AMES: Asymmetric and Memory-Efficient Similarity Estimation for Instance-level Retrieval

Pavel Suma[1], Giorgos Kordopatis-Zilos[1], Ahmet Iscen[2], and Giorgos Tolias[1]

[1]VRG, FEE, Czech Technical University in Prague [2]Google DeepMind

Fig. 1: AMES estimates the image-to-image similarity based on local descriptor sets. Top: 100 (query) *vs*. 100 (database) descriptors. Bottom: memory-efficient and asymmetric variant with 100 *vs*. 30 local descriptors. Circle size reflects descriptor importance within AMES; descriptors of the common object get higher importance.

Abstract. This work investigates the problem of instance-level image retrieval re-ranking with the constraint of memory efficiency, ultimately aiming to limit memory usage to 1KB per image. Departing from the prevalent focus on performance enhancements, this work prioritizes the crucial trade-off between performance and memory requirements. The proposed model uses a transformer-based architecture designed to estimate image-to-image similarity by capturing interactions within and across images based on their local descriptors. A distinctive property of the model is the capability for asymmetric similarity estimation. Database images are represented with a smaller number of descriptors compared to query images, enabling performance improvements without increasing memory consumption. To ensure adaptability across different applications, a universal model is introduced that adjusts to a varying number of local descriptors during the testing phase. Results on standard benchmarks demonstrate the superiority of our approach over both handcrafted and learned models. In particular, compared with current state-of-the-art methods that overlook their memory footprint, our approach not only attains superior performance but does so with a significantly reduced memory footprint. The code and pretrained models are publicly available at: `https://github.com/pavelsuma/ames`

1 Introduction

Image retrieval approaches typically focus on optimizing performance metrics, often effectively due to modern representations [1, 5, 13, 40]. Higher-dimensional representations provide a better estimation of the image-to-image similarity and result in better retrieval accuracy [32, 43]. However, what is typically overlooked is the memory footprint requirements of storing such representations. Storing high-dimensional, and usually dense, representations results in a large memory footprint, making them not applicable for web-scale applications. Therefore, it becomes important to consider performance alongside the memory footprint. This work focuses precisely on this dual aspect, which has been underestimated in the literature. We analyze the trade-off between performance and memory usage in the realm of instance-level image retrieval. Specifically, we investigate and improve this trade-off assuming the use of both local and global descriptors, ideally with both provided by a universal model [8].

Global descriptors offer a convenient way to handle image retrieval in large databases due to computational and memory benefits. They provide a fast way to perform retrieval and allow additional speed-up with approximate nearest neighbor search techniques [4, 25]. At the same time, storing the representation of database images comes with a low footprint, which is further reduced by dimensionality reduction or other compression techniques [19, 23, 39]. Nevertheless, despite the high performance achieved in multiple benchmarks due to deep models, global descriptors have drawbacks. This is especially the case when it comes to severe background clutter, such as small objects and large occlusions [16].

One way to improve performance is to represent an image with a set of local descriptors while using a similarity function to compare two local descriptor sets. The similarity function can be hand-crafted [41, 59] or include learnable parts [27, 58]. An important aspect of such approaches is that the local descriptors are usually used at a *re-ranking stage* [36, 58], after a more efficient initial ranking stage, *e.g.* retrieval with global descriptors. In this fashion, the high computational cost of using local descriptors becomes less of a weakness if the number of images to re-rank is limited. Their large memory footprint, however, still remains a major drawback.

We explore and optimize the trade-off between retrieval performance and the memory requirements, especially in a low memory footprint regime such as 1KB per image, to allow scalable solutions. Our methodology targets both major factors regarding memory, namely operating with a low number of local descriptors and the footprint per descriptor. We design an image-to-image similarity model that uses transformers to capture within and across image interactions. The underlined trade-off is optimized in a four-fold way: (i) through asymmetric similarity estimation where the query image is represented by a higher number of local descriptors, without affecting the database footprint (Figure 1), (ii) training the model to operate on binary input vectors learned in an end-to-end manner, (iii) with a distillation process where the teacher model uses a richer representation than the student, (iv) by a proper combination of the similarity given by global and local descriptors during the re-ranking stage.

The number of local descriptors per database or query image may need to vary in a practical system to control both the memory and the retrieval speed. We obtain a universal model, tackling all cases, by varying their number per image during the training. This approach successfully alleviates the sensitivity in transformers observed during our experiments, where performance tends to decline due to discrepancies in the number of input tokens between the training and testing phases. The list of contributions is outlined below.

- We present the first systematic study that investigates the trade-off between performance and memory consumption in the context of deep models and their representations for instance-level image retrieval.
- We introduce AMES, a new similarity estimation architecture and training strategy inspired by existing techniques, specifically tailored to address our unique requirements and objectives.
- We reveal the sensitivity of transformers to changes in the number of the input tokens and develop a universal model that exhibits robustness to such changes, ensuring stable performance across different operating conditions.

2 Related work

Similarity using global descriptors. Traditional methods in image retrieval extract global representations by aggregating hand-crafted local features, *e.g.* BoW [10,50], Fisher vectors [35,45], VLAD [18], or Triangulation embedding [24], and compute a simple metric, *e.g.* cosine similarity, to perform the retrieval. The memory footprint of these representations is further improved by reducing their dimensionality with different variants of PCA [19, 39], or product quantization (PQ) [23]. Since the advance of deep networks, utilizing learned, *i.e.* not hand-crafted, descriptors is also a popular choice for image retrieval [1, 31]. Early work in the field employs neural networks trained for classification and transfers them to image retrieval by aggregating internal activation maps [5, 40]. More recent methods directly train a neural network specifically for image retrieval by optimizing for robust global representations through different metric learning losses [42,47], sampling of the training data [30,51,54,65], and architectures [52, 53]. In recent work, SuperGlobal [48] descriptors are extracted via a repurposed architecture of an already trained retrieval model that becomes effective with appropriate hyper-parameter tuning. In addition to global, some models derive local descriptors too through separate network branches [8, 55, 69].

Similarity with local descriptors. In contrast to aggregating local representations into a global one, specific metrics are designed to compare the local descriptors and estimate the image-to-image similarity. Traditional approaches follow the BoW paradigm but utilize match kernels to compute the similarity between local features assigned to the same visual words [20, 59]. Hamming Embedding [20] opts for binary codes to significantly reduce the memory as it requires indexing additional features. The kernels are combined with weighing to reduce the burstiness effect [21] or to account for local density in the embedding

space [2]. ASMK [59] takes a middle route and aggregates only the local features belonging to the same visual word. This reduces the memory footprint of the stored database features, which is further reduced with binarization. These methods are shown to be outperformed by learnable re-ranking approaches [58] or improved with appropriate representation learning [63].

Different forms of geometric constraints are shown helpful for image similarity estimation, *e.g.* consistency in spatial neighborhoods [50], angles and scales [22], or in relative spatial ordering [14,68]. Geometric verification through RANSAC-like [11] procedures is one of the most popular methods to estimate the similarity of not only hand-crafted local descriptors [36,37] but also local features extracted from deep networks [8,33,49]. This is a time-consuming re-ranking approach with an additional memory overhead due to storing local feature geometry.

Recent advancements in deep learning have led to the development of models that estimate image-to-image similarity based on local descriptors [58,70]. These models allow for the direct optimization of similarity computation for specific tasks, enhancing performance. RRT [58] and R^2Former [70] improve over spatial verification in performance and processing time but require to store several hundred full-precision descriptors. CVNet [27] processes cross-scale correlation between dense feature maps with a 4D convolutional network, requiring even more memory. The proposed model draws inspiration from the RRT architecture and effectively extends it. To benchmark our model, we directly compare its performance to RRT, R^{2}Former, and CVNet, demonstrating its superiority in terms of performance and memory.

Furthermore, our architecture design incorporates ideas from image matching networks [29,46,57]. These approaches utilize a combination of self-attention and cross-attention layers to facilitate information exchange within individual images as well as across the two images. By integrating these concepts, our model allows for a more effective similarity estimation approach.

Asymmetric similarity is used in prior work to optimize the performance and memory trade-off by comparing binary to full-precision vectors [17]. This approach is complementary to our contributions. Recently, a variety of deep learning methods optimizes the performance *vs*. query time trade-off using light-weight query processing [7,56,66] or model ensembling on the database side [67]. Our universal model is able to control query time by operating on a varying number of query local descriptors.

Transformer sensitivity to token-set size discrepancy. In natural language processing (NLP), tasks such as question answering inherently involve sentences of varying lengths during training and testing [61]. Models exhibit some robustness to such variations unless significantly longer sentences are encountered during testing [9,44]. Inspired by this observation, we propose to train our models with a varying number of input tokens. A similar phenomenon is observed in Flexible Vision Transformers (ViTs) [6] and CAPE [28], where variations in patch or image size are considered that consequently affect the token-set size.

3 Method for memory-efficient similarity estimation

We first introduce the task of memory-efficient retrieval and then present the proposed approach for training an image-to-image similarity model that operates on two sets of local descriptors as input while keeping the memory footprint low.

3.1 Problem formulation

The objective of image retrieval is to search a database of images $\mathcal{X}$ using a given query image q and retrieve relevant images. In this work, relevance is defined in terms of depicting the same specific object, focusing on instance-level retrieval. The image-to-image similarity between the query and an image $x \in \mathcal{X}$ is measured by the function $s(x, q) \in \mathbb{R}$. Subsequently, the results are sorted to generate a ranking of the database images.

In contrast to the majority of prior work, which predominantly emphasizes performance metrics, we argue that the quality of a retrieval approach should be considered as a composite of both performance and memory footprint. The latter concerns the memory required to store the representation of image x necessary for similarity estimation. Memory allocation for the query is less crucial as it is released at the end of the retrieval process. Consequently, we explore and find merit in asymmetric setups within this work.

3.2 Preliminaries

A single-vector image representation, referred to as *global descriptor*, is a memory-efficient and computationally efficient way of computing similarity. The ℓ_2-normalized global descriptors for database image x and query image q are denoted by $\mathbf{x} \in \mathbb{R}^{d_g}$ and $\mathbf{q} \in \mathbb{R}^{d_g}$, respectively. Image similarity is then simplified to the dot product $s_g(x, q) = \mathbf{x}^\top \mathbf{q}$, also referred to as *global similarity*.

To enhance retrieval quality, a common approach involves representing each image with multiple vectors, such as *local descriptors*. Each database image x is then represented by L_x D-dimensional vectors in $X \in \mathbb{R}^{D \times L_x}$, and the query q by L_q vectors in $Q \in \mathbb{R}^{D \times L_q}$. Although working with order-less vector sets, we employ matrix notation for the sake of clarity. Image-to-image similarity is defined as $s_l(x, q) = S(X, Q)$, referred to as *local similarity*, where $S : \mathbb{R}^{D \times L_x} \times \mathbb{R}^{D \times L_q} \to \mathbb{R}$ is a similarity function operating on two vector sets. The two sets do not need to have the same size. While various options exist for designing and implementing S, the computational complexity is inevitably higher compared to the global similarity estimation. A practical solution is to estimate global similarity in an initial ranking stage, and local similarity during re-ranking of the most similar images according to the first step. However, this approach requires storing both global and local representations for image x. Depending on the values of d_g, D, L_x, this may quickly become impractical for large databases. Therefore, maintaining a low memory footprint is crucial.

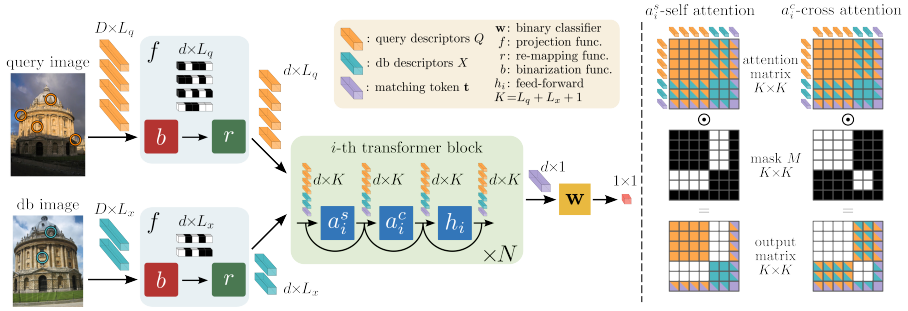

Fig. 2: Overview of the AMES model designed for estimating image-to-image similarity when provided two local descriptor sets, where one set has a smaller size. Descriptors are processed by projection (f) comprised of binarization (b) and re-mapping to the real coordinate space (r). During testing, binarization (re-mapping) is performed offline (online); therefore, we need to store only the binary vectors for the database images. Descriptors, together with a learnable matching token, form the input token set for a transformer-based architecture, which, together with a binary classifier, estimate the similarity. Sequentially, inter-image (self) and intra-image (cross) attention is performed by standard self-attention blocks alongside appropriate masking.

3.3 Proposed approach - AMES

Given the local descriptor sets X and Q for images x and q, respectively, we design the image-to-image similarity function $S(X, Q)$ with a transformer-based network. Together with the training and testing strategy, we refer to the proposed approach as "Asymmetric and Memory-Efficient Similarity" or AMES.

AMES architecture. We use a transformer-based architecture [62] in a processing pipeline that interchanges between self-attention and cross-attention to capture both intra-image and inter-image interactions between local descriptors. As a first step, the representation space undergoes transformation, optionally involving dimensionality reduction, through a projection function $f : \mathbb{R}^D \to \mathbb{R}^d$. This transformation is applied individually to each local descriptor, with the application to the entire set represented as $f(X)$ and $f(Q)$.

Each local descriptor becomes a transformer token whose input is composed by concatenation of the projected X and Q, and a *matching token* $\mathbf{t} \in \mathbb{R}^d$. We include this token by following common practice, similar to the classification token [58,62], initialize it by a random vector, and treat it as a learnable variable. This plays the role of information aggregation from both images through the attention mechanism. In summary, the input to AMES consists of $K = L_x + L_q + 1$ tokens and is denoted by

$$Z_0 = [\ f(X) \quad f(Q) \quad \mathbf{t}\ ]. \tag{1}$$

The network consists of N consecutive processing blocks that are identical in design. Each block i, $i \leq N$, comprises two consecutive attention operations and a feed-forward layer with residual connections. The output of the i-th block

$$Z_i = [\ X_i \quad Q_i \quad \mathbf{t}_i\ ], \tag{2}$$

matches the input to $(i{+}1)$-th block, while processing by a block is given by

$$\hat{Z}_i = a^s_{i+1}(Z_i; M^s) + Z_i \tag{3}$$

$$\tilde{Z}_i = a^c_{i+1}(\hat{Z}_i; M^c) + \hat{Z}_i \tag{4}$$

$$Z_{i+1} = \quad h_{i+1}(\tilde{Z}_i) + \tilde{Z}_i, \tag{5}$$

where (3) and (4) correspond to self and cross attention, respectively. In the former (latter), local descriptors across images (within the same image) do not attend to each other. Masks $M^s, M^c \in \{0,1\}^{K\times K}$ are binary matrices that allow to perform both operations via masked attention across all input tokens[1]. Mask M^s (M^c) has zeros for elements across the two images (within the same image) and ones otherwise. The feed-forward process in (5) is an MLP operating independently per token. Note that the matching token $\mathbf{t}$ is never masked in order to aggregate information from both images and both operations simultaneously. We treat the output matching token $\mathbf{t}_N$, at the last block, as a representation of the image pair and pass it to a linear binary classifier represented by vector $\mathbf{w} \in \mathbb{R}^d$. The final similarity, with the use of a sigmoid function σ, ranges in $[0,1]$ and is given by

$$S(X,Q) = \sigma(\gamma \mathbf{t}_N^\top \mathbf{w}), \tag{6}$$

where γ is a temperature variable that is set and fixed to 1 during training. The overall architecture is shown in Figure 2.

Local descriptor footprint. The memory-demanding variant of AMES implements the projection function f with a linear layer. The output dimensionality d controls the memory footprint since $f(X)$ is part of the model input and is stored for each database image. We refer to it as *full-precision* (fp) variant.

To reduce the memory footprint, we consider an alternative projection function that consists of binarization function b mapping local descriptors to the Hamming space and re-mapping function r projecting binary descriptors back to the real coordinate space. The projection is given by $f = r \circ b$, with the benefit of only storing $b(X)$, which is more compact, and applying r during query time to recover the real-valued vector. This is what we refer to as *binary* (bin) variant.

Formally, given an arbitrary local descriptor $\mathbf{u}$, the binarization function $b\colon \mathbb{R}^D \to \{-1,1\}^d$ performs dimensionality reduction and vector binarization by $b(\mathbf{u}) = \operatorname{sgn}(\mathbf{u}W)$, where $W \in \mathbb{R}^{D\times d}$ is a matrix of trainable parameters, and sgn denotes the element-wise sign function, which is not differentiable. Therefore, we use a smooth approximation [26] given by $b(\mathbf{u}) = \operatorname{erf}(\mathbf{u}W/\sqrt{2\delta^2})$, where erf is the error function, and δ a hyper-parameter that controls the smoothness of the approximation. During training, we use the smooth variant, allowing gradients to flow, while during inference, we use sign-based binarization. The re-mapping function r consists of a linear layer and Layer Normalization [3].

[1] Both a^s_i and a^c_i perform attention across all input tokens, but it is the masking that makes them reduce down to a type of self-attention (within image) and cross-attention (across image). Super-scripts are there to indicate different learnable parameters; each block has a different set of learnable parameters.

AMES testing. We describe the retrieval stage with AMES and the memory requirements of different variants.

Global-local similarity ensemble: We assume access to both global and local descriptors but incorporate local similarity only during the re-ranking stage, where the image-to-image similarity is then expressed as a linear combination of global and local similarities

$$s(x,q) = \lambda s_g(x,q) + (1-\lambda) s_l(x,q), \tag{7}$$

referred to as *ensemble similarity*. Although most prior work [33,49,58] considers $\lambda = 0$, *i.e.* re-ranking solely with local similarity, the ensemble (7) is shown to bring benefits in recent work [8, 27]. To effectively perform the similarity combination, we tune two hyper-parameters, namely λ and temperature γ in (6) to properly align and balance the two similarity distributions. The tuning is performed with grid search according to performance on a validation set.

Asymmetric similarity: We differentiate between the number of local descriptors per image used during training and testing by L_q^{train}, L_x^{train}, and L_q^{test}, L_x^{test}, respectively. The footprint scales linearly with L_x^{test}, whereas a higher number proves advantageous [58], up to a certain extent, thus establishing the performance *vs*. memory trade-off. Asymmetric similarity estimation, achieved by setting $L_q^{\text{test}} > L_x^{\text{test}}$, is a viable option that avoids compromising database memory and demonstrates performance improvements in our experiments.

Varying number of local descriptors: We treat L_q^{test} as a hyper-parameter that increases the model complexity according to the underlined limits of query time, *e.g.* per user or environment. Despite L_x^{test} being fixed for a particular database, a different system may need to use a different value. Therefore, a universal model handling different values both for L_x^{test} and L_q^{test} is necessary. The training of a universal model and its challenges are discussed in the following section.

AMES training. We perform supervised training based on labels of image pairs, optionally combined with a distillation process.

Universal AMES: When there is a mismatch in the local descriptor set size between the training and testing phases, we observe a performance decline. The sensitivity is attributed to the attention blocks and is due to the change in the number of tokens. A naive solution is to separately train a different model for each size, *i.e.* many different models trained with $L_q^{\text{test}} = L_q^{\text{train}}$ and $L_x^{\text{test}} = L_x^{\text{train}}$. We refer to these models as *specific models*. Nevertheless, this solution significantly increases the training complexity and requires maintaining multiple models for inference of different combinations of $(L_x^{\text{test}}, L_q^{\text{test}})$. Instead, we train a single model by randomly sampling values for L_x^{train} and L_q^{train} per batch and using a subset of the images' local descriptors[2]. This *universal model* performs well when tested across different set sizes, even compared to the specific models.

Supervised training: An image pair (x, q) in the training set is associated with label $y_{xq} = 1$ if the two images are relevant (matching) or label $y_{xq} = 0$,

[2] The order of the input images does not matter; differentiating between sizes for image x and q is important for the testing setup, *e.g.* asymmetry, but not for the training.

otherwise. Using the pair labels, we perform label-balanced training with back-propagation through binary cross-entropy loss, which for a specific pair is given by $\mathcal{L}_{\text{bce}}(x,q) = -y_{xq} \log s_l(x,q) - (1-y_{xq}) \log(1-s_l(x,q))$. In our task, this corresponds to a contrastive loss pushing the similarity of matching pairs to 1 and non-matching to 0. We optimize the parameters of functions a_i^s, a_i^c, h_i for $i \in \{1,...,N\}$, along with $\mathbf{w}$, W and matching token $\mathbf{t}$. We find it beneficial to initialize W of the binarization layer with the result of ITQ [12] trained on a large set of local descriptors.

Distillation-guided supervised training: We improve the (student) binary variant of AMES by using an already trained full-precision variant as a teacher. During this distillation procedure, the teacher receives a large set of local descriptors as input for both images, while the student operates with a varying set size that is smaller than that of the teacher. At each batch, the student descriptor set is a subset of the teacher one. The distillation process operates in the latent representation space and not in the output space of scalar similarities, where the supervised loss is applied. In particular, it operates right before the classifier, at the output of the last transformer block. Let $Z_N^{(t)}$ and $Z_N^{(s)}$ be those output matrices for the teacher and the student model, respectively, for the same image pair, and $\bar{Z}_N^{(t)}$ a trimmed version of $Z_N^{(t)}$ to contain the tokens that correspond to those of the student model. Distillation is performed via ℓ_2 loss as

$$\mathcal{L}_{\text{dis}} = \frac{1}{dK} \left\| \bar{Z}_N^{(t)} - Z_N^{(s)} \right\|_F, \tag{8}$$

where $\|\cdot\|_F$ is the Frobenius norm. In that way, the transformer of the student network is guided to generate similar output tokens as the transformer of the teacher even though it operates with lower precision local descriptors, *i.e.* binarized, and only with a subset of the local descriptors. To this end, we optimize a weighted sum of the two losses as follows $\mathcal{L} = \mathcal{L}_{\text{bce}} + \beta \mathcal{L}_{\text{dis}}$, where β is a hyper-parameter that tunes the impact of $\mathcal{L}_{dis}$.

4 Experiments

4.1 Experimental setup

Datasets and evaluation metrics. We use Google Landmarks dataset v2 [64] (GLDv2) for training, validation, and testing. From the *clean* part of the training set, we randomly select 24K classes with more than 10 images and at most 500 images per class. The final training set contains 754K images, corresponding to roughly half of the clean part. We use the 761K *index* images as the database and the 379 *public* and 750 *private* query images for validation and testing, respectively. Performance is evaluated using mean Average Precision at top-100 images (mAP@100). $\mathcal{R}$Oxford [36, 38] and $\mathcal{R}$Paris [37, 38] datasets are used together with the accompanying 1M distractor images. Performance is evaluated with mAP. We present results across the *medium* and *hard* setups of both datasets by averaging the four performance values, denoted by $\mathcal{R}$OP+1M.

Global and local descriptors. AMES is generic and applicable to any type of local descriptor. For global descriptors, we use CVNet [27] trained on GLDv2 with a ResNet101 [15] backbone, its SuperGlobal (SG) [48] training-free enhancement, and pretrained DINOv2 [34] through its CLS token. For local descriptors, we extract descriptors from either the CVNet backbone or DINOv2. Unless mentioned otherwise, our default choices are the SG global and the CVNet local descriptors. We do not consider DINOv2 as the default option since $\mathcal{R}$Oxford and $\mathcal{R}$Paris are listed among the datasets used for its self-supervised training. Following prior work [8,60,69], we train a local feature detector in order to select the L strongest local descriptors per image during testing; see the supplementary material for more details.

The dimensionality of the global descriptors is equal to $d_g = 2048$ or $d_g = 768$ for CVNet or DINOv2, and $d = 128$ for the local descriptors. Global descriptors are either full-precision (fp)[3] or are compressed with product quantization (PQ) [23]. PQ is parameterized to work with 1 byte per sub-space, and the sub-space dimension is set equal to 1, 4, or 8, denoted by PQ1, PQ4, or PQ8, respectively. Unless otherwise stated, PQ8 is used. The reported memory corresponds to the storage of global and local descriptors for database images. The storage for the network parameters is not included since it is a constant and a negligible amount.

AMES training and testing. During mini-batch sampling for the universal model training, L_q^{train} and L_x^{train} are randomly sampled in range $[10, 400]$ and are, therefore, not necessarily equal to each other. We use $N = 5$ blocks in AMES for all settings, which has the same number of parameters as RRT for their default choice of 6 blocks. During testing, we re-rank the top-m images obtained with global similarity. Local descriptor set sizes L_x^{test} and L_q^{test} correspond to the database images and query, respectively. The main variant used in our experiments, unless otherwise stated, is the universal model with binary representation after distillation, $m = 1600$, $L_q^{\text{test}} = 600$, and varying L_x^{test} to control the memory. For each experiment, we train three networks with different seeds and report average performance.

Existing approaches. Using the same global/local descriptors, we perform a direct comparison with ASMK [59][4], RRT [58], and R^2Former [70]. We use their official publicly-available implementation and their default model hyperparameters. All models are trained within our implementation framework with varying descriptor set size and the same projection function f at their input for fair comparison. The same ensemble similarity tuning strategy is used for all these methods. We compare with the reported results for CVNet [27] with the same representation network as in our setup. We exclude it from the trade-off comparison because it performs dense similarity estimation that is unsuitable for low-memory regimes.

[3] Variant (fp) for global and local descriptors: Half-precision floating points are used.
[4] ASMK's memory is based on the effective number of descriptors after aggregation.

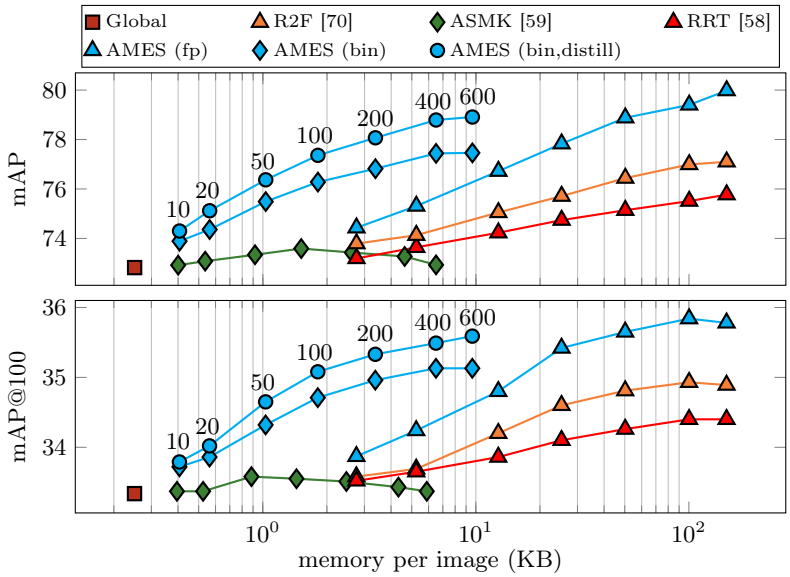

Fig. 3: Performance *vs.* memory trade-off on $\mathcal{R}$OP+1M (top) and GLDv2 (bottom). All methods use global descriptors with PQ8 for initial ranking and ensemble similarity to re-rank $m = 1600$ images. We vary the number of local descriptors L_x^{test} for database images, which is shown with text labels, indicatively, for one variant. Binary/full-precision local descriptors denoted by bin/fp. All methods are trained and tested by us, within the same implementation framework.

global desc.	**local desc.**	**re-ranking**			$\mathcal{R}$**OP+1M**	**GLDv2**
		approach	**top-m**			
DELG [8]	n/a	n/a	n/a	♡	51.6	24.1
	GV [8]	**DELG** [8]	100	♡	56.5	24.3
	RRT [58]	**DELG** [8]	100	♡	57.6	–
CVNet [27]	n/a	n/a	n/a	♡	67.6	32.5
	CVNet [27]	**CVNet re-rank** [27]	100	♡	73.0	34.9
		AMES (fp)	100	♠	72.2	35.3
		AMES (bin,distill)	100	♠	72.0	35.1
		CVNet re-rank [27]	400	♡	75.6	–
		AMES (fp)	400	♠	**75.8**	**35.5**
		AMES (bin,distill)	400	♠	75.4	35.1
SG [48]	n/a	n/a	n/a	♠	72.9	33.3
	CVNet [27]	**AMES** (fp)	1600	♠	80.0	35.8
		AMES (bin,distill)	1600	♠	78.9	35.6
	DINOv2 [34]	**AMES** (fp)	1600	♠	83.5	38.3
		AMES (bin,distill)	1600	♠	81.8	37.8
	n/a	**SG re-rank** [48]	1600	♠	80.5	34.1
	CVNet [27]	**SG re-rank** [48] + **AMES** (fp)	1600	♠	82.3	36.0
		SG re-rank [48] + **AMES** (bin,distill)	1600	♠	81.9	35.8
	DINOv2 [34]	**SG re-rank** [48] + **AMES** (fp)	1600	♠	**84.5**	**38.5**
		SG re-rank [48] + **AMES** (bin,distill)	1600	♠	83.3	38.0

Table 1: Comparison with the state-of-the-art. Backbone architectures are ResNet50 for DELG, ResNet101 for CVNet/SG, and ViT-B/14 for DINOv2. Global descriptors are in full precision. Scores for SG without and with re-ranking are with our best possible reproduction using the publicly available implementation. ♡: reported in the literature, ♠: evaluated by us. AMES is used with $L_x^{\text{test}} = 600$ and $L_q^{\text{test}} = 600$ local descriptors for the database and query image, respectively.

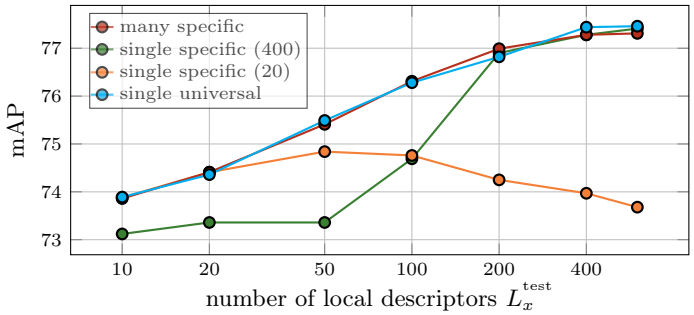

Fig. 4: Universal *vs*. specific AMES models. Single specific: one model trained with a fixed number of local descriptors L_x^{train} (value in brackets) and tested with varying L_x^{test}. Many specific: six models in total, trained and tested per number of local descriptors ($L_x^{\text{test}} = L_x^{\text{train}}$). All models are without distillation.

4.2 Results

Performance/memory comparison. Figure 3 shows the performance of the compared approaches on different memory settings. AMES achieves the best trade-off; it performs better than all other methods for the same memory or performs the same for much less memory. The binary variant significantly decreases memory over the full-precision variant at the cost of a small performance drop. On top of that, our distillation consistently recovers most of this performance loss. ASMK provides small improvements and does not benefit from more descriptors. We outperform R^2Former and RRT in all cases.

State-of-the-art (SoA) comparison. Reported results of SoA methods are shown in Table 1. AMES achieves SoA performance when combined with either CVNet or SuperGlobal as global descriptors and using the same value m as the competing approaches. In the direct comparison with CVNet re-ranking, we outperform it with a clear margin while our per-image memory footprint is significantly lower, *i.e*. 301KB and ∼1400KB for AMES (fp) and CVNet, respectively. Note that CVNet requires the full dense feature map representation. Our binary variant achieves very competitive performance requiring only ∼10KB, further optimizing performance-memory trade-off. In addition, AMES improves performance over the query expansion re-ranking approach of SuperGlobal [48] (AMES applied after SG re-rank), highlighting that SG re-rank is complementary in nature to AMES. In contrast, CVNet re-ranking does not improve over SG re-ranking (cf. Shao *et al*. [48]).

Universal *vs*. specific models. In Figure 4, we demonstrate performance for varying number of local descriptors per database image L_x^{test} used during testing. Specific models are trained with fixed L_x^{train} and either tested for $L_x^{\text{test}} = L_x^{\text{train}}$ (many specific) or tested for varying L_x^{test} (single specific). Our universal model achieves about the same performance as each of the many specific models, even though those are tested for the number of local descriptors they are trained with. Single specific variants demonstrate notable sensitivity to the discrepancy between L_x^{test} and L_x^{train}.

architecture	L_x^{test}	L_q^{test}	bin	dis	global	$\mathcal{R}$**OP+1M**	**GLDv2**	**mem.**
AMES	50	50	–	–	full	75.9	34.4	20.5
AMES	50	600	–	–	full	76.7	34.8	20.5
AMES	50	600	✓	–	full	75.6	34.3	8.8
AMES	50	600	✓	✓	full	76.5	34.6	8.8
AMES	50	600	✓	✓	PQ8	76.4	34.7	1.0

Table 2: Impact of AMES components on performance and memory/image (KB).

L_x^{test}	L_q^{test}	$\mathcal{R}$**OP+1M**	**GLDv2**
50	600	75.5	34.3
	50	74.8	34.0
20	600	74.4	33.9
	20	73.6	33.5
10	600	73.9	33.7
	10	73.2	33.4
global-PQ8		72.8	33.3

Table 3: Impact of asymmetry on performance. Experiment with varying number of descriptors L_q^{test} (query) and L_x^{test} (database) during testing. Global similarity performance is shown for reference.

mem.	global	L_x^{test}	$\mathcal{R}$**OP+1M**	**GLDv2**
3KB	PQ1	64	76.9	34.8
	PQ4	160	77.8	35.2
	PQ8	176	78.0	35.3
2KB	PQ1	0	72.8	33.3
	PQ4	96	77.4	35.1
	PQ8	112	77.4	35.2
1KB	PQ4	32	75.9	34.3
	PQ8	48	76.4	34.6

Table 4: Performance of AMES for small memory footprint per image using various global-local descriptor combinations. Global descriptor quantization and the number of local descriptors vary.

Ablation study. Table 2 reports performance and memory for an ablation study. Asymmetric similarity is beneficial, while binarization slightly compromises the performance but provides large benefits in memory requirements. Distillation improves performance without compromising memory. Compressing the global descriptor with PQ8 saves memory with negligible performance loss.

Impact of asymmetry. Table 3 reports the results for symmetric ($L_x^{\text{test}} = L_q^{\text{test}}$) and asymmetric ($L_x^{\text{test}} < L_q^{\text{test}}$) similarity. The number of descriptors varies only on the query side and remains fixed on the database side. Increasing L_q brings a significant performance increase in all cases. Figure 1 shows the importance of local descriptors based on the dot product similarity of the respective tokens and the matching token at the output of the N-th transformer block. In the asymmetric setting, only a few descriptors that correspond to the query landmark are present, to which our model correctly assigns high importance.

Small memory footprints. Table 4 shows results for different combinations of local/global descriptors to achieve a target memory per image. Observe that it is worth compromising the precision of the global descriptor as an exchange for a larger number of additional local descriptors (PQ1/64 is worse than PQ8/176 at 3KB, and PQ4/32 is worse than PQ8/48 at 1KB).

CVNet *vs*. DINOv2 for global and local descriptors. In Figure 5, we show performance for four different combinations of global and local descriptors. CVNet, and consequently SG, are the outcome of task-specific training, while DINOv2 is a foundational model. Observe how SG is noticeably better as

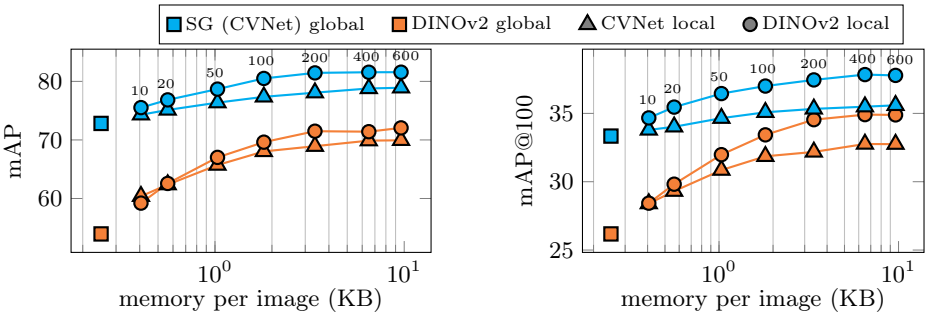

Fig. 5: AMES with different backbones for global and local representation on $\mathcal{R}$OP+1M (left) and GLDv2 (right).

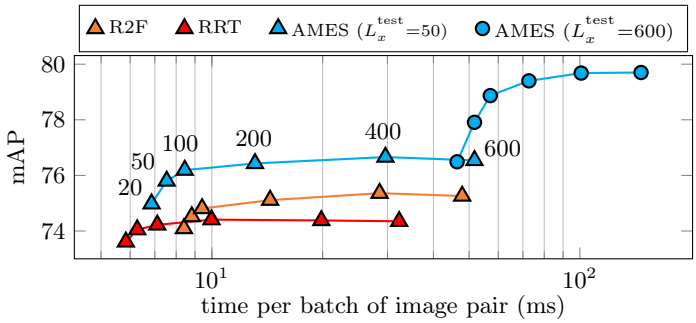

Fig. 6: Performance *vs.* time on $\mathcal{R}$OP+1M for varying L_q^{test} (shown in text labels) and two values of L_x^{test} (R^2Former and RRT use $L_x^{\text{test}} = 50$) using batches of 70 pairs. Global similarity takes 0.03 ms. All three models use standard PyTorch modules.

a global descriptor, while DINOv2 as a local descriptor. Their combination, *i.e.* SG global with DINOv2 local, is the winning entry of Table 1.

Inference speed and GPU memory complexity. Inference time is reported in Fig 6, showing how L_q^{test} and L_x^{test} impact latency and that AMES provides a better trade-off. Regarding GPU memory, the maximum batch size during testing on an 80GB GPU with $L_q^{\text{test}} = 600$ and $L_x^{\text{test}} = 50$ is 6.1k, 4.3k, and 5.2k pairs for RRT, R^2Former, and AMES, respectively.

5 Conclusions

This work highlights the practical importance of keeping the memory requirements in retrieval low and provides an experimental benchmark to allow future comparisons. We propose a new image-to-image similarity model that optimizes the performance *vs.* memory trade-off in the best way compared to existing approaches. The comparisons are performed in a direct and fair way based on the same representation backbone and using the same implementation framework whenever possible. We anticipate local similarity will be a key ingredient for tasks at a scale much larger than the current largest databases.

Acknowledgments: This work was supported by the Junior Star GACR GM 21-28830M and the Czech Technical University in Prague grant No. SGS23/173/OHK3/3T/13.

References

1. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: CVPR (2016)
2. Arandjelović, R., Zisserman, A.: DisLocation: Scalable descriptor distinctiveness for location recognition. In: ACCV (2014)
3. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. In: arXiv (2016)
4. Babenko, A., Lempitsky, V.: The inverted multi-index. In: CVPR (2012)
5. Babenko, A., Lempitsky, V.: Aggregating deep convolutional features for image retrieval. In: ICCV (2015)
6. Beyer, L., Izmailov, P., Kolesnikov, A., Caron, M., Kornblith, S., Zhai, X., Minderer, M., Tschannen, M., Alabdulmohsin, I., Pavetic, F.: Flexivit: One model for all patch sizes. In: CVPR (2023)
7. Budnik, M., Avrithis, Y.: Asymmetric metric learning for knowledge transfer. In: CVPR (2021)
8. Cao, B., Araujo, A., Sim, J.: Unifying deep local and global features for image search. In: ECCV (2020)
9. Chen, S., Wong, S., Chen, L., Tian, Y.: Extending context window of large language models via positional interpolation. In: arXiv (2023)
10. Csurka, G., Dance, C.R., Fan, L., Willamowski, J.K., Bray, C.: Visual categorization with bags of keypoints. In: ECCVW (2004)
11. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)
12. Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F.: Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. PAMI (2012)
13. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: End-to-end learning of deep visual representations for image retrieval. IJCV (2017)
14. Hausler, S., Garg, S., Xu, M., Milford, M., Fischer, T.: Patch-netvlad: Multi-scale fusion of locally-global descriptors for place recognition. In: CVPR (2021)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
16. Iscen, A., Tolias, G., Avrithis, Y., Furon, T., Chum, O.: Efficient diffusion on region manifolds: Recovering small objects with compact CNN representations. In: arXiv (2016)
17. Jain, M., Jégou, H., Gros, P.: Asymmetric hamming embedding: Taking the best of our bits for large scale image search. In: ACM Multimedia (2011)
18. Jegou, H., Douze, M., Schmid, C., Perez, P.: Aggregating local descriptors into a compact image representation. In: CVPR (2010)
19. Jégou, H., Chum, O.: Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In: ECCV (2012)
20. Jégou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: ECCV (2008)
21. Jégou, H., Douze, M., Schmid, C.: On the burstiness of visual elements. In: CVPR (2009)

22. Jégou, H., Douze, M., Schmid, C.: Improving bag-of-features for large scale image search. IJCV (2010)
23. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. PAMI (2011)
24. Jégou, H., Zisserman, A.: Triangulation embedding and democratic aggregation for image search. In: CVPR (2014)
25. Kalantidis, Y., Avrithis, Y.: Locally optimized product quantization for approximate nearest neighbor search. In: CVPR (2014)
26. Kordopatis-Zilos, G., Tzelepis, C., Papadopoulos, S., Kompatsiaris, I., Patras, I.: DnS: Distill-and-select for efficient and accurate video indexing and retrieval. IJCV (2022)
27. Lee, S., Seong, H., Lee, S., Kim, E.: Correlation verification for image retrieval. In: CVPR (2022)
28. Likhomanenko, T., Xu, Q., Synnaeve, G., Collobert, R., Rogozhnikov, A.: CAPE: Encoding relative positions with continuous augmented positional embeddings. In: NIPS (2021)
29. Lindenberger, P., Sarlin, P.E., Pollefeys, M.: Lightglue: Local feature matching at light speed. In: ICCV (2023)
30. Lu, J., Xu, C., Zhang, W., Duan, L.Y., Mei, T.: Sampling wisely: Deep image embedding by top-k precision optimization. In: ICCV (2019)
31. Miech, A., Laptev, I., Sivic, J.: Learnable pooling with context gating for video classification. In: arXiv (2017)
32. Musgrave, K., Belongie, S., Lim, S.N.: A metric learning reality check. In: ECCV (2020)
33. Noh, H., Araujo, A., Sim, J., Weyand, T., Han, B.: Large-scale image retrieval with attentive deep local features. In: ICCV (2017)
34. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: Dinov2: Learning robust visual features without supervision. In: arXiv (2024)
35. Perronnin, F., Liu, Y., Sánchez, J., Poirier, H.: Large-scale image retrieval with compressed Fisher vectors. In: CVPR (2010)
36. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR (2007)
37. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR (2008)
38. Radenović, F., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Revisiting oxford and paris: Large-scale image retrieval benchmarking. In: CVPR (2018)
39. Radenović, F., Jegou, H., Chum, O.: Multiple measurements and joint dimensionality reduction for large scale image search with short vectors. In: ICMR (2015)
40. Radenović, F., Tolias, G., Chum, O.: Fine-tuning cnn image retrieval with no human annotation. PAMI (2019)
41. Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S.: CNN features off-the-shelf: An astounding baseline for recognition. In: CVPRW (2014)
42. Revaud, J., Almazán, J., Rezende, R.S., Souza, C.R.d.: Learning with average precision: Training image retrieval with a listwise loss. In: ICCV (2019)
43. Roth, K., Milbich, T., Sinha, S., Gupta, P., Ommer, B., Cohen, J.P.: Revisiting training strategies and generalization performance in deep metric learning. In: ICML (2020)

44. Ruoss, A., Delétang, G., Genewein, T., Grau-Moya, J., Csordás, R., Bennani, M., Legg, S., Veness, J.: Randomized positional encodings boost length generalization of transformers. In: arXiv (2023)
45. Sánchez, Perronnin, F., Mensink, T., Verbeek, J.: Image classification with the fisher vector: Theory and practice. IJCV (2013)
46. Sarlin, P.E., DeTone, D., Malisiewicz, T., Rabinovich, A.: Superglue: Learning feature matching with graph neural networks. In: CVPR (2020)
47. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: CVPR (2015)
48. Shao, S., Chen, K., Karpur, A., Cui, Q., Araujo, A., Cao, B.: Global features are all you need for image retrieval and reranking. In: ICCV (2023)
49. Simeoni, O., Avrithis, Y., Chum, O.: Local features and visual words emerge in activations. In: CVPR (2019)
50. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV (2003)
51. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: NeurIPS (2016)
52. Song, C.H., Yoon, J., Choi, S., Avrithis, Y.: Boosting vision transformers for image retrieval. In: WACV (2023)
53. Song, C.H., Yoon, J., Hwang, T., Choi, S., Gu, Y.H., Avrithis, Y.: On train-test class overlap and detection for image retrieval. In: CVPR (2024)
54. Song, H.O., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: arXiv (2015)
55. Song, Y., Zhu, R., Yang, M., He, D.: Dalg: Deep attentive local and global modeling for image retrieval. In: arXiv (2022)
56. Suma, P., Tolias, G.: Large-to-small image resolution asymmetry in deep metric learning. In: WACV (2023)
57. Sun, J., Shen, Z., Wang, Y., Bao, H., Zhou, X.: Loftr: Detector-free local feature matching with transformers. In: CVPR (2021)
58. Tan, F., Yuan, J., Ordonez, V.: Instance-level image retrieval using reranking transformers. In: CVPR (2021)
59. Tolias, G., Avrithis, Y., Jégou, H.: Image search with selective match kernels: aggregation across single and multiple images. IJCV (2015)
60. Tolias, G., Jenicek, T., Chum, O.: Learning and aggregating deep local descriptors for instance-level recognition. In: ECCV (2020)
61. Varis, D., Bojar, O.: Sequence length is a domain: Length-based overfitting in transformer models. In: arXiv (2021)
62. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
63. Weinzaepfel, Philippe and Lucas, Thomas and Larlus, Diane and Kalantidis, Yannis: Learning Super-Features for Image Retrieval. In: ICLR (2022)
64. Weyand, T., Araujo, A., Cao, B., Sim, J.: Google landmarks dataset v2 - A large-scale benchmark for instance-level recognition and retrieval. In: CVPR (2020)
65. Wu, C.Y., Manmatha, R., Smola, A.J., Krahenbuhl, P.: Sampling matters in deep embedding learning. In: ICCV (2017)
66. Wu, H., Wang, M., Zhou, W., Li, H., Tian, Q.: Contextual similarity distillation for asymmetric image retrieval. In: CVPR (2022)
67. Wu, H., Wang, M., Zhou, W., Lu, Z., Li, H.: Asymmetric feature fusion for image retrieval. In: CVPR (2023)
68. Wu, Z., Ke, Q., Isard, M., Sun, J.: Bundling features for large scale partial-duplicate web image search. In: CVPR (2009)

69. Yang, M., He, D., Fan, M., Shi, B., Xue, X., Li, F., Ding, E., Huang, J.: Dolg: Single-stage image retrieval with deep orthogonal fusion of local and global features. In: CVPR (2021)
70. Zhu, S., Yang, L., Chen, C., Shah, M., Shen, X., Wang, H.: R2former: Unified retrieval and reranking transformer for place recognition. In: CVPR (2023)