Selective Deep Convolutional Features for Image Retrieval: Reproducibility Companion Paper

Tuan Hoang
Singapore University of Technology and Design
nguyenanhtuan_hoang@mymail.sutd.edu.sg

Thanh-Toan Do
The University of Liverpool
thanh-toan.do@liverpool.edu.uk

Ngai-Man Cheung
Singapore University of Technology and Design
ngaiman_cheung@sutd.edu.sg

ABSTRACT

In this companion paper, firstly, we briefly summarize the contributions of our main manuscript: Selective Deep Convolutional Features for Image Retrieval, published in ACM MultiMedia 2017 [7]. In addition, we provide detail instructions together with pre-configured MATLAB scripts which allows experiments to be executed and to reproduce the results reported in our main manuscript effortlessly. The source code is available at https://github.com/hnanhtuan/selectiveConvFeatures_ACMMM_reproducibility.

1 CONTRIBUTION SUMMARY

The two main issues SIFT-based retrieval methods are the lack of discriminability and the strong effect of *burstiness* [8], i.e. numerous descriptors are almost similar within the same image, which can considerably degrade the quality of SIFT-based image representations for the image retrieval task [4, 8, 10]. While the discriminability issue can be well handled by using conv. features [1, 2, 12, 13, 21] . The *burstiness* effect has not been investigated for the conv. features.

To address this issue, inspired by the fact that convolutional (conv.) feature maps preserve certain levels of spatial information, we firstly propose mask schemes which select local conv. features of salient regions while ignoring background regions. In specific, we proposed various novel masking schemes **SIFT-mask**, **SUM-mask**, **MAX-mask** to select a representative subset of local conv. features and remove a large number of redundant features. In addition, we proposed to leverage the state-of-the-art embedding methods (e.g., Fisher Vector (FV) [15], VLAD [10], Triangular Embedding (T-emb) [11], and F-FAembd [5]), and aggregating methods (e.g., Democratic pooling [11]) in combination with our novel masking schemes to obtain an effective image retrieval framework. The overview of our proposed framework is shown in Figure 1. Please refer to the main manuscript for the details of the proposed framework. We summarize the notations in Table 1.

Table 1: Notations and their corresponding meanings. $\mathcal{M}, \phi, \psi$ denote masking, pooling and embedding respectively.

Notation	Meaning
$\mathcal{M}_{\text{SIFT}}$	SIFT-mask
$\mathcal{M}_{\text{SUM}}$	SUM-mask
$\mathcal{M}_{\text{MAX}}$	MAX-mask
ψ_{a}	Average-pooling
ψ_{s}	Sum-pooling
ψ_{d}	Democratic-pooling [11]
ϕ_{FV}	FV [15]
ϕ_{VLAD}	VLAD [10]
ϕ_{Δ}	T-emb [11]
$\phi_{\text{F-FAemb}}$	F-FAemb [5]
d	PCA retained dimension
$\vert C\vert$	Codebook size of embedding methods
D	Global representation dimension

2 EXPERIMENTS

2.1 Datasets and Evaluation protocols

Oxford Buildings dataset[1]: The *Oxford5k* dataset [17] consists of 5,063 images of buildings and 55 query images corresponding to 11 distinct buildings in Oxford. Each query image contains a bounding box indicating the region of interest. Following the standard practice [5, 6, 11, 21], we use the cropped query images based on provided bounding boxes.

Paris dataset[2]: The *Paris6k* dataset [18] consists of 6412 images of famous landmarks in Paris. Similar to *Oxford5k*, this dataset has 55 queries corresponding to 11 landmarks. We also use provided bounding boxes to crop the query images accordingly.

INRIA Holidays dataset[3]: The *Holidays* dataset [9] contains 1,491 images corresponding to 500 scenes. The query image set consists of one image from each scene. Following [2, 3, 12], we manually rotate images (by ±90 degrees) to fix the incorrect image orientation.

Oxford105k and Paris106k datasets: We additionally combine *Oxford5k* and *Paris6k* with 100k Flickr images [17][4] to form larger databases, named *Oxford105k* and *Paris106k* respectively. The new databases are used to evaluate retrieval performance at a larger scale.

[1] http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/

[2] http://www.robots.ox.ac.uk/~vgg/data/parisbuildings/

[3] http://lear.inrialpes.fr/people/jegou/data.php#holidays

[4] http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/flickr100k.html

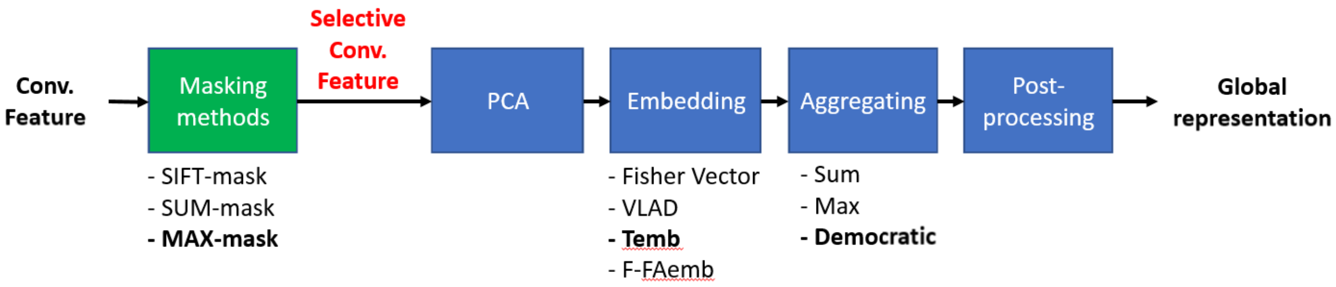

Figure 1: The overview of the proposed framework.

All the datasets can be downloaded using the bash script: *extract_feature_map/download_datasets.sh*. Please note that the bash script does not delete the downloaded packages. Please handle this case manually if the storage capacity is limited.

Dataset	# images	Download Size (GB)
Oxford5k	5,063	2
Paris6k	6,412	2.6
Holidays	1,491	2.8
Flickr100k	100,071	39.6

Table 2: Dataset summary.

Evaluation protocols. The retrieval performance is reported as mean average precision (mAP) over query sets for all datasets. In addition, the junk images, which are defined as unclear to be relevant or not, are removed from the ranking.

Furthermore, in the image retrieval task, it is important to use held-out datasets to learn all necessary parameters as to avoid overfitting [2, 6, 19]. In particular, the set of 5000 Flickr images[5] is used as the held-out dataset to learn parameters for *Holidays*. Similarly, *Oxford5k* is used for *Paris6k* and *Paris106k*, and *Paris6k* for *Oxford5k* and *Oxford105k*.

2.2 Flatform, softwares, and dependencies

All the experiments are conducted in Ubuntu 16.04, with 64GB memory and a NVIDIA 1080 Ti GPU (CUDA toolkit 9.0). We use the MatConvet (version 1.0-beta25) and Vlfeat (version 0.9.21) on MATLAB 2016a. The MatConvnet framework is used to extract the convolutional features of a pretrained CNN model for high-resolution images. Hence, a decent GPU (with at least 8GB memory) is needed to facilitate the process. Additionally, about 250 GB storage is required for the datasets (≈ 47 GB), extracted conv. features (≈ 153 GB), and learned parameters (≈ 40 GB).

We include the MATLAB script: *tools/download_compile_packages.m* to download and compile the MatConvnet (with GPU enabled) and Vlfeat toolkits. Please follow the instruction provided in the script to ensure proper installation. In addition, please execute the MATLAB script: *utils/make.m* to compile the MATLAB **mex** files for functions using for F-FAemb and Temb methods.

Table 3: Configurations of different embedding methods.

Methods	d	$\|C\|$	D
FV [15]	48	44	$2 \times d \times \|C\| = 4224$
VLAD [10]	64	66	$d \times \|C\| = 4224$
T-emb [11]	64	68	$d \times \|C\| - 128 = 4224$
F-FAemb [5]	32	10	$\frac{(\|C\| - 2) \times d \times (d+1)}{2} = 4224$

2.3 Experiment 1 - Framework analysis

To obtain the optimal framework, we first conduct experiment to comprehensively compare various embedding and aggregating frameworks in combination with different proposed masking schemes. To make a fair comparison, we empirically set the retained PCA components-d and size of the visual codebooks-$|C|$ so as to produce the same final feature dimensionality-D as mentioned in Table 3. In this experiment, we extract conv. features from the last conv. layer of the pretrained VGG16 network [20], i.e., conv5-3, with input images are resized such that the largest dimension is 1024 while preserving the aspect ratio.

Execution procedure:

(1) Extract conv. features and locations of SIFT features by using the MATLAB script:
extract_feature_map/extract_feature_VGG16_main.m.
Please note that the extracted features from this MATLAB script consume about 21 GB of storage.

(2) Execute the experiment using the MATLAB script:
exp_table3_run_all.m.
The output table (corresponding to Table 3 - Section 5.2.1 in main manuscript) is printed out in the MATLAB command window.

Expected outcomes:

- The framework $\phi_{\Delta} + \psi_d$ and $\phi_{\text{F-FAemb}} + \psi_d$ generally achieves comparable performances across various masking schemes and datasets. These two frameworks also outperform $\phi_{\text{FV}} + \psi_a$ and $\phi_{\text{VLAD}} + \psi_s$.
- The proposed masking schemes generally help to improve performance in comparison with using no mask.
- The MAX-mask achieves the best performances in majority of settings.

[5] We randomly select 5000 images from the 100.071 Flickr image set [17]. This could results in larger differences in reproduced and reported results.

2.4 Experiment 2 - Impact of Power-law normalization (PN)

The burstiness of visual elements [8] is known as a major drawback of hand-crafted local descriptors, e.g. SIFT [14], such that numerous descriptors are almost similar within the same image. As a result, this phenomenon strongly affects the measure of similarity between two images. By applying power-law normalization [16] (with the power factor of 0.5) to the final image representation ψ and subsequently l2-normalization, it has been shown to be an effcient way to reduce the effect of burstiness [8]. The power-law normalization formula is given as $PN(x) = \text{sign}(x)|x^{\alpha}|$, where $0 \leq \alpha \leq 1$ is a constant [16].

We conduct experiment to evaluate the burstiness effect on conv. features in comparison with the effect on SIFT feature using Oxford5k and Holidays datasets. Additionally, the experiment also demonstrates that the proposed masking schemes can further reduce the burstiness effects. The stronger effect of the power-law normalization on the retrieval performance, i.e., the larger change in performance as α varies, indicates the more severe effects of burstiness on the final image representation.

Execution procedure:

(1) Execute the experiment using the MATLAB script:
exp_figure3_run_all.m.
The output figures (corresponding to Figure 3 - Section 4.4 in main manuscript) are exported in PDF format and saved in *results/charts/exp_figure3* directory.

Expected outcomes:

- The graphs show that the performance when using conv. features does not significantly change as α varies as observed in the case of SIFT features.
- When using SIFT/SUM/MAX-mask, the performance is even more stable to the change of α.

2.5 Experiment 3 - Impact of final representation dimensionality

As the representation dimensionality can significantly affect the retrieval performance, we investigate the performance of our proposed framework with various masking schemes at different dimensionalities using Oxford5k and Paris6k datasets. We control the final representation dimensionalities D by empirically set the number of retained PCA components d and the codebook size $|C|$ as shown in Table 4.

Table 4: Number of retained PCA components and anchor points when varying the dimensionality.

Dim. D	512-D	1024-D	2048-D	4096-D	8064-D
PCA d	32	64	64	64	128
$\lvert C\rvert$	20	18	34	66	64

Execution procedure:

(1) Execute the experiment using the MATLAB script:
exp_figure4_run_all.m.
The output figures (corresponding to Figure 4 - Section 5.2.2 in main manuscript) are exported in PDF format and saved in *results/charts/exp_figure4* directory.

Expected outcomes:

- Higher performance is achieved as higher dimensionality of the final representation is used.
- The proposed masking schemes, i.e., SIFT/SUM/MAX-mask consistently help to gain extra performance across different dimensionalities in comparison with using no mask.

2.6 Experiment 4 - Impact of input image size

Since our method takes a set of local conv. features as input, it is important to evaluate our method with a smaller image size, as the image size can significantly affect the number of local conv. features and their quality. We conduct experiments with input images are resized such that the maximum dimension is 1024 and 724 while preserving aspect ratios. We investigate this aspect using Oxford5k and Paris6k datasets with SUM and MAX-maskes.

Execution procedure:

(1) Extract the additional conv. features by using the MATLAB script:
extract_feature_map/extract_feature_VGG16_exp_table5.m.
Specifically, we extract conv. features (of the last conv. layer of the pretrained VGG16 network [20]) for *Oxford5k* and *Paris6k* datasets when the input images are resized so that the maximum dimension is 724 while preserving aspect ratios.

(2) Execute the experiment using the MATLAB script:
exp_table5_run_all.m.
The output table (corresponding to Table 5 - Section 5.2.3 in main manuscript) is printed out in the MATLAB command window.

Expected outcomes:

- Generally, as the smaller input images are used, smaller retrieval performance is achieved. This is understandable as with bigger images, the CNN can take a closer *"look"* on smaller details in the images.

2.7 Experiment 5 - Impact of conv. layer

In this experiment, we investigate the impact of using the conv. features from different conv. layers, including conv5-3, conv5-2, conv5-1, conv4-3, conv4-2, and conv4-1, on the retrieval performance.

Execution procedure:

(1) Extract the additional conv. features by using the MATLAB script:
extract_feature_map/extract_feature_VGG16_exp_figure5.m.
Specifically, we extract conv. features of the different conv. layers, including conv5-3, conv5-2, conv5-1, conv4-3, conv4-2, and conv4-1, of the pretrained VGG16 network [20]). Please note that the additional conv. features required for this experiment consume approximately 100 GB of storage.

(2) Execute the experiment using the MATLAB script:
exp_figure5_run_all.m.
The output figures (corresponding to Figure 4 - Section 5.2.2 in main manuscript) are exported in PDF format and saved in *results/charts/exp_figure5* directory.

Expected outcomes:

- The conv. features of deeper conv. layer result in better performance.

2.8 Experiment 6 - Comparison with state of the art

We conduct experiments to compare our proposed framework with state-of-the-art retrieval methods on 5 standard benchmark datasets: *Oxford5k*, *Oxford105k*, *Paris6k*, *Paris106k*, and *Holidays*. In this experiment, we use the conv. features extracted from (i) off-the-shelf VGG16 network [20], which is trained on ImageNet dataset for classification task, and (ii) the VGG16-based siaMAC network [19], which is fine-tuned for image retrieval task.

Execution procedure:

(1) Extract the additional conv. features by using the MATLAB script:
extract_feature_map/extract_feature_siaMAC_exp_table6.m.
Specifically, we extract conv. features of the last conv. layer of the VGG16-based siaMAC network [19], which is finetuned for the retrieval task. The input images are resized so that the maximum dimension is 1024 while preserving aspect ratios. Please note that the additional features from this MATLAB script consume about 32 GB of storage.

(2) Execute the experiment using the MATLAB script:
exp_table6_part1.m.
exp_table6_part2.m.
The output table (corresponding to Table 6 - Section 5.3 in main manuscript) is printed out in the MATLAB command window.

Expected outcomes:

- Our proposed framework achieves comparable performance with or outperform state-of-the-art methods with the same dimensionality.
- When using the conv. features from the fine-tuned network, i.e., siaMAC [19], our framework achieves considerable gains in performance and outperform compared methods.

REFERENCES

[1] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. 2016. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*.

[2] Artem Babenko and Victor Lempitsky. 2015. Aggregating Local Deep Features for Image Retrieval. In *ICCV*.

[3] Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky. 2014. Neural codes for image retrieval. In *ECCV*.

[4] Jonathan Delhumeau, Philippe-Henri Gosselin, Hervé Jégou, and Patrick Pérez. 2013. Revisiting the VLAD image representation. In *ACM MM*.

[5] Thanh-Toan Do and Ngai-Man Cheung. 2017. Embedding based on function approximation for large scale image search. *TPAMI* (2017).

[6] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. 2016. Deep Image Retrieval: Learning Global Representations for Image Search. In *ECCV*.

[7] Tuan Hoang, Thanh-Toan Do, Dang-Khoa Le Tan, and Ngai-Man Cheung. 2017. Selective Deep Convolutional Features for Image Retrieval. In *ACM Multimedia 2017*.

[8] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. 2009. On the burstiness of visual elements. In *CVPR*.

[9] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. 2010. Improving Bag-of-Features for Large Scale Image Search. *IJCV* 87, 3 (May 2010), 316–336.

[10] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. 2010. Aggregating local descriptors into a compact image representation. In *CVPR*.

[11] Hervé Jégou and Andrew Zisserman. 2014. Triangulation embedding and democratic aggregation for image search. In *CVPR*.

[12] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. 2016. Cross-dimensional Weighting for Aggregated Deep Convolutional Features. In *ECCV Workshops*.

[13] Ying Li, Xiangwei Kong, Liang Zheng, and Qi Tian. 2016. Exploiting Hierarchical Activations of Neural Network for Image Retrieval. In *ACM MM*.

[14] David G. Lowe. 1999. Object Recognition from Local Scale-Invariant Features. In *ICCV*.

[15] Florent Perronnin and Christopher Dance. 2007. Fisher Kernels on Visual Vocabularies for Image Categorization. In *CVPR*.

[16] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. 2010. Improving the fisher kernel for large-scale image classification. In *ECCV*.

[17] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. 2007. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*.

[18] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. 2008. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*.

[19] Filip Radenović, Giorgos Tolias, and Ondřej Chum. 2016. CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples. In *ECCV*.

[20] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[21] Giorgos Tolias, Ronan Sicre, and Hervé Jégou. 2016. Particular object retrieval with integral max-pooling of CNN activations. In *ICLR*.