Efficient Object Instance Search Using Fuzzy Objects Matching

Tan Yu,[1] Yuwei Wu,[1,2] Sreyasee Bhattacharjee,[1] Junsong Yuan[1]
[1]Rapid Rich Object Search Lab, Interdisciplinary Graduate School,
Nanyang Technological Univeristy, Singapore, 637553
[2]Beijing Laboratory of Intelligent Information Technology, School of Computer Science,
Beijing Institute of Technology, Beijing, 100081
{tyu008, dbhattacharjee, jsyuan}@ntu.edu.sg, wuyuwei@bit.edu.cn

Abstract

Recently, global features aggregated from local convolutional features of the convolutional neural network have shown to be much more effective in comparison with hand-crafted features for image retrieval. However, the global feature might not effectively capture the relevance between the query object and reference images in the object instance search task, especially when the query object is relatively small and there exist multiple types of objects in reference images. Moreover, the object instance search requires to localize the object in the reference image, which may not be achieved through global representations. In this paper, we propose a Fuzzy Objects Matching (FOM) framework to effectively and efficiently capture the relevance between the query object and reference images in the dataset. In the proposed FOM scheme, object proposals are utilized to detect the potential regions of the query object in reference images. To achieve high search efficiency, we factorize the feature matrix of all the object proposals from one reference image into the product of a set of fuzzy objects and sparse codes. In addition, we refine the feature of the generated fuzzy objects according to its neighborhood in the feature space to generate more robust representation. The experimental results demonstrate that the proposed FOM framework significantly outperforms the state-of-the-art methods in precision with less memory and computational cost on three public datasets.

The task of object instance search, is to retrieve all the images containing a specific object query and localize the query object in the reference images. It has received a sustained attention over the last decade, leading to many object instance search systems (Meng et al. 2010; Jiang, Meng, and Yuan 2012; Jiang et al. 2015; Tolias, Avrithis, and Jégou 2013; Tao et al. 2014; Razavian et al. 2014a; 2014b; Tolias, Sicre, and Jégou 2016; Meng et al. 2016; Bhattacharjee et al. 2016b; 2016a; Mohedano et al. 2016; Cao et al. 2016; Wu et al. 2016). Since the query object only occupies a small part of an image, the global representation may not be effective to capture the relevance between the query object with reference image. Therefore, the relevance between the query object and one reference image is not determined by the overall similarity between the query and the reference image.

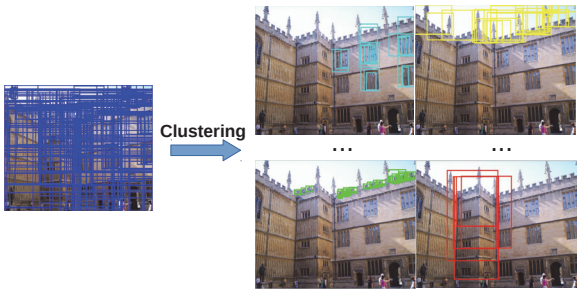


Figure 1: Hundreds of object proposals tend to overlap with each other. Through clustering, the object proposals containing the similar objects will be assigned to the same group.

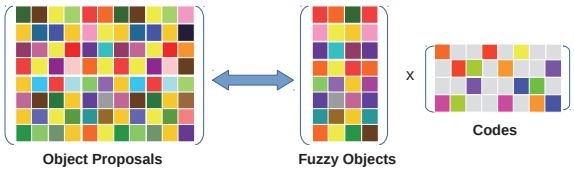


Figure 2: Hundreds of object proposals can be approximated by a compact set of fuzzy objects product the corresponding sparse codes.

Inspired by the success of the object proposal scheme (Zitnick and Dollár 2014) in object detection, we utilize object proposals as potential object candidates for object instance search. A few hundreds of object proposals with highest objectness scores are selected for each image. In this case, the relevance between the query object and a reference image is determined by the best matched object proposal. At the same time, the best-matched object proposal is then identified as the detected location of the query object in the reference image. However, exploring the benefits of hundreds of proposals for each reference image demands the storage of all object proposals and high computational cost to match them. It may be unaffordable when the dataset is large.

An important fact to note that, the object proposals from the same reference image can overlap with each other and thus produce much redundancy among features representing these object proposals. Since the object proposals contain-

ing the similar objects are close to each other in the feature space, through clustering, similar object proposals can be assigned to the same cluster, as shown in Figure 1.

Based on above the observations, we utilize k-mediods clustering (Park and Jun 2009) to generate a small set of fuzzy objects which are treated as the atoms of an image-specific dictionary. Thereafter the features of all object proposals generated from the reference image are encoded into a set of sparse locality-constrained linear codes, as illustrated in Figure 2. In this scenario, the similarity scores of the object proposals can be computed through multiplying the similarity scores of the fuzzy objects by the corresponding sparse codes. We define Fuzzy Objects Matching (FOM) as the above process. Benefiting from the sparsity of the linear codes, the FOM scheme requires much less memory and computational cost than exhaustively comparing the query object with all object proposals while achieving comparable object instance search precision. For example, given n object proposals with d-dimensional features from the reference image, exhaustive search in all object proposals of the reference image demands $\mathcal{O}(nd)$ time complexity. In contrast, the time complexity of the proposed FOM is only $\mathcal{O}(td + nz)$, where $t \sim 0.1n$ is the number of fuzzy objects and $z \sim 0.01d$ is the number of non-zero elements in the sparse code of each object proposal. Thereby the computational cost of the FOM is significantly reduced. Given the fact that any standard object proposal extraction scheme generates hundreds of object proposals for each image and current visual search tasks usually need to deal with a large amount of images in a database, such a computational reduction holds a big promise.

In order to further boost the performance of our FOM, we refine the feature of fuzzy objects via their neighborhoods in the feature space. After neighborhood refinement, the representation of the fuzzy object fusing the information from other relevant images will be much more robust. The entire process of neighborhood refinement being completely offline does not affect the time cost in search, while providing a significant improvement in search accuracy. Impressive results are achieved in comprehensive experiments which significantly outperform the state-of-the-art methods.

Related Work

In (Mopuri and Babu 2015), the authors max-pooled all the proposal-level deep features from the reference image into a global feature. However, the global feature may be distracted when the query object only occupies a small area in the reference image and there exist dense clutters around it. (Razavian et al. 2014a; 2014b) cropped both query image and the reference image into k patches. In their framework, the similarity between the query and the reference image is computed by the average similarity of best matched pairs of patches. The patch-based cross-matching requires k^2 times comparisons which are relatively huge (*e.g.*, k^2 = 1024). Besides, it requires k times feature extraction for k patches of the query online, resulting in computationally demanding. In addition, it can not support object localization.

In (Tolias, Sicre, and Jégou 2016; Mohedano et al. 2016), the authors conducted spatial search by uniformly sampling overlapped candidate regions in different scales in reference images. Tens of candidate regions are selected to compare with the query object and the best-matched region of the reference image determines the relevance of the reference image with the query object. However, tens of sampled regions are not enough to capture the small object in the reference image. In (Cao et al. 2016), the query-adaptive matching kernel was proposed to match query with a set of base image regions. Nevertheless, the query-adaptive matching kernel requires to solve a quadratic programming problem which is quite computationally demanding and it is not applicable for the object localization in the reference image.

In contrast, the proposed FOM scheme is efficient as the set of fuzzy objects is of relatively small scale and the codes matrix is sparse. In the proposed FOM scheme, the relevance of the reference image is determined by the estimated similarity score of the estimated best-matched object proposal. The location of the estimated best-matched object proposal will be further utilized to localize the query object in the reference image.

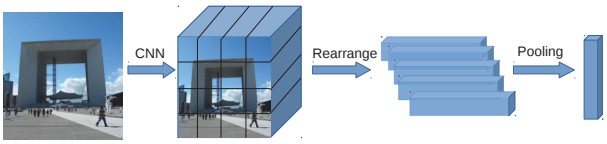


Figure 3: Local convolutional features of each proposal are extracted from the last convolutional layer of CNN and they are further pooled or aggregated into a global feature.

Fuzzy Objects Encoding and Matching

Given a set of reference images, the ultimate task of object instance search is to identify a subset of reference images containing the similar instances of objects to the query. This also involves localizing the object's region within the underlying reference image up to the bounding box. In order to achieve this goal, we design an effective representation scheme which simultaneously considers the effectiveness and efficiency of the matching phase. We then discuss the proposed fuzzy objects encoding and matching in details.

Fuzzy Objects Encoding

Given an image I, a set of object proposals $\mathcal{P} = \{p_i\}_{i=1}^{n}$ are generated by Edge Boxes (Zitnick and Dollár 2014). The extracted object proposals will represent the potential object candidates in the reference image.

The flow of feature extraction for each object proposal p_i is illustrated in Figure 3. Particularly, after being resized into a fixed scale, each object proposal p_i will be fed into the convolutional neural network (CNN) (Simonyan and Zisserman 2014) and a tensor $\mathbf{T}_i \in \mathbb{R}^{w \times h \times d}$ will be obtained from last convolutional layer of CNN. The tensor $\mathbf{T}_i$ is further split into a set of local features $\mathcal{T}_i = \{\mathbf{t}_{im} \in \mathbb{R}^d\}_{m=1}^{wh}$. Finally, $\mathbf{p}_i$, the global deep feature for p_i is generated by pooling or aggregating the local convolutional features in $\mathcal{T}_i$. In this paper, four types of aggregation or pooling methods: max-pooling, sum-pooling, bilinear-pooling (Gao et al.

2016) and vector of locally aggregated descriptors (VLAD) (Arandjelovic and Zisserman 2013) are implemented for local convolutional features pooling/aggregation to generate the object proposal representation and their performance are evaluated respectively in the experiment section.

We denote by $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_n] \in \mathbb{R}^{d \times n}$ the ℓ_2-normalized d-dimensional features of n object proposals from the reference image. Given a query represented by ℓ_2-normalized feature $\mathbf{q}$, the similarity scores of all the object proposals from the reference image can be computed by

$$\mathbf{s} = \mathbf{q}^\top \mathbf{P}. \tag{1}$$

In order to efficiently obtain the similarity scores for the object proposals, we group the features of all the proposals $\{\mathbf{f}_i\}_{i=1}^n$ from the same image into t clusters $\{\mathcal{C}_l\}_{l=1}^t$ by k-mediods clustering (Park and Jun 2009), which achieves better performance than k-means clustering in our experiments. Given the ℓ_2-normalized centroid of cluster $\mathcal{C}_l$, a fuzzy object $\mathbf{o}_l$ is obtained by

$$\begin{aligned} \mathbf{c}_l &= \frac{1}{|\mathcal{C}_l|} \sum_{\mathbf{f} \in \mathcal{C}_l} \mathbf{f}, \\ \mathbf{o}_l &= \mathbf{c}_l / \|\mathbf{c}_l\|_2. \end{aligned} \tag{2}$$

We treat the set of fuzzy objects $\mathbf{O} = [\mathbf{o}_1, ..., \mathbf{o}_t] \in \mathbb{R}^{d \times t}$ as a dictionary and further learn a matrix consisting of sparse codes $\mathbf{H} = [\mathbf{h}_1, ..., \mathbf{h}_n] \in \mathbb{R}^{t \times n}$ such that

$$\mathbf{P} \approx \mathbf{O}\mathbf{H}. \tag{3}$$

While this process is similar to (Iscen, Rabbat, and Furon 2016) in spirit, the fundamental difference is that the proposed encoding process being at the image level instead of the dataset level, offers a more compact object description scheme. The image-specific fuzzy objects encoding scheme is designed to extract a sparse yet discriminative object proposal representation. It can successfully address the problem of handling a huge number of object proposals without loss of search precision, as will be shown in experiments.

Different from sparse coding used in (Iscen, Rabbat, and Furon 2016) which learns the dictionary by solving the ℓ_1-norm penalty optimization problem, the fuzzy objects are simply learned by k-mediods clustering. At the same time, we reconstruct the feature of the object proposal using only the fuzzy objects which are close to the object proposal in the feature space. This process is closely related to Locality-constrain Linear Coding (LLC) (Wang et al. 2010). However, LLC is employed to extract features for classification. In contrast, we use it to achieve efficient object instance search. Given the feature of the object proposal $\mathbf{p}_i$, we use its z nearest neighbours in fuzzy objects set $\mathbf{O}$ as the local bases $\mathbf{O}_i$ and obtain the codes by solving the linear system

$$\min_{\mathbf{h}_i \in \mathbb{R}^z} \|\mathbf{p}_i - \mathbf{h}_i \mathbf{O}_i\|_2 \ \ s.t., \ \mathbf{1}^\top \mathbf{h}_i = 1. \tag{4}$$

The solution of Eq. (4) can be derived analytically by

$$\begin{aligned} \hat{\mathbf{h}}_i &= (\mathbf{O}_i - \mathbf{p}_i \mathbf{1}^\top)(\mathbf{O}_i - \mathbf{p}_i \mathbf{1}^\top)^\top \backslash \mathbf{1}, \\ \mathbf{h}_i &= \hat{\mathbf{h}}_i / (\mathbf{1}^\top \hat{\mathbf{h}}_i). \end{aligned} \tag{5}$$

Fuzzy Objects Matching

After obtaining the fuzzy object sets $\mathbf{O}$ and sparse codes $\mathbf{H}$, the estimated similarity scores between the query object and all the object proposals from the reference image is computed by

$$\tilde{\mathbf{s}} = \mathbf{q}^\top \mathbf{O}\mathbf{H}. \tag{6}$$

The relevant between the query object and the reference image is then determined by the maximum item in the estimated similarity scores vector $\tilde{\mathbf{s}}$:

$$R(q, I) = \max_{l=1,...,n} \tilde{\mathbf{s}}(l). \tag{7}$$

The reference images in the dataset will further be ranked according to their relevance scores.

The complexity of computing $\mathbf{q}^\top \mathbf{O}\mathbf{H}$ is only $\mathcal{O}(dt + zn)$, where z is the number of non-zero elements in each sparse code. Due to the information redundancy observed in the generated proposals from an image, the number of clusters t is chosen to be much smaller compared with the total number of proposals n. Typically $t \sim 0.1n$ works excellent in our experiments. In addition, the codes are very sparse and the typical choices of z is less than $0.01d$. This yields a significant reduction in computation and memory cost. We will show that the object instance search precision obtained by the FOM is comparable with exhaustive matching the query object with all the object proposals but with much less memory and computational cost.

Fuzzy Objects Refinement

In order to further improve the accuracy of object instance search, we refine the features of all the fuzzy objects of all the reference images according to their neighborhood information. In our neighborhood refinement scheme, every fuzzy object $\mathbf{o}_l$ is treated as a pseudo query and m most similar fuzzy objects $\{\mathbf{o}_l^k\}_{k=1}^m$ in the dataset are retrieved to refine the feature of the fuzzy object. The refined fuzzy object $\tilde{\mathbf{o}}_l$ is computed by

$$\tilde{\mathbf{o}}_l = \frac{(\mathbf{o}_l + \sum_{k=1}^m \mathbf{o}_l^k)}{m+1}. \tag{8}$$

The proposed neighborhood refinement follows the spirit of the average query expansion (Chum et al. 2007). However, the average query expansion only refines the feature of the query which is conducted online when the query comes. In contrast, the proposed neighborhood refinement scheme is to refine the features of the fuzzy objects which can be carried out offline and does not effect the search time. It is worth noting that our neighborhood refinement can work in parallel with average query expansion to further boost object instance search precision. In the experiment section, we will evaluate the influence of neighborhood refinement and average query expansion, respectively.

In fact, $\mathbf{o}_l^k$ can be generated from different reference images in the dataset. As $\tilde{\mathbf{o}}_l$ generated from Eq.(8) fuses the information from different reference images, we term $\tilde{\mathbf{o}}_l$ as **inter-image fuzzy object**. In contrast, the fuzzy object $\mathbf{o}_l$ in Eq.(2) are generated from fusing the features of proposals from the same reference image, therefore, we term $\mathbf{o}_l$

as **intra-image fuzzy object**. After $\tilde{\mathbf{o}}_l$ is generated, the approximated similarity between query and all proposals will be computed by

$$\tilde{\mathbf{s}} = \mathbf{q}^\top \tilde{\mathbf{O}}\mathbf{H}, \tag{9}$$

where $\tilde{\mathbf{O}} = [\tilde{\mathbf{o}}_1, ..., \tilde{\mathbf{o}}_t]$ represent the refined fuzzy objects. Here, the neighbourhood refinement only adjusts $\mathbf{O}$ for each reference image and keeps the codes $\mathbf{H}$ unchanged.

Experimental Results

In this section, we carry out comprehensive experiments on three benchmark datasets, *i.e.*, Oxford5K (Philbin et al. 2007) , Paris6K (Philbin et al. 2008) and Sculptures6k (Arandjelović and Zisserman 2011). Object search performance of the proposed framework is evaluated by mean average precision (mAP) which is widely used in evaluating the effectiveness of the image retrieval system.

In this work, the CNN model we used is the 16-layer VGG network (Simonyan and Zisserman 2014) pre-trained on the ImageNet dataset. Each proposal is resized to the size 448×448 prior to being fed into the network. The 512-dimensional local features are extracted from the last convolutional layer conv5_3. In this scenario, the spatial size of conv5_3 layer is $w \times h = 28 \times 28$.

We evaluate our FOM scheme using features from four different aggregation methods: max-pooling, sum-pooling, bilinear-pooling and VLAD. To be more specific, the cluster number of VLAD is set to be 64 and we conduct intra-normalization proposed in (Arandjelovic and Zisserman 2013) to post-process VLAD features. In the VLAD and bilinear pooling aggregations, the dimension of the local convolutional feature is reduced into 64 by principle component analysis (PCA). Finally, all the global features generated from max-pooling, sum-pooling, bilinear-pooling and VLAD are further processed by PCA and whitening followed by ℓ_2 normalization and the dimensions of four types of features are fixed as 512.

Compared with Exhaustive Object Proposals Matching

In Table 1, we evaluate the performance of the proposed Fuzzy Objects Matching (FOM) in comparison with Exhaustive Object Proposals Matching (EOPM) which directly compares the query object with all the object proposals of all the reference images as Eq.(1). We also show the performance from the method using global feature to represent the whole image. It is worth noting that the fuzzy objects we evaluated in this section is the intra-image fuzzy objects without neighborhood refinement. Since our intra-image FOM is an estimation of EOPM, the performance of EOPM should be better than that of the intra-image FOM.

We can observe from Table 1 that the proposed FOM scheme using tens of fuzzy objects (FOs) can achieve comparable mAP with EOPM scheme using 300 object proposals (OPs) in object search. Moreover, comparing the mAP of EOPM using 20 object proposals with the mAP of FOM using 20 fuzzy objects, we find the FOM is much more effective than the EOPM when the memory and computational cost are comparable. Besides, the mAP of the method using global features is much lower than EOPM and FOM, which verifies the effectiveness of the object proposals.

The larger number of the fuzzy objects brings the higher precision as well as higher memory and computational cost. To balance the effectiveness and efficiency, we choose the default number of fuzzy objects as 20. Therefore, for each image, it requires 20×512 dimensions to store the fuzzy objects, $300 \times 3 \times 2$ dimensions to store the value and indices of non-zero elements in $\mathbf{H}$ and 4×300 dimensions to cache the locations of 300 object proposals. In total, for each reference image, it demands only around 13k dimensions to store all the information required by the FOM. In contrast, given 300 object proposals, the dimensions required by the EOPM for each reference image is around 150k, which is much larger than that from the FOM.

Compared with Other Fast Strategies

In this section, we compare the proposed FOM method with other two alternative fast strategies. They are Representative Object Matching and Sparse Coding Matching.

For each reference image, Representative Object Matching (ROM) scheme selects representative proposals from all the object proposals by k-mediods (Park and Jun 2009). To be more specific, the features of n object proposals in the matrix $\mathbf{P} \in \mathbb{R}^{d \times n}$ are grouped into t clusters by k-mediods algorithm and the mediods of the clusters are selected to be the representatives of the n object proposals. In this scenario, we only need to compare the query object with the selected object proposals and both the computational and memory cost are reduced from $\mathcal{O}(nd)$ to $\mathcal{O}(td)$.

Sparse Coding Matching (SCM) (Iscen, Rabbat, and Furon 2016) factorizes matrix $\mathbf{P}$ by sparse coding. The dictionary $\mathbf{D} \in \mathbb{R}^{d \times t}$ is learned by solving the following ℓ_1-penalized optimization:

$$\min_{\mathbf{D}} \|\mathbf{P} - \mathbf{D}\mathbf{C}\|_F^2 + \lambda \|\mathbf{C}\|_{1,1}. \tag{10}$$

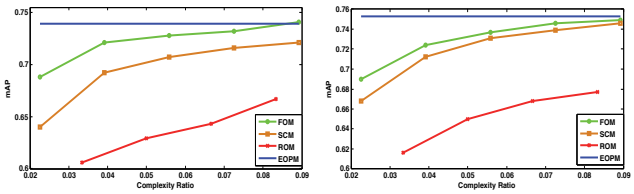
The sparsity of codes $\mathbf{C}$ is strictly controlled by orthogonal matching pursuit and the number of non-zero elements in each column of $\mathbf{C}$ is fixed as z. Computing $\mathbf{q}^\top \mathbf{D}\mathbf{C}$ only requires $\mathcal{O}(td + nz)$ time complexity which is as efficient as the proposed FOM.

We define complexity ratio as the computational complexity of fast strategies over the computational complexity of exhaustive object proposals matching. In this case, the complexity ratio of ROM is t/n and the complexity ratio of FOM and SCM is $t/n + z/d$.

Figure 4 compares the performance of FOM with other two fast strategies using max-pooling features on Oxford5K and Paris6K datasets respectively. In the implementation, we fix the number of non-zero elements z in both FOM and SCM as 3 and changes the number of atoms t from 5 to 20. We can observe that the performance of FOM is much better than ROM. At the same time, the proposed FOM is also better than SCM, which is attributed to the locality-constrained property of the proposed FOM.

Method	Exhaustive Object Proposals Matching						Fuzzy Objects Matching				Global Features
# of OPs/FOs	10	20	40	80	160	300	5	10	20	40	1
Max	49.5	57.7	62.8	68.3	71.9	**73.9**	68.8	72.1	**73.2**	74.5	31.5
Sum	57.5	62.9	67.3	71.1	75.1	**77.0**	65.1	69.7	**72.2**	74.4	40.9
Bilinear	50.8	56.4	63.2	68.9	72.5	**74.5**	59.7	65.4	**67.8**	71.5	37.6
VLAD	57.1	62.1	67.0	71.2	74.3	**76.5**	66.2	70.7	**72.7**	75.1	37.8

Table 1: The performance comparison of EOPM, FOM and the method using Global Features on the Oxford5K dataset. The fuzzy objects are generated from 300 object proposals and the default number of fuzzy objects is 20.



(a) Oxford5K (b) Paris6K

Figure 4: The performance comparison of FOM, SCM and ROM on Oxford5K and Paris6K datasets. FOM gives the best performance.

Dataset	Oxford5K		Paris6K	
Feature	**Intra**	**Inter**	**Intra**	**Inter**
Max	73.2	**83.1**	74.6	**83.8**
Sum	72.2	**76.4**	77.6	**83.5**
Bilinear	67.8	**74.8**	74.3	**80.7**
VLAD	72.7	**80.4**	83.1	**89.0**

Table 2: The performance comparison of intra-image FOM and inter-image FOM on Oxford5K and Paris6K datasets.

Intra-image Fuzzy Objects Versus Inter-image Fuzzy Objects

To validate the performance of the proposed neighborhood refinement, we compare the performance of intra-image FOM and inter-image FOM using max-pooling, sum-pooling, bilinear-pooling and VLAD features. The results are shown in Table 2.

In the experiment, the number of proposals n to generate fuzzy objects is set as 300, the number of fuzzy objects t is set as 20 and the number of neighborhoods m in Eq.(8) is set as 20. As can be seen from the Table 2, the performance of inter-image FOM outperforms that from intra-image FOM in all of four features on these datasets. For example, on the Oxford5K dataset, inter-image FOM improves the mAP of Max-pooling features from 73.2 to 83.1. On the Paris6K dataset, inter-image FOM improves the mAP of VLAD feature from 83.1 to 89.0.

Average Query Expansion

We conduct average query expansion (AQE) (Chum et al. 2007) to further improve the performance of the FOM scheme. For each query q, s fuzzy objects closest to q in the feature space are retrieved. We further compute the mean of feature of query q and features of the s fuzzy objects.

Dataset	Oxford5K		Paris6K	
Feature	**Inter**	**Inter+AQE**	**Inter**	**Inter+AQE**
Max	83.1	**88.9**	83.8	**90.7**
Sum	76.4	**84.8**	83.5	**88.5**
Bilinear	74.8	**82.3**	80.7	**85.4**
VLAD	80.4	**88.7**	89.0	**92.5**

Table 3: The performance of Average Query Expansion on Oxfrord5K and Paris6K datasets.

The generated mean vector will serve as the new feature of the query in order to re-rank the reference images in the database. Table 3 shows the performance of the proposed search system with and without AQE. In our implementation, we set s as 20, which is equal to the number of fuzzy objects. As it can be seen, AQE can effectively improve the performance of the system. For example, on the Oxford5K dataset, it improves the mAP of max-pooling feature by 5.8. On the Paris6K dataset, it improves the mAP of VLAD feature by 3.5.

Comparison with State-of-the-art Methods

The first part of Table 4 shows the object instance search performance from methods using hand-crafted local features. (Perronnin et al. 2010) achieved 41.8 mAP by improved Fisher vector encoding method on the Oxford5K dataset and (Arandjelović and Zisserman 2011) achieved 55.5 mAP on the Oxford5K dataset by aggregating SIFT features using VLAD. In contrast, ours can achieve 88.1 mAP using max-pooling features. In (Tolias, Avrithis, and Jégou 2013), the authors proposed selective matching kernel scheme and achieved excellent performance on both oxford5K and Paris6K datasets with high memory cost. Ours outperforms them with less memory and computation cost.

The second part of Table 4 compares our method with other methods using CNN features. Neural Codes (Babenko et al. 2014) adopt finetuned deep features and achieved 55.7 mAP on Oxford5K dataset. SPoc (Babenko and Lempitsky 2015) achieved 65.7 mAP on Oxford5K dataset by sum-pooling the local convolutional feature. (Ng, Yang, and Davis 2015) aggregated the local convolutional features using VLAD. Note that (Babenko et al. 2014; Babenko and Lempitsky 2015; Ng, Yang, and Davis 2015) all represented the whole image as a global feature and thus might be incapable of handling the scenarios when there are multiple objects in the reference image. We can observe in the Table 4 that the methods in (Babenko et al. 2014; Babenko and Lempitsky 2015; Ng, Yang, and Davis 2015)

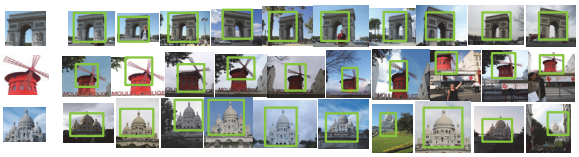
Method	D	O5K	P6K	S6K
Hand-crafted Local Features				
(Perronnin et al. 2010)	2k	41.8	-	-
(Arandjelovic and Zisserman 2013)	32k	55.5	-	-
(Arandjelović and Zisserman 2011)	-	-	-	45.4
(Mikulík et al. 2010)	64k	74.2	74.9	-
(Tao et al. 2014)	-	77.8	-	-
(Tolias, Avrithis, and Jégou 2013)	64k	80.4	77.0	-
CNN Features				
(Babenko et al. 2014)	256	55.7	-	-
(Babenko and Lempitsky 2015)	256	65.7	-	-
(Ng, Yang, and Davis 2015)	128	59.3	59.0	-
(Razavian et al. 2014a)	15k	68.0	79.5	42.3
(Razavian et al. 2014b)	32k	84.4	85.3	67.4
(Tolias, Sicre, and Jégou 2016)	-	77.3	86.5	-
(Mohedano et al. 2016)	-	78.8	84.8	-
(Cao et al. 2016)	-	78.1	87.4	60.8
Our FOM Scheme				
$\text{Max}_{\text{Inter}}$+AQE	13k	**88.9**	**90.7**	65.1
$\text{Sum}_{\text{Inter}}$+AQE	13k	**84.8**	**88.5**	65.8
$\text{Bilinear}_{\text{Inter}}$+AQE	13k	82.3	85.4	59.4
$\text{VLAD}_{\text{Inter}}$+AQE	13k	**88.7**	**92.5**	**73.1**

Table 4: Performance comparisons with state-of-the-art methods on Oxford5K, Paris6K and Sculptures6K datasets. We compare the performance of ours with those methods using hand-crafted local features and CNN features. D denotes the dimensionality of the features.

are not competitive with that from (Razavian et al. 2014b; Tolias, Sicre, and Jégou 2016; Cao et al. 2016).

In (Razavian et al. 2014a; 2014b), the authors cropped both reference images and query images into patches and the relevance between the query and reference images are conducted by cross-matching between patches. Our methods outperform both of them, for example, we achieved 88.1 mAP on the Oxford5K dataset, whereas the mAP of (Razavian et al. 2014b) is only 84.4. Moreover, we should point out that, the cross-matching scheme in (Razavian et al. 2014a) is much less efficient than ours. For example, it need conduct 1024 comparisons between 32 patches from the query and 32 patches from the reference image. In contrast, we only need 20 comparisons between the query and 20 fuzzy objects from the reference image. In (Tolias, Sicre, and Jégou 2016; Mohedano et al. 2016), the authors conducted re-ranking process through spatial search in reference images and achieved much better performance than global representation methods (Babenko et al. 2014; Ng, Yang, and Davis 2015; Babenko and Lempitsky 2015). A most recent work (Cao et al. 2016) achieved 77.3 mAP on the Oxford5k dataset , 86.5 mAP on the Paris6K dataset and 60.8 mAP on the Sculptures6K dataset. In contrast, our FOM with VLAD feature achieved 88.7 mAP on the Oxford5K dataset, 92.5 mAP on the Paris6K dataset and 73.1 mAP on the Sculptures6K dataset. Some visual results of our FOM scheme using VLAD features are shown in Figure 5. We show the estimated best-matched object proposal to validate the performance of our scheme in object localization.

(a) Oxford5K



(b) Paris6K

(c) Sculptures6K

Figure 5: Selected results of top-10 retrievals of our FOM scheme on the Oxford5K and Paris6K datasets. The bounding box in the reference image correspond to the estimated best-matched object proposal.

Conclusion

In this paper, we propose the Fuzzy Objects Matching (FOM) scheme in order to achieve efficient and effective instance search. We conduct comprehensive experiments using object proposal representation generated from four different aggregation methods on three public datasets. The state-of-the-art precision and the efficient implementation verifies the outstanding performance of our method.

Acknowledgements: This work is supported in part by Singapore Ministry of Education Academic Research Fund Tier 2 MOE2015-T2-2-114 and Tier 1 RG27/14, and in part by the Outstanding Youth Teachers Program of BIT under Grant 3070012331601. This research was carried out at the ROSE Lab at the Nanyang Technological University, Singapore. The ROSE Lab is supported by the National Research Foundation, Prime Ministers Office, Singapore, under its IDM Futures Funding Initiative and administered by the Interactive and Digital Media Programme Office.

References

Arandjelović, R., and Zisserman, A. 2011. Smooth object retrieval using a bag of boundaries. In *Proc. of the IEEE International Conference on Computer Vision*, 375–382.

Arandjelovic, R., and Zisserman, A. 2013. All about vlad. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 1578–1585.

Babenko, A., and Lempitsky, V. 2015. Aggregating local deep features for image retrieval. In *Proc. of the IEEE International Conference on Computer Vision*, 1269–1277.

Babenko, A.; Slesarev, A.; Chigorin, A.; and Lempitsky, V. 2014. Neural codes for image retrieval. In *European Conference on Computer Vision*, 584–599.

Bhattacharjee, S. D.; Yuan, J.; Hong, W.; and Ruan, X. 2016a. Query adaptive instance search using object sketches. In *Proc. of the ACM on Multimedia Conference*, 1306–1315.

Bhattacharjee, S. D.; Yuan, J.; Tan, Y.-P.; and Duan, L.-Y. 2016b. Query-adaptive small object search using object proposals and shape-aware descriptors. *IEEE Transactions on Multimedia* 18(4):726–737.

Cao, J.; Liu, L.; Wang, P.; Huang, Z.; Shen, C.; and Shen, H. T. 2016. Where to focus: Query adaptive matching for instance retrieval using convolutional feature maps. *arXiv preprint arXiv:1606.06811*.

Chum, O.; Philbin, J.; Sivic, J.; Isard, M.; and Zisserman, A. 2007. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proc. of the IEEE International Conference on Computer Vision*, 1–8.

Gao, Y.; Beijbom, O.; Zhang, N.; and Darrell, T. 2016. Compact bilinear pooling. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 317–326.

Iscen, A.; Rabbat, M.; and Furon, T. 2016. Efficient large-scale similarity search using matrix factorization. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2073–2081.

Jiang, Y.; Meng, J.; Yuan, J.; and Luo, J. 2015. Randomized spatial context for object search. *IEEE Transactions on Image Processing* 24(6):1748–1762.

Jiang, Y.; Meng, J.; and Yuan, J. 2012. Randomized visual phrases for object search. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 3100–3107. IEEE.

Meng, J.; Yuan, J.; Jiang, Y.; Narasimhan, N.; Vasudevan, V.; and Wu, Y. 2010. Interactive visual object search through mutual information maximization. In *Proc. of the ACM International Conference on Multimedia*, 1147–1150.

Meng, J.; Yuan, J.; Yang, J.; Wang, G.; and Tan, Y.-P. 2016. Object instance search in videos via spatio-temporal trajectory discovery. *IEEE Transactions on Multimedia* 18(1):116–127.

Mikulík, A.; Perdoch, M.; Chum, O.; and Matas, J. 2010. Learning a fine vocabulary. In *European Conference on Computer Vision*, 1–14.

Mohedano, E.; Salvador, A.; McGuinness, K.; Marques, F.; O'Connor, N. E.; and Giro-i Nieto, X. 2016. Bags of local convolutional features for scalable instance search. *arXiv preprint arXiv:1604.04653*.

Mopuri, K., and Babu, R. 2015. Object level deep feature pooling for compact image representation. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 62–70.

Ng, J.; Yang, F.; and Davis, L. 2015. Exploiting local features from deep networks for image retrieval. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 53–61.

Park, H.-S., and Jun, C.-H. 2009. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications* 36(2):3336–3341.

Perronnin, F.; Liu, Y.; Sánchez, J.; and Poirier, H. 2010. Large-scale image retrieval with compressed fisher vectors. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 3384–3391.

Philbin, J.; Chum, O.; Isard, M.; Sivic, J.; and Zisserman, A. 2007. Object retrieval with large vocabularies and fast spatial matching. In *Proc. of the IEEE International Conference on Computer Vision*, 1–8.

Philbin, J.; Chum, O.; Isard, M.; Sivic, J.; and Zisserman, A. 2008. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–8.

Razavian, A. S.; Azizpour, H.; Sullivan, J.; and Carlsson, S. 2014a. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 512–519.

Razavian, A. S.; Sullivan, J.; Maki, A.; and Carlsson, S. 2014b. A baseline for visual instance retrieval with deep convolutional networks. *arXiv preprint arXiv:1412.6574*.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Tao, R.; Gavves, E.; Snoek, C. G.; and Smeulders, A. W. 2014. Locality in generic instance search from one example. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2091–2098.

Tolias, G.; Avrithis, Y.; and Jégou, H. 2013. To aggregate or not to aggregate: Selective match kernels for image search. In *Proc. of the IEEE International Conference on Computer Vision*, 1401–1408.

Tolias, G.; Sicre, R.; and Jégou, H. 2016. Particular object retrieval with integral max-pooling of cnn activations. In *Proc. of International Conference on Learning Representations*.

Wang, J.; Yang, J.; Yu, K.; Lv, F.; Huang, T.; and Gong, Y. 2010. Locality-constrained linear coding for image classification. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 3360–3367.

Wu, Y.; Wang, Z.; Yuan, J.; and Duan, L. 2016. A compact binary aggregated descriptor via dual selection for visual search. In *Proc. of the ACM on Multimedia Conference*, 426–430.

Zitnick, C. L., and Dollár, P. 2014. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, 391–405.