a 64-bit HUGGINGHASH$^+$. Default settings are marked in bold. The dotted lines mark the MAP results under default settings

results on MSCOCO and MSRVTT datasets are illustrated in Figs. 13 and 14, respectively. We can see fine-grained alignment improves global hash representations in two regards. (i) Fine-grained alignment is conceptual-aware and helps to capture more complete cues. As shown in Fig. 13b, clusters $\boldsymbol{c}_5$ and $\boldsymbol{c}_7$ are relevant to cat and dog respectively, and their attention maps also present a concentration on corresponding areas in the image. Therefore, in Fig. 13a, the [CLS] attention map of hugging demonstrates a comprehensive grasp of the images. Whereas the [CLS] attention map of handshaking seems to miss the attention to the dog, leading to inferior hash codes. A similar phenomenon can be learned from Fig. 14, where handshaking fails to focus on the dog in frame#2. (ii) Fine-grained alignment regularizes global learning and helps to focus on details more precisely. As shown in Fig. 14a, while handshaking spreads the attention to larger and vaguer areas, hugging attends to the people in frame#1 more accurately so that we can see the outlines. As a result, hugging helps to produce more discriminative hash codes.

4.3.3 Hyper-Parameter Analysis

In this section, we analyze how hyper-parameters influence HUGGINGHASH and HUGGINGHASH$^+$. We conduct detailed experiments on MSCOCO since it is a standard benchmark dataset for vision-language tasks. Without loss of generality, we analyze most hyper-parameters with HUGGINGHASH and analyze the quantization-related and reranking-related hyper-parameters with HUGGINGHASH$^+$. Results are illustrated in Fig. 15.

Effects of loss terms λ is the weight of $\mathcal{L}_{\text{FA}}$ (i.e., fine-grained alignment loss) that essentially controls its task gradient contribution to the learning process. Adjusting λ from 0 to 0.2 can boost the performance, verifying that $\mathcal{L}_{\text{FA}}$ is beneficial. However, we can see the gain drops as λ increases beyond 0.2. This is because the auxiliary task of fine-grained alignment dominates the learning and even adversely constricts the main task of aligning hash codes. γ controls the strength of regularization $\mathcal{R}_{\text{quant}}$ and a proper range is [0.5, 1].

Effects of τ The temperature factor, τ controls the penalty intensity of contrastive learning, which is sensitive. Figure 15c illustrates its effect. We can learn that a suitable range for τ is [0.2, 0.25].

Effects of cluster number Figure 15d shows the effect of active cluster (i.e., latent concept) number N_c in GhostVLAD. While 15 and 7 are reasonable choices for N_c, we set $N_c = 7$ by default to pursue a higher training efficiency.

Effects of used transformers We equip HUGGINGHASH with different transformers (Sanh et al., 2019; Liu et al., 2019c; Yang et al., 2019; Touvron et al., 2021; Liu et al., 2021b; Bao et al., 2022). Figure 15e and f show that large BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019c) variants are good choices for the text transformer. Swin Transformer (Liu et al., 2021b) is a good choice for image encoder.

Effects of optimized quantization Note that the quantization codes are taken to rerank top-10% hash-ranked items by default, but we report the MAP metric *w.r.t.* the whole database. Hence, considering that the influence of different settings in the quantization is smoothed, we focus more on the trends rather than their magnitudes. N_{h} denotes the number of Householder matrices in Eq. (27) set to approximate the optimal rotation matrix. The effect of N_{h} is shown in Fig. 15g, where $N_{\text{h}} = 0$ means not enabling rotation matrix such that Eq. (25) is simplified to Eq. (20). We can see that $N_{\text{h}} \geq 16$ yields converged results. We take $N_{\text{h}} = 16$ by default to slightly reduce computation while maintaining satisfactory performance. The quantization sub-codebook number in each local head, M, controls the distortion and also the bit-length of quantization codes. Figure 15h illustrates the sensitivity of M. The retrieval performance continues to improve as it increases until 8. We set $M = 4$ by default for efficiency concerns.

Effects of reranking range As illustrated in Fig. 5, we narrow the reranking range to top-$N_{\mathcal{D}'}$ items by the hash-based ranking. We investigate the effect of rerank proportion, $N_{\mathcal{D}'}/N_{\mathcal{D}}$ in Fig. 15i, where "0" indicates only global hash-based ranking and "1" means (re)ranking with fine-grained quantization codes on the whole database. We set reranking top-10% of items by default. The performance keeps improving as we increase $N_{\mathcal{D}'}/N_{\mathcal{D}}$ from 0% $\sim$ 75% and the trend is prone to converge after $N_{\mathcal{D}'}/N_{\mathcal{D}} > 75\%$.

4.3.4 Retrieval Visualization Analysis

We visualize the top-10 retrieval results on two datasets: Flickr25K for image-text retrieval and MSRVTT for video-text retrieval. We compare three different methods: **Handshaking**, **Hugging$^+$ w/o Reranking**, and **Hugging$^+$ w/ Reranking**, as illustrated in Figs. 16 and 17.

In Fig. 16, we observe that **Hugging$^+$** yields a more accurate result set compared to **Handshaking**. Furthermore, with the use of fine-grained reranking, the images most relevant to the query, specifically those involving landscapes, are ranked higher. At the same time, a museum image that initially appeared at Rank 3, which was deemed a negative sample because it failed to match the label despite containing a 'lamb', was demoted, thereby improving AP@10[3].

[3] The definition of Average Precision (AP) in information retrieval:

$$AP@K = \frac{1}{|Rel(K)|} \sum_{k=1}^{K} P(k) \cdot r(k), \tag{45}$$

where K is the number of items considered for evaluation. $|Rel(K)|$ is the total amount of relevant items. $P(k)$ denotes the precision at the kth

Table 6 Retrieval performance of UCMH approaches on MSCOCO and MSRVTT datasets

Method		Dataset (Metrics) →			MSCOCO (MAP@All)						MSRVTT (Geometric Mean of R@{1,5,10})							
					Text-to-Image Retrieval			Image-to-Text Retrieval			Text-to-Video Retrieval				Video-to-Text Retrieval			
	Year	Use Trf?	Hug?	ReR?	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	256 bits	512 bits	1K bits	2K bits	256 bits	512 bits	1K bits	2K bits
DJSRH	2019	✗	✗	✗	0.573	0.578	0.584	0.572	0.575	0.579	–	–	–	–	–	–	–	–
		✓	✗	✗	0.623	0.621	0.627	0.619	0.624	0.627	0.149	0.180	0.259	0.332	0.141	0.170	0.262	0.295
		✓	✓	✗	0.637	0.626	0.628	0.630	0.634	0.637	0.337	0.371	0.451	0.527	0.330	0.364	0.459	0.492
		✓	✓	✓	0.643	0.636	0.636	0.641	0.643	0.644	0.343	0.504	0.575	0.588	0.354	0.510	0.578	0.594
DSAH	2020	✗	✗	✗	0.606	0.599	0.620	0.598	0.589	0.609	–	–	–	–	–	–	–	–
		✓	✗	✗	0.641	0.636	0.642	0.637	0.639	0.648	0.161	0.196	0.281	0.359	0.156	0.187	0.290	0.321
		✓	✓	✗	0.656	0.659	0.665	0.669	0.663	0.670	0.336	0.374	0.465	0.547	0.336	0.367	0.475	0.507
		✓	✓	✓	0.660	0.666	0.671	0.674	0.670	0.678	0.344	0.507	0.577	0.591	0.348	0.507	0.575	0.592
DGCPN	2021a	✗	✗	✗	0.594	0.603	0.616	0.587	0.594	0.612	–	–	–	–	–	–	–	–
		✓	✗	✗	0.623	0.621	0.627	0.633	0.641	0.643	0.128	0.177	0.284	0.371	0.120	0.198	0.278	0.316
		✓	✓	✗	0.645	0.662	0.659	0.644	0.650	0.658	0.301	0.352	0.463	0.550	0.295	0.378	0.460	0.502
		✓	✓	✓	0.661	0.676	0.669	0.650	0.661	0.664	0.340	0.500	0.576	0.591	0.352	0.511	0.582	0.596
CIRH	2023	✗	✗	✗	0.587	0.592	0.606	0.596	0.603	0.610	–	–	–	–	–	–	–	–
		✓	✗	✗	0.632	0.636	0.645	0.629	0.633	0.641	0.129	0.179	0.288	0.373	0.124	0.204	0.286	0.325
		✓	✓	✗	0.648	0.661	0.669	0.647	0.653	0.657	0.307	0.358	0.467	0.558	0.303	0.385	0.471	0.515
		✓	✓	✓	0.655	0.670	0.681	0.657	0.659	0.668	0.338	0.499	0.572	0.586	0.346	0.500	0.566	0.579
UCHSTM	2023	✗	✗	✗	0.595	0.601	0.611	0.589	0.600	0.615	–	–	–	–	–	–	–	–
		✓	✗	✗	0.636	0.642	0.648	0.631	0.639	0.648	0.121	0.168	0.271	0.351	0.114	0.187	0.264	0.302
		✓	✓	✗	0.651	0.665	0.676	0.648	0.654	0.673	0.308	0.356	0.464	0.548	0.305	0.378	0.460	0.503
		✓	✓	✓	0.659	0.674	0.685	0.660	0.661	0.677	0.339	0.501	0.577	0.590	0.345	0.502	0.574	0.588
HUGGINGHASH or HUGGINGHASH^{+}	Ours	✓	✗	✗	0.633	0.644	0.650	0.631	0.636	0.644	0.333	0.507	0.598	0.628	0.328	0.501	0.591	0.621
		✓	✓	✗	0.652	0.661	0.663	0.646	0.653	0.662	0.357	0.527	0.616	0.641	0.358	0.531	0.615	0.643
		✓	✓	✓	0.658	0.664	0.669	0.655	0.660	0.668	0.368	0.537	0.619	0.642	0.375	0.540	0.618	0.643

In particular, by modifying 5 state-of-the-art baselines, we examine the compatibility and efficacy of our proposed designs with transformer-based UCMH . 'Use Trf?' means whether to use transformers. 'ReR?' indicates whether to enable a reranking stage after ranking by hash codes. 'Hug?' means whether the model is trained by hugging (w/o reranking) or hugging^{+} (w/ reranking)

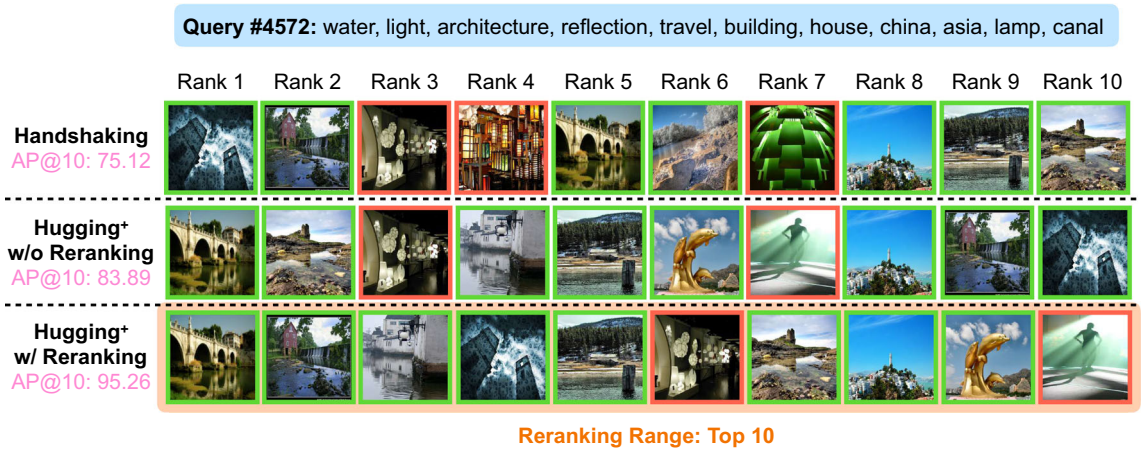

Fig. 16 Top-10 ranked images using different strategies on the Flickr25K dataset. For **Hugging**$^+$ **w/ Reranking**, we further rerank the top 10 ranked images. The ground-truth labels of query #4572 are 'sky', 'structure', and 'water'. The green and red bounding boxes for images mark positive and negative images, respectively. The evaluation metric is the average precision at 10, i.e., AP@10 (Color figure online)

In Fig. 17, two cases are presented. For query #9340, the **Hugging**$^+$ model successfully highlights the "trying to comfort" detail within the query by training fine-grained alignment. This pushes the videos with matching scenarios to the forefront. Reranking further exploits this fine-grained information, effectively improving the rank of videos closely aligned with the query. For query #8914, although not exemplary in terms of evaluation metrics, it serves to demonstrate the generalizability of **Hugging**$^+$ and fine-grained reranking. Despite the absence of ground-truth videos at higher ranks, the reranked list still features videos somewhat relevant to the query text. Specifically, reranking elevates videos featuring 'models' and 'runways', which might be considered false negatives due to limited annotations in the evaluation dataset.

4.3.5 Improving Existing Approaches with Hugging and Hugging$^+$

In Fig. 7, we have illustrated how transformers, hugging and hugging$^+$ help to improve the in-domain performance and cross-domain generalizability, respectively. In this section, we conduct more detailed experiments on both text-image (i.e., MSCOCO) and text-video (i.e., MSRVTT) datasets to investigate the compatibility and effectiveness of our proposed designs with existing UCMH approaches. We integrate the transformers, hugging and hugging$^+$, with each of the 5 state-of-the-art UCMH approaches, namely DJSRH, DSAH, DGCPN, CIRH, and UCHSTM.

We report the results in Table 6, from which we see consistent improvements in the three designs. To be specific, on MSCOCO, when equipped with the same designs, several existing approaches show competitive or even superior performance to HuggingHash (or HuggingHash$^+$), e.g. UCHSTM + hugging$^+$ *vs.* HuggingHash$^+$. It is acceptable because HuggingHash and HuggingHash$^+$ are two simple instantiation models. Using more delicate global alignment mechanisms to improve the instantiation is reasonable.

Quite differently, on MSRVTT, the baselines still fall behind HuggingHash (or HuggingHash$^+$) by considerable gaps even with the same proposed designs. The phenomenon is partially due to the unsatisfied basic assumption, as we have discussed in Sect. 4.2.2. Despite the performance gaps, we can learn that hugging and hugging$^+$ significantly enhance retrieval results. By introducing fine-grained alignment based on contrastive learning, hugging brings large performance gains to the handshaking (i.e., only using transformers) models, which confirms the positive synergy between global and fine-grained alignment in training. Besides, hugging$^+$ with reranking further improves the recall beyond hugging, while the scale of the improvements varies with different bit-lengths. Under shorter bit-length, e.g. 256 bits, the gain of reranking is relatively mild. The reason is that global hash-based ranking misses many positive items in the top-10% subset, and hence reranking is unable to help. Under medium bit-length, e.g. 512 or 1024 bits, the global hash-based ranking is capable of including more positive items in the top-10% subset but still insufficient to rank them precisely. Fortunately, the fine-grained reranking compensates for this shortcoming and we observe large relative improvements of about 20% to 40%. Under longer bit-length, e.g. 2048 bits, the gain begins to shrink because the hash-based ranking has become more accurate.

position. $r(k)$ is the relevance of the kth ranked item (0: irrelevant; 1: relevant).

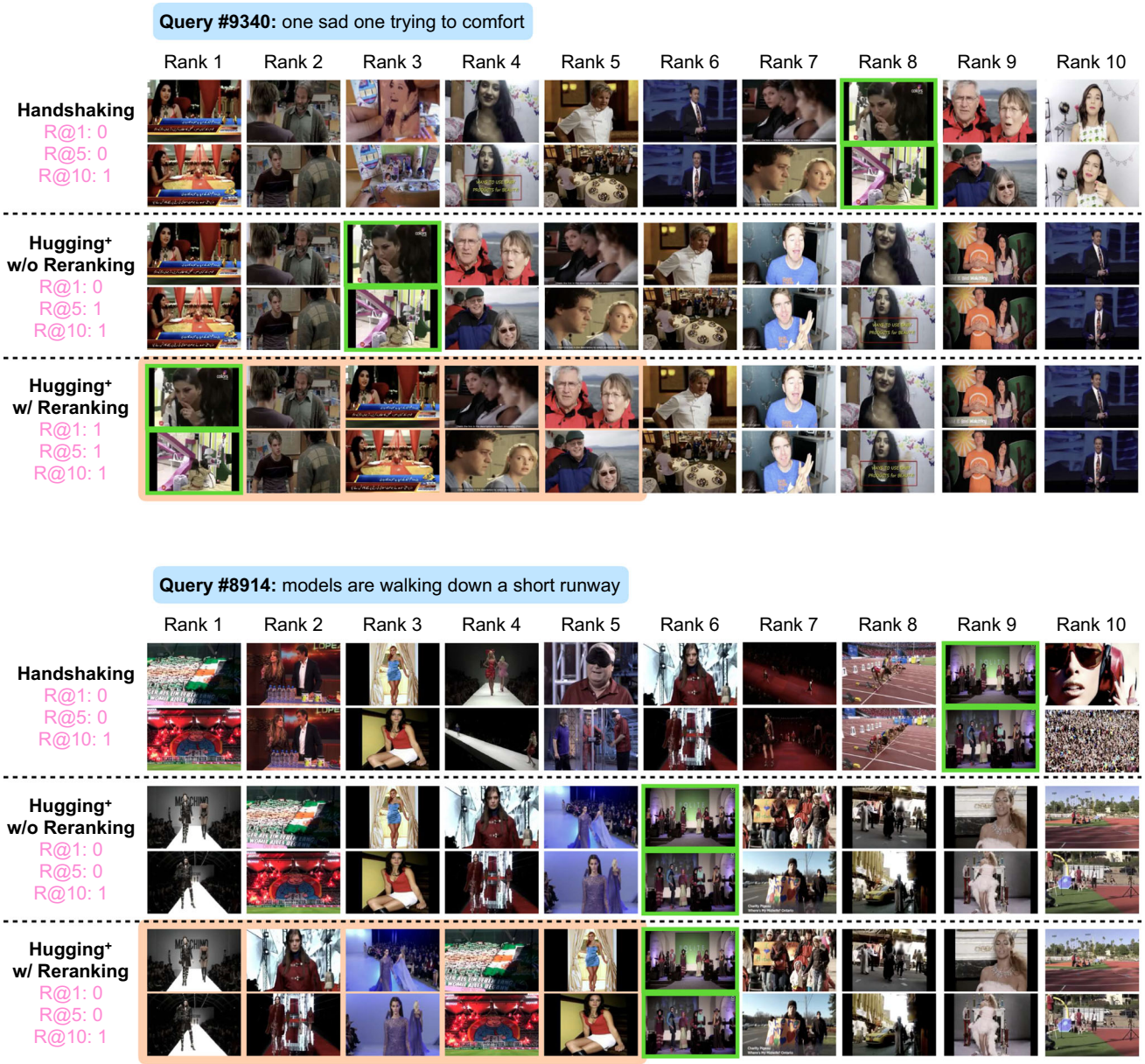

Fig. 17 Top-10 ranked videos using different strategies on the MSRVTT dataset. For **Hugging$^+$ w/ Reranking**, we further rerank the top 5 ranked videos. We show two search cases. The green bounding box marks the ground-truth video *w.r.t.* the text query. The evaluation metrics are the recalls at 1, 5, and 10, i.e., R@{1,5,10} (Color figure online)

5 Conclusions

This paper studies the new and practical problem of transformer-based unsupervised cross-modal hashing (UCMH). We propose a *hugging* framework that unifies multi-granularity cross-modal alignment as solid self-supervision for hash learning. Besides, we extend hugging to *hugging*$^+$ that effectively learns optimized quantized representations and aligns fine-grained cross-modal correspondence simultaneously. It retains the benefit of improving global hash codes and also provides fine-grained quantization code as a gift. Reranking with fine-grained quantization codes effectively boosts retrieval performance while enjoying high efficiency. We build HUGGINGHASH and HUGGINGHASH$^+$ to instantiate hugging and hugging$^+$, respectively, and show their advantages on text-image and text-video retrieval. We conduct extensive experiments to investigate the efficacy of different components in our design. We also assemble the proposed hugging and hugging$^+$ to several state-of-the-art UCMH

approaches, showing that our design is flexible and compatible with UCMH when choosing transformers as backbones.

Acknowledgements This work is supported in part by the National Natural Science Foundation of China under grant 62171248, 62301189, Guangdong Basic and Applied Basic Research Foundation under grant 2021A1515110066, Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies (2022B1212010005), the PCNL KEY project (PCL2021A07), and Shenzhen Science and Technology Program under Grant JCYJ20220818101012025, RCBS20221008093124061, GXWD20220811172936001.

References

An, X., Deng, J., Yang, K., Li, J., Feng, Z., Guo, J., Yang, J., & Liu, T. (2023). Unicom: Universal and compact representation learning for image retrieval. In *ICLR*. OpenReview.net.

Arandjelovic, R., Gronát, P., Torii, A., Pajdla, T., & Sivic, J. (2016). Netvlad: CNN architecture for weakly supervised place recognition. In *CVPR* (pp. 5297–5307). IEEE Computer Society.

Asadi, N., & Lin, J. (2013). Effectiveness/efficiency tradeoffs for candidate generation in multi-stage retrieval architectures. In *SIGIR* (pp. 997–1000). ACM.

Babenko, A., & Lempitsky, V. S. (2014). Additive quantization for extreme vector compression. In *CVPR* (pp. 931–938). IEEE Computer Society.

Bain, M., Nagrani, A., Varol, G., & Zisserman, A. (2021). Frozen in time: A joint video and image encoder for end-to-end retrieval. In *ICCV*, (pp. 1708–1718). IEEE.

Bao, H., Wang, W., Dong, L., Liu, Q., Mohammed, O. K., Aggarwal, K., Som, S., Piao, S., & Wei, F. (2022b). Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. In *NeurIPS*.

Bao, H., Dong, L., Piao, S., & Wei, F. (2022). Beit: BERT pre-training of image transformers. In *ICLR*. OpenReview.net.

Bengio, Y., Léonard, N., & Courville, A. C. (2013). Estimating or propagating gradients through stochastic neurons for conditional computation. CoRR abs/1308.3432.

Cao, Y., Liu, B., Long, M., & Wang, J. (2018). Cross-modal hamming hashing. In *ECCV*, volume 11205 (pp. 207–223). Springer.

Cao, Y., Long, M., Wang, J., & Liu, S. (2017). Deep visual-semantic quantization for efficient image retrieval. In *CVPR* (pp. 916–925). IEEE Computer Society.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. In *ECCV*, volume 12346 (pp. 213–229). Springer.

Chen, D. L., & Dolan, W. B. (2011). Collecting highly parallel data for paraphrase evaluation. In *ACL* (pp. 190–200). The Association for Computer Linguistics.

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. E. (2020a). A simple framework for contrastive learning of visual representations. In *ICML*, volume 119 (pp. 1597–1607). PMLR.

Chen, T., Li, L., & Sun, Y. (2020b). Differentiable product quantization for end-to-end embedding compression. In *ICML*, volume 119 (pp. 1617–1626). PMLR.

Chen, Y., Wang, S., Lu, J., Chen, Z., Zhang, Z., & Huang, Z. (2021). Local graph convolutional networks for cross-modal hashing. In *ACM Multimedia* (pp. 1921–1928). ACM.

Chen, Z., Yu, W., Li, C., Nie, L., & Xu, X. (2018). Dual deep neural networks cross-modal hashing. In *AAAI* (pp. 274–281). AAAI.

Chen, Y., Zhang, S., Liu, F., Chang, Z., Ye, M., & Qi, Z. (2022). Transhash: Transformer-based hamming hashing for efficient image retrieval. In *ICMR* (pp. 127–136). ACM.

Chua, T., Tang, J., Hong, R., Li, H., Luo, Z., & Zheng, Y. (2009). NUS-WIDE: A real-world web image database from national university of singapore. In *CIVR*. ACM.

Cui, H., Zhu, L., Li, J., Cheng, Z., & Zhang, Z. (2021). Two-pronged strategy: Lightweight augmented graph network hashing for scalable image retrieval. In *ACM Multimedia* (pp. 1432–1440). ACM.

Datar, M., Immorlica, N., Indyk, P., & Mirrokni, V. S. (2004). Locality-sensitive hashing scheme based on p-stable distributions. In *SCG* (pp. 253–262). ACM.

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL* (pp. 4171–4186). Association for Computational Linguistics.

Ding, G., Guo, Y., Zhou, J., & Gao, Y. (2016). Large-scale cross-modality search via collective matrix factorization hashing. *IEEE Transactions on Image Processing, 25*(11), 5427–5440.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net.

Dubey, S. R., Singh, S. K., & Chu, W. (2022). Vision transformer hashing for image retrieval. In *ICME* (pp. 1–6). IEEE.

Ester, M., Kriegel, H., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD* (pp. 226–231). AAAI.

Fang, B., Wu, W., Liu, C., Zhou, Y., Song, Y., Wang, W., Shu, X., Ji, X., & Wang, J. (2023). UATVR: Uncertainty-adaptive text-video retrieval. In *ICCV* (pp. 13677–13687). IEEE.

Gabeur, V., Sun, C., Alahari, K., & Schmid, C. (2020). Multi-modal transformer for video retrieval. In *ECCV*, volume 12349 (pp. 214–229). Springer.

Gao, D., Jin, L., Chen, B., Qiu, M., Li, P., Wei, Y., Hu, Y., & Wang, H. (2020). Fashionbert: Text and image matching with adaptive loss for cross-modal retrieval. In *SIGIR* (pp. 2251–2260). ACM.

Ge, T., He, K., Ke, Q., & Sun, J. (2013). Optimized product quantization for approximate nearest neighbor search. In *CVPR* (pp. 2946–2953). IEEE Computer Society.

Gong, Y., Lazebnik, S., Gordo, A., & Perronnin, F. (2013). Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35*(12), 2916–2929.

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. B. (2020). Momentum contrast for unsupervised visual representation learning. In *CVPR* (pp. 9726–9735). Computer Vision Foundation/IEEE.

He, X., Pan, Y., Tang, M., & Lv, Y. (2021). Self-supervised video retrieval transformer network. *CoRR*, abs/2104.07993.

Heo, J., Lee, Y., He, J., Chang, S., & Yoon, S. (2012). Spherical hashing. In *CVPR* (pp. 2957–2964). IEEE Computer Society.

Hoang, T., Do, T., Nguyen, T. V., & Cheung, N. (2020). Unsupervised deep cross-modality spectral hashing. *IEEE Transactions on Image Processing, 29*, 8391–8406.

Hoang, T., Do, T., Nguyen, T. V., & Cheung, N. (2023). Multimodal mutual information maximization: A novel approach for unsupervised deep cross-modal hashing. *IEEE Trans. Neural Networks Learn. Syst., 34*(9), 6289–6302.

Hu, H., Xie, L., Hong, R., & Tian, Q. (2020). Creating something from nothing: Unsupervised knowledge distillation for cross-modal hashing. In *CVPR*, (pp. 3120–3129). Computer Vision Foundation/IEEE.

Huiskes, M. J., & Lew, M. S. (2008). The MIR flickr retrieval evaluation. In *Multimedia Information Retrieval* (pp. 39–43). ACM.

Humenberger, M., Cabon, Y., Pion, N., Weinzaepfel, P., Lee, D., Guérin, N., Sattler, T., & Csurka, G. (2022). Investigating the role of image retrieval for visual localization: An exhaustive benchmark. *International Journal of Computer Vision, 130*(7), 1811–1836.

Hu, D., Nie, F., & Li, X. (2019). Deep binary reconstruction for cross-modal hashing. *IEEE Transactions on Multimedia, 21*(4), 973–985.

Hu, M., Yang, Y., Shen, F., Xie, N., Hong, R., & Shen, H. T. (2019). Collective reconstructive embeddings for cross-modal hashing. *IEEE Transactions on Image Processing, 28*(6), 2770–2784.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37 (pp. 448–456). JMLR.org.

Irie, G., Arai, H., & Taniguchi, Y. (2015). Alternating co-quantization for cross-modal hashing. In *ICCV* (pp. 1886–1894). IEEE Computer Society.

Jang, Y. K., Cho, N. I. (2021). Self-supervised product quantization for deep unsupervised image retrieval. In *ICCV* (pp. 12065–12074). IEEE.

Jégou, H., Douze, M., & Schmid, C. (2011). Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 33*(1), 117–128.

Jiang, Q., Li, W. (2017). Deep cross-modal hashing. In *CVPR* (pp. 3270–3278). IEEE Computer Society.

Jin, P., Huang, J., Xiong, P., Tian, S., Liu, C., Ji, X., Yuan, L., & Chen, J. (2023a). Video-text as game players: Hierarchical banzhaf interaction for cross-modal representation learning. In *CVPR* (pp. 2472–2482). IEEE.

Jin, P., Li, H., Cheng, Z., Li, K., Ji, X., Liu, C., Yuan, L., & Chen, J. (2023b). Diffusionret: Generative text-video retrieval with diffusion model. In *ICCV* (pp. 2470–2481). IEEE.

Johnson, J., Douze, M., & Jégou, H. (2021). Billion-scale similarity search with gpus. *IEEE Trans. Big Data, 7*(3), 535–547.

Kalantidis, Y., & Avrithis, Y. (2014). Locally optimized product quantization for approximate nearest neighbor search. In *CVPR* (pp. 2329–2336). IEEE Computer Society.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *ICLR*.

Klein, B. E., Wolf, L. (2019). End-to-end supervised product quantization for image search and retrieval. In *CVPR* (pp. 5041–5050). Computer Vision Foundation / IEEE.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NeurIPS*, (pp. 1106–1114).

Kumar, S., & Udupa, R. (2011). Learning hash functions for cross-view similarity search. In *IJCAI* (pp. 1360–1365). IJCAI/AAAI.

Le, Q. V., Mikolov, T. (2014). Distributed representations of sentences and documents. In *ICML*, volume 32 (pp. 1188–1196). JMLR.org.

Li, M., & Wang, H. (2021). Unsupervised deep cross-modal hashing by knowledge distillation for large-scale cross-modal retrieval. In *ICMR* (pp. 183–191). ACM.

Li, C., Deng, C., Wang, L., Xie, D., & Liu, X. (2019). Coupled cyclegan: Unsupervised hashing network for cross-modal retrieval. In *AAAI* (pp. 176–183). AAAI.

Li, G., Duan, N., Fang, Y., Gong, M., & Jiang, D. (2020a). Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *AAAI* (pp. 11336–11344). AAAI.

Li, S., Li, X., Lu, J., & Zhou, J. (2021b). Self-supervised video hashing via bidirectional transformers. In *CVPR* (pp. 13549–13558). Computer Vision Foundation / IEEE.

Li, P., Xie, H., Ge, J., Zhang, L., Min, S., & Zhang, Y. (2022a). Dual-stream knowledge-preserving hashing for unsupervised video retrieval. In *ECCV*, volume 13674 (pp. 181–197). Springer.

Lin, T., Maire, M., Belongie, S. J., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *ECCV*, volume 8693 (pp. 740–755). Springer.

Lin, X., Tiwari, S., Huang, S., Li, M., Shou, M. Z., Ji, H., & Chang, S. (2023). Towards fast adaptation of pretrained contrastive models for multi-channel video-language retrieval. In *CVPR* (pp. 14846–14855). IEEE.

Liong, V. E., Lu, J., Wang, G., Moulin, P., & Zhou, J. (2015). Deep hashing for compact binary codes learning. In *CVPR* (pp. 2475–2483). IEEE Computer Society.

Li, Q., Sun, Z., He, R., & Tan, T. (2020). A general framework for deep supervised discrete hashing. *International Journal of Computer Vision, 128*(8), 2204–2222.

Liu, Y., Albanie, S., Nagrani, A., and Zisserman, A. (2019b). Use what you have: Video retrieval using representations from collaborative experts. In *BMVC*, (p. 279). BMVA.

Liu, S., Fan, H., Qian, S., Chen, Y., Ding, W., and Wang, Z. (2021a). Hit: Hierarchical transformer with momentum contrast for video-text retrieval. In *ICCV* (pp. 11895–11905). IEEE.

Liu, H., Ji, R., Wu, Y., Huang, F., & Zhang, B. (2017). Cross-modality binary code learning via fusion similarity hashing. In *CVPR* (pp. 6345–6353). IEEE Computer Society.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021b). Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV* (pp. 9992–10002). IEEE.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019c). Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Liu, S., Qian, S., Guan, Y., Zhan, J., & Ying, L. (2020). Joint-modal distribution-based similarity hashing for large-scale unsupervised deep cross-modal retrieval. In *SIGIR* (pp. 1379–1388). ACM.

Liu, T. (2009). Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval, 3*(3), 225–331.

Liu, H., Wang, R., Shan, S., & Chen, X. (2019). Deep supervised hashing for fast image retrieval. *International Journal of Computer Vision, 127*(9), 1217–1234.

Liu, Z., Xiong, C., Lv, Y., Liu, Z., & Yu, G. (2023). Universal vision-language dense retrieval: Learning A unified representation space for multi-modal retrieval. In *ICLR*. OpenReview.net.

Li, F., Wang, T., Zhu, L., Zhang, Z., & Wang, X. (2021). Task-adaptive asymmetric deep cross-modal hashing. *Knowl. Based Syst., 219*, 106851.

Li, T., Zhang, Z., Pei, L., & Gan, Y. (2022). Hashformer: Vision transformer based deep hashing for image retrieval. *IEEE Signal Processing Letters, 29*, 827–831.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision, 60*(2), 91–110.

Lu, J., Batra, D., Parikh, D., & Lee, S. (2019). Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS* (pp. 13–23).

Lu, D., Wang, J., Zeng, Z., Chen, B., Wu, S., & Xia, S. (2021). Swinfghash: Fine-grained image retrieval via transformer-based hashing network. In *BMVC* (p. 432). BMVA.

Luo, H., Ji, L., Zhong, M., Chen, Y., Lei, W., Duan, N., & Li, T. (2022). Clip4clip: An empirical study of CLIP for end to end video clip retrieval and captioning. *Neurocomputing, 508*, 293–304.

Martinez, J., Clement, J., Hoos, H. H., & Little, J. J. (2016). Revisiting additive quantization. In *ECCV*, volume 9906 (pp. 137–153). Springer.

Messina, N., Amato, G., Esuli, A., Falchi, F., Gennaro, C., & Marchand-Maillet, S. (2021). Fine-grained visual textual alignment for cross-modal retrieval using transformer encoders. *ACM Trans. Multim. Comput. Commun. Appl.*, 17(4):128:1–128:23.

Mikriukov, G., Ravanbakhsh, M., & Demir, B. (2022). Unsupervised contrastive hashing for cross-modal retrieval in remote sensing. In *ICASSP* (pp. 4463–4467). IEEE.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E. Z., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS* (pp. 8024–8035).

Patrick, M., Huang, P., Asano, Y. M., Metze, F., Hauptmann, A. G., Henriques, J. F., & Vedaldi, A. (2021). Support-set bottlenecks for video-text representation learning. In *ICLR*. OpenReview.net.

Qi, M., Qin, J., Yang, Y., Wang, Y., & Luo, J. (2021). Semantics-aware spatial-temporal binaries for cross-modal video retrieval. *IEEE Transactions on Image Processing, 30*, 2989–3004.

Radenovic, F., Dubey, A., Kadian, A., Mihaylov, T., Vandenhende, S., Patel, Y., Wen, Y., Ramanathan, V., & Mahajan, D. (2023). Filtering, distillation, and hard negatives for vision-language pre-training. In *CVPR* (pp. 6967–6977). IEEE.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. In *ICML*, volume 139 (pp. 8748–8763). PMLR.

Rasiwasia, N., Pereira, J. C., Coviello, E., Doyle, G., Lanckriet, G. R. G., Levy, R., & Vasconcelos, N. (2010). A new approach to cross-modal multimedia retrieval. In *ACM Multimedia* (pp. 251–260). ACM.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. CoRR, abs/1910.01108.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV* (pp. 618–626). IEEE Computer Society.

Shen, Y., Liu, L., & Shao, L. (2019). Unsupervised binary representation learning with deep variational networks. *International Journal of Computer Vision, 127*(11–12), 1614–1628.

Shen, H. T., Liu, L., Yang, Y., Xu, X., Huang, Z., Shen, F., & Hong, R. (2021). Exploiting subspace relation in semantic labels for cross-modal hashing. *IEEE Transactions on Knowledge and Data Engineering, 33*(10), 3351–3365.

Shi, Y., Chung, Y. (2021). Efficient cross-modal retrieval via deep binary hashing and quantization. In *BMVC* (p. 409). BMVA.

Shin, A., Ishii, M., & Narihira, T. (2022). Perspectives and prospects on transformer architecture for cross-modal tasks with language and vision. *International Journal of Computer Vision, 130*(2), 435–454.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *ICLR*.

Song, Y., & Soleymani, M. (2019). Polysemous visual-semantic embedding for cross-modal retrieval. In *CVPR* (pp. 1979–1988). Computer Vision Foundation/IEEE.

Song, J., Yang, Y., Yang, Y., Huang, Z., & Shen, H. T. (2013). Inter-media hashing for large-scale retrieval from heterogeneous data sources. In *SIGMOD* (pp. 785–796). ACM.

Song, J., He, T., Gao, L., Xu, X., Hanjalic, A., & Shen, H. T. (2020). Unified binary generative adversarial network for image retrieval and compression. *International Journal of Computer Vision, 128*(8), 2243–2264.

Su, S., Zhong, Z., & Zhang, C. (2019). Deep joint-semantics reconstructing hashing for large-scale unsupervised cross-modal retrieval. In *ICCV* (pp. 3027–3035). IEEE.

Sun, C., Latapie, H., Liu, G., & Yan, Y. (2022). Deep normalized cross-modal hashing with bi-direction relation reasoning. In *CVPRW* (pp. 4937–4945). IEEE.

Sun, C., Song, X., Feng, F., Zhao, W. X., Zhang, H., & Nie, L. (2019). Supervised hierarchical cross-modal hashing. In *SIGIR* (pp. 725–734). ACM.

Tan, W., Zhu, L., Guan, W., Li, J., & Cheng, Z. (2022). Bit-aware semantic transformer hashing for multi-modal retrieval. In *SIGIR* (pp. 982–991). ACM.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In *ICML*, volume 139 (pp. 10347–10357). PMLR.

Tu, J., Liu, X., Lin, Z., Hong, R., & Wang, M. (2022). Differentiable cross-modal hashing via multimodal transformers. In *ACM Multimedia* (pp. 453–461). ACM.

Tu, R., Mao, X., Lin, Q., Ji, W., Qin, W., Wei, W., & Huang, H. (2023). Unsupervised cross-modal hashing via semantic text mining. *IEEE Transactions of Multimedia, 25*, 8946–8957.

van den Oord, A., Li, Y., & Vinyals, O. (2018). Representation learning with contrastive predictive coding. CoRR, abs/1807.03748.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In *NeurIPS* (pp. 5998–6008).

Wang, W., Shen, Y., Zhang, H., Yao, Y., & Liu, L. (2020b). Set and rebase: Determining the semantic graph connectivity for unsupervised cross-modal hashing. In *IJCAI* (pp. 853–859). ijcai.org.

Wang, Y., Wang, J., Chen, B., Zeng, Z., & Xia, S. (2023). Contrastive masked autoencoders for self-supervised video hashing. In *AAAI* (pp. 2733–2741). AAAI.

Wang, J., Zeng, Z., Chen, B., Dai, T., & Xia, S. (2022a). Contrastive quantization with code memory for unsupervised image retrieval. In *AAAI*, (pp. 2468–2476). AAAI.

Wang, J., Zeng, Z., Chen, B., Wang, Y., Liao, D., Li, G., Wang, Y., & Xia, S. (2022b). Hugs are better than handshakes: Unsupervised cross-modal transformer hashing with multi-granularity alignment. In *BMVC* (p. 1035). BMVA.

Wang, X., Zhu, L., and Yang, Y. (2021b). T2VLAD: global-local sequence alignment for text-video retrieval. In *CVPR* (pp. 5079–5088). Computer Vision Foundation/IEEE.

Wang, J., Liu, W., Kumar, S., & Chang, S. (2016). Learning to hash for indexing big data - A survey. *Proceedings of the IEEE, 104*(1), 34–57.

Wang, L., Yang, J., Zareapoor, M., & Zheng, Z. (2021). Cluster-wise unsupervised hashing for cross-modal similarity search. *Pattern Recognition, 111*, 107732.

Wang, Z., Zhang, Z., Luo, Y., Huang, Z., & Shen, H. T. (2021). Deep collaborative discrete hashing with semantic-invariant structure construction. *IEEE Transactions of Multimedia, 23*, 1274–1286.

Wang, J., Zhang, T., Song, J., Sebe, N., & Shen, H. T. (2018). A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 40*(4), 769–790.

Wang, T., Zhu, L., Cheng, Z., Li, J., & Gao, Z. (2020). Unsupervised deep cross-modal hashing with virtual label regression. *Neurocomputing, 386*, 84–96.

Weiss, Y., Torralba, A., & Fergus, R. (2008). Spectral hashing. In *NeurIPS* (pp. 1753–1760). Curran Associates, Inc.

Wu, G., Lin, Z., Han, J., Liu, L., Ding, G., Zhang, B., & Shen, J. (2018). Unsupervised deep hashing via binary latent factor models for large-scale cross-modal retrieval. In *IJCAI* (pp. 2854–2860). ijcai.org.

Wu, W., Luo, H., Fang, B., Wang, J., & Ouyang, W. (2023). Cap4video: What can auxiliary captions do for text-video retrieval? In *CVPR* (pp. 10704–10713). IEEE.

Xu, J., Mei, T., Yao, T., & Rui, Y. (2016). MSR-VTT: A large video description dataset for bridging video and language. In *CVPR* (pp. 5288–5296). IEEE Computer Society.

Yang, J., Bisk, Y., Gao, J. (2021). Taco: Token-aware cascade contrastive learning for video-text alignment. In *ICCV* (pp. 11542–11552). IEEE.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J. G., Salakhutdinov, R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS* (pp. 5754–5764).

Yang, D., Wu, D., Zhang, W., Zhang, H., Li, B., & Wang, W. (2020). Deep semantic-alignment hashing for unsupervised cross-modal retrieval. In *ICMR* (pp. 44–52).

Yao, L., Huang, R., Hou, L., Lu, G., Niu, M., Xu, H., Liang, X., Li, Z., Jiang, X., & Xu, C. (2022). FILIP: Fine-grained interactive language-image pre-training. In *ICLR*. OpenReview.net.

Ye, M., Shen, J., Lin, G., Xiang, T., Shao, L., & Hoi, S. C. H. (2022). Deep learning for person re-identification: A survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(6), 2872–2893.

Yu, H., Ding, S., Li, L., & Wu, J. (2022). Self-attentive CLIP hashing for unsupervised cross-modal retrieval. In *ACM Multimedia* (pp. 8:1–8:7). ACM.

Yu, Y., Kim, J., & Kim, G. (2018). A joint sequence fusion model for video question answering and retrieval. In *ECCV*, volume 11211 (pp. 487–503). Springer.

Yu, T., Yang, Y., Li, Y., Liu, L., Fei, H., & Li, P. (2021b). Heterogeneous attention network for effective and efficient cross-modal retrieval. In *SIGIR* (pp. 1146–1156). ACM.

Yu, J., Zhou, H., Zhan, Y., & Tao, D. (2021a). Deep graph-neighbor coherence preserving network for unsupervised cross-modal hashing. In *AAAI* (pp. 4626–4634). AAAI.

Yu, T., Meng, J., Fang, C., Jin, H., & Yuan, J. (2020). Product quantization network for fast visual search. *International Journal of Computer Vision, 128*(8), 2325–2343.

Zala, A., Cho, J., Kottur, S., Chen, X., Oguz, B., Mehdad, Y., & Bansal, M. (2023). Hierarchical video-moment retrieval and step-captioning. In *CVPR* (pp. 23056–23065). IEEE.

Zeng, Z., Wang, J., Chen, B., Wang, Y., & Xia, S. (2022). Motion-aware graph reasoning hashing for self-supervised video retrieval. In *BMVC* (p. 82). BMVA.

Zhang, T., Du, C., & Wang, J. (2014). Composite quantization for approximate nearest neighbor search. In *ICML*, volume 32 (pp. 838–846). JMLR.org.

Zhang, J., Peng, Y., & Yuan, M. (2018). Unsupervised generative adversarial cross-modal hashing. In *AAAI* (pp. 539–546). AAAI.

Zhang, Z., Lai, Z., Huang, Z., Wong, W. K., Xie, G., Liu, L., & Shao, L. (2019). Scalable supervised asymmetric hashing with semantic and latent factor embedding. *IEEE Transactions on Image Processing, 28*(10), 4803–4818.

Zhang, P., Li, Y., Huang, Z., & Xu, X. (2022). Aggregation-based graph convolutional hashing for unsupervised cross-modal retrieval. *IEEE Transactions of Multimedia, 24*, 466–479.

Zhang, P., Luo, Y., Huang, Z., Xu, X., & Song, J. (2021). High-order nonlocal hashing for unsupervised cross-modal retrieval. *World Wide Web, 24*(2), 563–583.

Zhang, Z., Luo, H., Zhu, L., Lu, G., & Shen, H. T. (2023). Modality-invariant asymmetric networks for cross-modal hashing. *IEEE Transactions on Knowledge and Data Engineering, 35*(5), 5091–5104.

Zhang, J., & Peng, Y. (2020). Multi-pathway generative adversarial hashing for unsupervised cross-modal retrieval. *IEEE Transactions of Multimedia, 22*(1), 174–187.

Zhang, Z., Wang, J., Zhu, L., Luo, Y., & Lu, G. (2023). Deep collaborative graph hashing for discriminative image retrieval. *Pattern Recognition, 139*, 109462.

Zheng, C., Zhu, L., Lu, X., Li, J., Cheng, Z., & Zhang, H. (2020). Fast discrete collaborative multi-modal hashing for large-scale multimedia retrieval. *IEEE Transactions on Knowledge and Data Engineering, 32*(11), 2171–2184.

Zhong, Y., Arandjelovic, R., & Zisserman, A. (2018). Ghostvlad for set-based face recognition. In *ACCV*, volume 11362 (pp. 35–50). Springer.

Zhong, Z., Zheng, L., Cao, D., & Li, S. (2017). Re-ranking person re-identification with k-reciprocal encoding. In *CVPR* (pp. 3652–3661). IEEE Computer Society.

Zhou, J., Ding, G., & Guo, Y. (2014). Latent semantic sparse hashing for cross-modal similarity search. In *SIGIR* (pp. 415–424). ACM.

Zhu, X., Huang, Z., Shen, H. T., & Zhao, X. (2013). Linear cross-modal hashing for efficient multimedia search. In *ACM Multimedia* (pp. 143–152). ACM.

Zhu, Y., Kiros, R., Zemel, R. S., Salakhutdinov, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV* (pp. 19–27). IEEE Computer Society.

Zhu, H., Long, M., Wang, J., & Cao, Y. (2016). Deep hashing network for efficient similarity retrieval. In *AAAI* (pp. 2415–2421). AAAI.

Zhuo, Y., Li, Y., Hsiao, J., Ho, C., & Li, B. (2022). Clip4hashing: Unsupervised deep hashing for cross-modal video-text retrieval. In *ICMR*, (pp. 158–166). ACM.

Zhu, L., Wu, X., Li, J., Zhang, Z., Guan, W., & Shen, H. T. (2023). Work together: Correlation-identity reconstruction hashing for unsupervised cross-modal retrieval. *IEEE Transactions on Knowledge and Data Engineering, 35*(9), 8838–8851.