Asymmetric Feature Fusion for Image Retrieval

Hui Wu[1] Min Wang[2*] Wengang Zhou[1,2*] Zhenbo Lu[2] Houqiang Li[1,2]
[1]CAS Key Laboratory of Technology in GIPAS, University of Science and Technology of China
[2]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
wh241300@mail.ustc.edu.cn, {wangmin,luzhenbo}@iai.ustc.edu.cn, {zhwg,lihq}@ustc.edu.cn

1. More Details about Experiment Setup

Testing details. For all testing datasets, images are resized with the larger dimension equal to $1,024$ pixels, which preserve the original aspect ratio. Besides, image feature is extracted at three scales, *i.e.*, $\{1/\sqrt{2}, 1, \sqrt{2}\}$. L_2 normalization is performed for each scale independently and features of three scales are mean averaged, followed by another L_2 normalization to form the final feature. Under the *asymmetric retrieval* setting, queries are embedded with the lightweight query model ϕ_q, while the gallery images are embedded by various large models, which are further aggregated into compact embedding via the proposed mixer ϕ^{mix} for efficient retrieval.

Gallery features. Tab. 1 provides details about the gallery features adopted in this work. In real-world applications, retrieval latency, feature extraction latency and storage overhead are three important considerations.

(1) *Retrieval latency*. Although inverted index greatly speeds up image retrieval based on local features, the online retrieval latency is still larger compared to compact global features, *e.g.*, 0.995s for HOW [8] *vs*. 0.345s for DELG [1], when searching in a gallery of 1 million images. Our approach aggregates local features into compact descriptors at the gallery side, which reduces online retrieval latency while improving asymmetric retrieval accuracy.

(2) *Feature extraction latency*. To achieve scale invariance, existing deep local features are extracted at multiple image scales. An image is scaled to different sizes and passed through feature extractor multiple times, which notably increases the computational complexity, *e.g.*, it takes about 258.1ms to extract local features of an image for HOW [8]. In contrast, compact global feature is derived from the feature maps of the deep representation model by spatial pooling, which is suitable for resource-constrained scenarios. For example, it only takes about 16.5ms to extract a global feature when MobileNetV2 is adopted as feature extractor. Our method just deploys a lightweight model on the query side to extract global features while deploying multiple models on the gallery side for feature fusion. It advances the existing asymmetric retrieval systems without introducing any extra overhead to the query side.

(3) *Storage overhead*. Typically, there are thousands of local features in a single image, which greatly increases the storage overhead of the gallery set. For example, for $\mathcal{R}$1M [6], it takes about 480G to store all the local features

FEATURE	TYPE	BACKBONE	# NUM.	DIM.	EXT. (*ms*)	LAT. (*s*)
DELG [1]	global	ResNet101	1	$2,048$	113.7	0.345
Token [9]	global	ResNet101	1	$1,024$	141.3	0.143
CVNet [5]	global	ResNet101	1	$2,048$	113.7	0.345
DOLG [11]	global	ResNet101	1	512	199.6	0.111
*DELF [1]	local	ResNet101	$\approx 1,000$	128	388.9	1.042
HOW [8]	local	ResNet50	$\approx 2,000$	128	258.1	0.995

Table 1. **Various gallery features**. # NUM.: the number of feature per image; DIM.: feature dimension; EXT.: average latency for feature extraction; LAT: average latency of a single retrieval in a gallery set with 1 Million images. We perform feature extraction and retrieval 100 times and report the average latency. All the results are counted on a machine with a single thread GPU (RTX 2080Ti) and CPU (Intel Xeon CPU E5-2640 v4 @ 2.40GHz).

QUERY NET ϕ_q	GALLERY NET ϕ_g	FLOPs (G)		PARAM. (MB)	
		ABS	‰	ABS	‰
Mixer	Mixer	175.12	1000.0	202.34	1000.0
GhostNet	Mixer	1.36	7.76	4.64	22.9
ShuffleNetV2$^{0.5\times}$	Mixer	0.84	1.96	2.44	5.74
MobileNetV2$^{0.5\times}$	Mixer	1.31	7.48	3.31	16.35
MobileNetV3$^{0.5\times}$	Mixer	0.66	3.76	2.11	10.42
ShuffleNetV2	Mixer	1.44	8.22	3.35	16.56
MobileNetV3	Mixer	1.82	10.39	4.94	24.41
MobileNetV2	Mixer	2.50	14.27	4.85	23.97
EfficientNetB0	Mixer	2.86	16.33	6.63	32.77
EfficientNetB1	Mixer	3.92	9.15	9.13	21.49

Table 2. **FLOPs and model size for lightweight models** used in this work, absolute (ABS) and relative (‰) to the gallery model. All the models are modified slightly to adapt to image retrieval. The FLOPs are calculated with a input image of 362×362.

*Corresponding Author: Min Wang and Wengang Zhou.

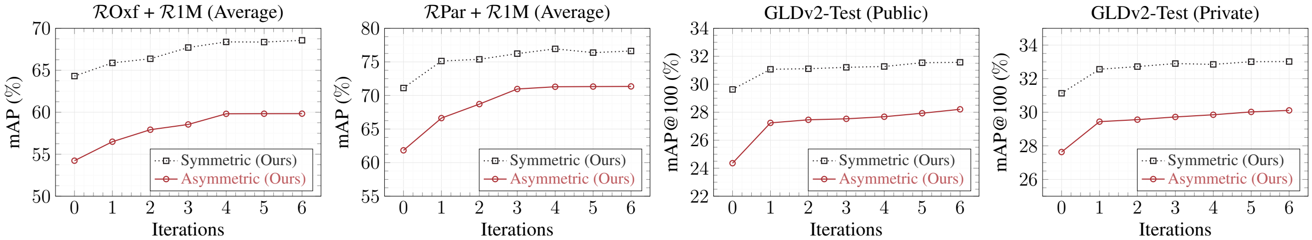

Figure 1. **Performance *vs*. Number of iterations**. Iterations demote the hyper-parameter C in the Eq. (10) of main paper. On the gallery side, our method aggregates DELG [1], Token [9], DOLG [11] and CVNet [5] into compact embedding. *Asymmetric*: MobileNetv2 [4] is deployed on the query side for feature extraction. *Symmetric*: feature fusion is also adopted for the query side, which leads to heavy computational and storage overhead.

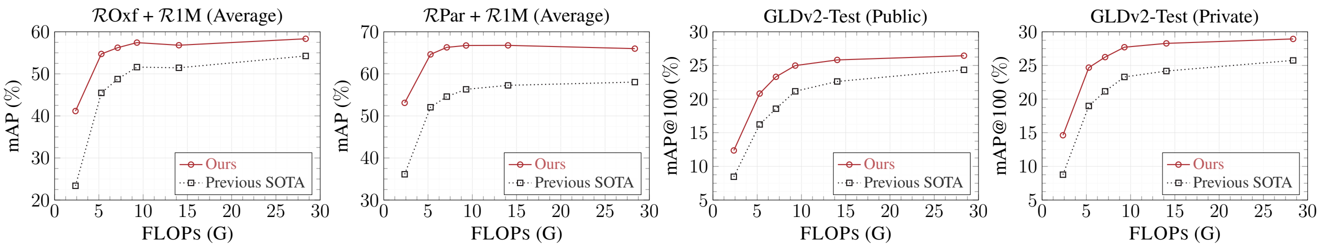

Figure 2. **Performance *vs*. inference FLOPs** when varying the image size. *Ours*: MobileNetV2 [7] is adopted as query model and gallery features are generated via aggregating DELG [1], Token [9], DOLG [11] and CVNet [5] into compact embedding with our proposed mixer. *Previous SOTA*: The latest asymmetric retrieval method CSD [10] is adopted to train query model with just CVNet [5] deployed as gallery model. Query features are extracted as the original single scale. The x-axis represents the average FLOPs (G) for five inferences, which is proportional to input size. We resize the queries to $\{0.4, 0.6, 1/\sqrt{2}, 0.8, 1.0, \sqrt{2}\}$ of the original size (1024×768).

extracted by HOW [8]. Even being binarized, the inverted index still takes up 14.2G of memory. While for the global feature DELG [1], it only needs 7.7G to store all the gallery features. In our method, multiple local and global features are aggregated into compact embedding on the gallery side. It exploits the complementarity of different features to enhance the discriminativeness of gallery features while ensuring a small storage overhead, *e.g.*, 7.7G for $\mathcal{R}$1M.

2. Additional Ablation and Analysis

Impact of C in the Eq. (10) of main paper. The proposed mixer dynamically extracts helpful features from various gallery features via a learnable fusion token. In this section, we explore the impact of different iteration numbers on the retrieval accuracy. As shown in Fig. 1, retrieval accuracy increases gradually as the number of iterations increases under both asymmetric and symmetric settings. The results show that helpful features are progressively aggregated to form discriminative image representations.

Inference computational overhead. In previous experiments, we follow the common setup to scale a test image to different sizes and forward them through feature extractor multiple times to extract multi-scale feature. However, multi-scale feature extraction notably increases the computational overhead for the query side. In this section, we investigate the relationship between retrieval performance and computational overhead. MobileNetV2 is deployed as the query model and single-scale feature extraction is performed. We scale test images to different sizes, which correspond to different computational overhead (FLOPs), to ex-

Query Net	Gallery Feature	GLDv2-Test		$\mathcal{R}$Oxf + 1M		$\mathcal{R}$Par + 1M	
		Private	Public	Medium	Hard	Medium	Hard
MobileNetV3$^{0.5\times}$	*Single*	21.84	19.92	57.84	37.89	66.71	44.76
	Mixed	**26.19**	**23.43**	**62.91**	**42.28**	**68.29**	**46.97**
MobileNetV3	*Single*	27.15	25.10	62.50	43.15	75.93	55.85
	Mixed	**29.34**	**26.58**	**68.71**	**46.87**	**79.85**	**62.42**
ShuffleNetV2$^{0.5\times}$	*Single*	21.84	19.92	52.58	33.24	52.69	33.27
	Mixed	**26.19**	**23.43**	**62.79**	**43.61**	**71.87**	**51.60**
ShuffleNetV2	*Single*	26.02	23.16	60.12	40.05	65.79	45.43
	Mixed	**28.76**	**25.64**	**68.38**	**48.92**	**75.92**	**56.82**
MobileNetV2$^{0.5\times}$	*Single*	23.54	21.13	56.88	37.55	65.68	44.05
	Mixed	**25.79**	**23.50**	**64.41**	**43.53**	**74.89**	**54.97**
MobileNetV2	*Single*	26.02	23.16	60.12	40.05	65.79	45.43
	Mixed	**29.85**	**27.68**	**70.47**	**49.16**	**80.01**	**62.58**
EfficientNetB0	*Single*	27.45	25.32	65.63	42.52	74.69	55.04
	Mixed	**30.96**	**27.30**	**71.72**	**50.21**	**79.96**	**61.85**
EfficientNetB1	*Single*	29.50	26.77	67.03	46.46	76.28	57.50
	Mixed	**31.52**	**27.98**	**72.44**	**51.56**	**81.80**	**63.97**
GhostNet	*Single*	26.50	23.51	61.72	42.12	74.61	55.06
	Mixed	**28.45**	**25.84**	**65.57**	**45.39**	**79.11**	**61.19**

Table 3. Analysis of **different lightweight query model architectures**. **Mixed**: Four global features including DELG [1], Token [9], DOLG [11] and CVNet [5] are aggregated into compact embedding on the gallery side. Query model and mixer are trained jointly. *Single*: CVNet is deployed as gallery model and CSD [10] is adopted to train the query model.

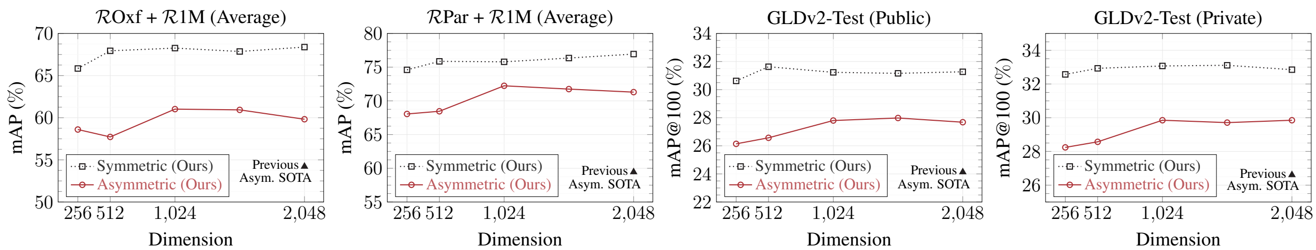

Figure 3. **Performance *vs*. output dimension**. On the gallery side, our method aggregates DELG [1], Token [9], DOLG [11] and CVNet [5] into compact embedding. *Asymmetric*: MobileNetv2 [4] is deployed on the query side for feature extraction. *Symmetric*: Feature fusion is also adopted for the query side. *Previous Asym. SOTA*: The latest asymmetric retrieval method CSD [10] is adopted to train query model (MobileNetV2), whose output dimension is $2,048$, with CVNet [5] deployed as gallery model. The x-axis represents different output dimension, which is associated with the storage of the gallery set.

tract single-scale query features. As shown in Fig. 2, both asymmetric and symmetric retrieval accuracy increases and then saturates as inference computational complexity increases. In some real-world applications, we need to scale an image to an appropriate size for the trade-off between retrieval accuracy and efficiency.

Different lightweight query models. In this section, various lightweight models, with different computational complexity and model size, are deployed as query models ϕ_q. As shown in Tab. 3, when various lightweight models are deployed as query model, our approach consistently advances the accuracy of asymmetric retrieval without introducing any additional overhead to the query side.

Output dimension. Since our method jointly trains the mixer and query model, it is easy to adjust the dimension of image feature. In the real-world retrieval system, there are billions of images and large feature dimension leads to huge storage overhead. In this section, we explore the impact of different output dimension on retrieval accuracy. As shown in Fig. 3, both asymmetric and symmetric retrieval accuracy increases gradually and then saturates as the dimension increases. In practical scenarios, we should choose appropriate feature dimension to achieve the trade-off between storage footprint and retrieval accuracy.

More details about generalizability analysis. In the main paper, our asymmetric feature fusion is combined with various existing asymmetric retrieval methods. Here, we provide more implementation details, when it is combined with CSD [10]. For an image, CSD aims to constrain the query model to maintain its neighborhood structure in the embedding space of the gallery model. It assumes that a well-trained gallery model is adopted to embed the images of a training gallery $\mathcal{G}^T$ into features $\mathbf{G} \in \mathbb{R}^{N\times d}$. These features are then adopted to mine neighbors of each training image during the learning of query model.

However, in our approach, the gallery features keep evolving with training. To approximate the fixed gallery features $\mathbf{G}$ in CSD, we maintain a memory to store features from the immmediate preceding mini-batches, which is denoted as $\widetilde{\mathbf{G}} \in \mathbb{R}^{M\times d}$. The introduction of the memory makes the size of training gallery much larger than that of mini-batch, which provides sufficient neighbors for each training image. Similar to MOCO [3], the features in the memory are progressively replaced, following the first-in-first-out principle. To improve the consistency of features in the memory, we also follow MOCO to maintain a momentum-updated version of mixer ($\widetilde{\phi}^{\mathrm{mix}}$) to aggregate multiple gallery features into more smooth embedding $\widetilde{\boldsymbol{g}}^{\mathrm{mix}}$, which is stored in the memory. During query model training, for each training image, we mine its neighbors just in the memory with the embedding generated by $\widetilde{\phi}^{\mathrm{mix}}$. The final objective function consists of two losses. One is ArcFace loss [2] (Eq. (11) in the main paper) for training mixer, the other is contextual similarity consistency loss defined in CSD, with the modifications mentioned above.

References

[1] Bingyi Cao, André Araujo, and Jack Sim. Unifying deep local and global features for image search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 726–743, 2020. 1, 2, 3

[2] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[3] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, 2020. 3

[4] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 2, 3

[5] Seongwon Lee, Hongje Seong, Suhyeon Lee, and Euntai Kim. Correlation verification for image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5374–5384, 2022. 1, 2, 3

[6] Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5706–5715, 2018. 1

[7] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[8] Giorgos Tolias, Tomas Jenicek, and Ondřej Chum. Learning and aggregating deep local descriptors for instance-level recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 460–477, 2020. 1, 2

[9] Hui Wu, Min Wang, Wengang Zhou, Yang Hu, and Houqiang Li. Learning token-based representation for image retrieval. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 2703–2711, 2022. 1, 2, 3

[10] Hui Wu, Min Wang, Wengang Zhou, Houqiang Li, and Qi Tian. Contextual similarity distillation for asymmetric image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9489–9498, 2022. 2, 3

[11] Min Yang, Dongliang He, Miao Fan, Baorong Shi, Xuetong Xue, Fu Li, Errui Ding, and Jizhou Huang. Dolg: Single-stage image retrieval with deep orthogonal fusion of local and global features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 11772–11781, 2021. 1, 2, 3