21st IAPR Int. Conf. on Discrete Geometry for Computer Imagery (DGCI), March 2019

Common Object Discovery as Local Search for Maximum Weight Cliques in a Global Object Similarity Graph

Cong Rao[1], Yi Fan[2,3], Kaile Su[3], and Longin Jan Latecki[1]

[1] Temple University, Philadelphia, USA
{cong.rao,latecki}@temple.edu
[2] Guangxi Key Lab of Trusted Software,
Guilin University of Electronic Technology, Guilin 541004, China
[3] Griffith University, Brisbane, Australia
{yi.fan4,k.su}@griffithuni.edu.au

Abstract. In this paper, we consider the task of discovering the common objects in images. Initially, object candidates are generated in each image and an undirected weighted graph is constructed over all the candidates. Each candidate serves as a node in the graph while the weight of the edge describes the similarity between the corresponding pair of candidates. The problem is then expressed as a search for the Maximum Weight Clique (MWC) in this graph. The MWC corresponds to a set of object candidates sharing maximal mutual similarity, and each node in the MWC represents a discovered common object across the images. Since the problem of finding the MWC is NP-hard, most research of the MWC problem focuses on developing various heuristics for finding good cliques within a reasonable time limit. We utilize a recently very popular class of heuristics called local search methods. They search for the MWC directly in the discrete domain of the solution space. The proposed approach is evaluated on the PASCAL VOC image dataset and the YouTube-Objects video dataset, and it demonstrates superior performance over recent state-of-the-art approaches.

Keywords: Common Object Discovery · Visual Similarity · Maximum Weight Clique · Local Search Algorithm

1 Introduction

For an undirected weighted graph $G = (V, E)$, where V is the set of vertices and E is the set of edges, a clique C is a subset of vertices in V in which each pair of vertices is connected by an edge in E. The Maximum Weight Clique (MWC) problem is to find a clique C which maximizes

$$w(C) = \sum_{v_i \in C} w_V(v_i) + \sum_{v_i, v_j \in C} w_E(v_i, v_j), \tag{1}$$

where $w_V : V \to \mathbb{R}$ and $w_E : E \to \mathbb{R}$ are the weight functions for the vertices and edges respectively. Successfully solving the MWC problem leads to various applications in practice.

In this paper, we focus on the task of common object discovery, which aims at discovering the objects of the same class in an image collection. Co-localizing objects in unconstrained environment is challenging. For images in the real-world applications, such as those in the PASCAL datasets [8, 9], the objects of the same class may look very different due to viewpoint, occlusion, deformation, illumination, *etc.* Also, there could be considerable diversities within certain object class such as human beings, for their differences in gender, age, costume, hair style or skin color. Besides, there could be multiple common objects in the given set of images, thus the definition of "common" may be ambiguous. In addition, the efficiency of the involved method is very significant in time sensitive applications such as object co-localization in large collections of images or video streams.

To achieve robust and efficient object co-localization, we formulate the task as a Maximum Weight Clique (MWC) problem. It aims at finding a group of objects that are most similar to each other, which corresponds to a MWC in the associated graph. The nodes in the graph correspond to the object candidates generated from the given image collection, while the weight on an edge indicates how similar two given candidates are. We can discover a set of common objects by finding the MWC in the associated graph. Each node in the MWC is a discovered common object across the images. The main idea of the paper is illustrated in Figure 1.

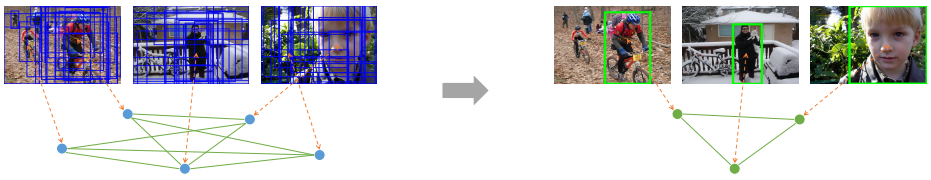

Fig. 1. Given a set of object candidates generated from an image collection (left), our goal is to find common objects by searching for the maximum weight clique in the associated graph. Each node in the clique (right) corresponds to a discovered common object.

The main contributions of this work are as follows. 1) We address the task of object co-localization as a well-defined MWC problem in the associated graph. It provides a practical and general solution for research and applications related to the MWC problem. 2) We develop a hashing based mechanism to detect the revisiting of the local optimum in the local search based MWC solver [35]. It can alleviate the cycling issue in the optimization process. 3) The Region Proposal Network (RPN) [25] is applied for efficiently generating the object

candidates. The candidates are then re-ranked to improve the robustness against the background noise. 4) A Triplet Network (TN) is learned to obtain the feature embeddings of the object candidates, so as to construct a reliable affinity measure between the candidates. 5) The performance is evaluated on the PASCAL VOC 2007 image dataset [8] and the YouTube-Objects video dataset [16]. Superior performance is obtained compared to recent state-of-the-art methods.

2 Related Works

The problem of common object discovery has been investigated extensively in the past few years. Papazoglou *et al* [22] view the task as a foreground object mining problem, where Optical Flow is used to estimate the object motion and the Gaussian Mixture model is utilized to capture the appearance of the foreground and background. Cho *et al* [6] tackle the problem using a part-based region matching method, where a probabilistic Hough transform is used to evaluate the quality of each candidate correspondence. Joulin *et al* [15] extend the method in [6] to co-localize objects in video frames, and a Frank-Wolfe algorithm is used to optimize the proposed quadratic programming problem. Zhang *et al* [37] apply a part-based object detector and a motion aware region detector to generate object candidates. The problem is then formulated as a joint assignment problem and the solution is refined by inferring shape likelihoods afterwards. Kwak *et al* [17] also focus on the problem of localizing dominant objects in videos, where an iterative process of detection and tracking is applied. Li *et al* [18] devise an entropy-based objective function to learn a common object detector, and they address the task with a Conditional Random Field (CRF) model. Wei *et al* [36] perform Principal Component Analysis (PCA) on the convolutional feature maps of all the images, and locate the most correlated regions across the images. Wang *et al* [32] use segmentations produced by Fully Convolutional Networks (FCN) as object candidates. Then they discover common objects by solving a N-Partite Graph Matching problem.

Many of these methods explicitly or implicitly employ graph based models to interpret the task of object co-localization. Similarly in this paper, an undirected weighted graph is first constructed over the given set of images, modeling the visual affinities between the object candidates. We find the common objects as the Maximum Weight Clique (MWC) in this graph, where each node in the clique corresponds to a detected common object across the images. The MWC problem is NP-hard and it is difficult to obtain a global optimal solution. Generally, there are two types of algorithms to solve the MWC problem: the exact methods such as [12, 14] and the heuristic methods such as [2, 24, 21, 10, 35]. Most existing works on the MWC problem focused on the heuristic approaches due to their efficiency in space and time. In this paper, our optimization algorithm adopts a simple variant of Tabu Search (TS) heuristic to discover the MWC, and it has several features: 1) it considers the local circumstance of a vertex in each step; 2) it takes only an auxiliary Boolean array for implementation; 3) it requires no extra parameters besides the time limit.

3 Problem Formulation

Given a set of N images $\mathcal{I} = \{I_1, I_2, \ldots, I_N\}$, we generate a set of object candidates from all images $\mathcal{B} = \{\boldsymbol{b} \mid \boldsymbol{b} \in \mathcal{P}(I),\ I \in \mathcal{I}\}$, where $\mathcal{P}(I)$ is the set of object candidates extracted from image I and $\boldsymbol{b}$ is a bounding box of that object candidate. Suppose n_i object candidates are extracted from image I_i, then $|\mathcal{B}| = \sum_{i=1}^{N} n_i$ candidates will be generated from the image collection $\mathcal{I}$ in total. We denote $n = |\mathcal{B}|$ in the remainder of the paper. Let $o(\boldsymbol{b}_i)$ be the score of some bounding box $\boldsymbol{b}_i$ containing the common object, and let $s(\boldsymbol{b}_i, \boldsymbol{b}_j)$ represent the similarity between two object candidates in $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$, then the task of object co-localization can be formulated as finding an optimal subset $\mathcal{B}^* \subset \mathcal{B}$ such that

$$w(\mathcal{B}^*) = \sum_{\boldsymbol{b}_i \in \mathcal{B}^*} o(\boldsymbol{b}_i) + \sum_{\boldsymbol{b}_i, \boldsymbol{b}_j \in \mathcal{B}^*, \boldsymbol{b}_i \neq \boldsymbol{b}_j} s(\boldsymbol{b}_i, \boldsymbol{b}_j) \tag{2}$$

is maximized, with the constraint that at most one object candidate can be selected from each image. For the reason explained in Section 4.2, we set $o(\boldsymbol{b}_i) = 0$ for all $\boldsymbol{b}_i$, which means it is a Maximum Edge Weight Clique problem. However, the proposed MWC solver can optimize problems with both vertex and edge weights.

Further, we assign a label $x_i \in \{0, 1\}$ to each object candidate $\boldsymbol{b}_i$, where $x_i = 1$ means that the object candidate $\boldsymbol{b}_i$ is selected in the subset $\mathcal{B}^*$. Thus, an indicator vector $\boldsymbol{x} \in \{0, 1\}^n$ is used to identify the common objects discovered in $\mathcal{B}$. Besides, an affinity matrix $A \in \mathbb{R}^{n \times n}$ is constructed, where

$$A_{ii} = o(\boldsymbol{b}_i),\ \forall \boldsymbol{b}_i \in \mathcal{B}, \quad \text{and} \quad A_{ij} = s(\boldsymbol{b}_i,\ \boldsymbol{b}_j),\ \forall \boldsymbol{b}_i, \boldsymbol{b}_j \in \mathcal{B}. \tag{3}$$

Here we assume the similarity metric $s(\boldsymbol{b}_i,\ \boldsymbol{b}_j)$ is symmetric and non-negative, namely $A_{ij} = A_{ji} \geq 0$. On the other hand, we remove the edge between object candidates $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ if they are present in the same image, hence they cannot be simultaneously selected in $\mathcal{B}^*$. Then the problem in (2) can be expressed as finding an optimal indicator vector $\boldsymbol{x} \in \{0, 1\}^n$, such that $\boldsymbol{x}^T A \boldsymbol{x}$ is maximized. Hence the selected nodes in $\mathcal{B}^*$ correspond to a MWC in the constructed graph, and they represent the discovered set of common objects. To summarize, the overall objective function of the MWC problem can be written in the matrix form as

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmax}}\ \ \boldsymbol{x}^T A \boldsymbol{x}, \quad \text{s. t.}\ \ \boldsymbol{x} \in \{0, 1\}^n. \tag{4}$$

To this end, the task of object co-localization is formulated as a Maximum Weight Clique (MWC) problem as described in (1).

4 Graph Construction

4.1 Object Candidates Generation

The nodes in the associated graph correspond to the object candidates in all the images. We expect those candidates to cover as many foreground objects as

possible. Meanwhile, the total number of candidates will also influence the search space for the MWC. Therefore, our first priority is to find a proper method to extract the object candidates. The Region Proposal Networks (RPN) [25] is used in our approach to generate rectangular object candidates from each image. We use the raw RPN proposals in the intermediate stage and apply Non-Maximum Suppression (NMS) [28] to remove redundant boxes. We choose the top-K scoring proposals from each image to construct the associated graph for computational efficiency. We consider two different proposal scoring measures. The first one is commonly used and is based on RPN objectness score of each object candidate. RPN also generates a vector of class likelihoods for each object candidate, and we propose to re-rank the object candidates according to the entropy of the class distribution. Since the entropy is a measure of uncertainty, it serves a similar purpose as the objectness score but tend to be more accurate in this setting. Hence we can re-rank the raw RPN proposals according to the entropy, and select the top-K scoring boxes with low uncertainty as object candidates in each image.

4.2 Common Objectness Score

For object co-localization, the underlying class of the common object is unknown in advance. Thus, the score $o(\boldsymbol{b})$ of some object $\boldsymbol{b}$ being the common one is difficult to estimate. A possible way is to set $o(\boldsymbol{b})$ as the objectness score of $\boldsymbol{b}$. But this can be problematic when $\boldsymbol{b}$ indeed contains an object but not the common one. Thus, it may lead to unexpected results if the objectness score is directly used, as observed in [31]. Therefore, we set the contribution of the score to the objective function (2) to zero, *i.e.*,

$$A_{ii} = o(\boldsymbol{b}_i) = 0, \forall \boldsymbol{b}_i \in \mathcal{B}. \tag{5}$$

In the case of object co-localization, it means we focus on the MWC problem with edge weight only. However, as shown in Section 5, the proposed MWC problem solver is generic and can be applied to other tasks where both vertex weights and edge weights are present.

4.3 Object Representation and Similarity

The edge weights in the associated graph represent visual similarity between the selected object candidates. Thus, we need an accurate way to represent the object candidates and evaluate their similarities. In this paper, we employ the Triplet Network framework [13] to learn the deep feature embeddings of the object candidates. Suppose a pre-trained Convolutional Neural Network (CNN) is selected to extract the deep features $f(\boldsymbol{b}; \boldsymbol{w})$ for each object candidate $\boldsymbol{b} \in \mathcal{B}$, where $\boldsymbol{w}$ is the set of parameters of the CNN. In this framework, a set of triplets is then constructed for fine-tuning the parameters $\boldsymbol{w}$. Each triplet consists of a reference object $\boldsymbol{b}_r$, a positive object $\boldsymbol{b}_p$ and a negative object $\boldsymbol{b}_n$. Namely, $\boldsymbol{b}_r$ and $\boldsymbol{b}_p$ represent a pair of similar objects, while $\boldsymbol{b}_r$ and $\boldsymbol{b}_n$ are a pair of dissimilar

objects. Two objects are viewed as similar if they belong to the same category and otherwise dissimilar. Then, the hinge loss of a triplet is defined as

$$l(\boldsymbol{b}_r, \boldsymbol{b}_p, \boldsymbol{b}_n) = \max\{0, \lambda + s(\boldsymbol{b}_r, \boldsymbol{b}_n) - s(\boldsymbol{b}_r, \boldsymbol{b}_p)\}, \tag{6}$$

where λ is a margin threshold controlling how different $s(\boldsymbol{b}_r, \boldsymbol{b}_n)$ and $s(\boldsymbol{b}_r, \boldsymbol{b}_p)$ should be. The goal of the Triplet Network learning is to find a set of optimal parameters $\boldsymbol{w}$, such that the sum of the hinge loss of all triplets

$$L(\mathcal{T}) = \sum_{(\boldsymbol{b}_r, \boldsymbol{b}_p, \boldsymbol{b}_n) \in \mathcal{T}} l(\boldsymbol{b}_r, \boldsymbol{b}_p, \boldsymbol{b}_n) \tag{7}$$

is minimized over a training set of triplets $\mathcal{T}$. Namely, in the specified metric space, the learning process makes similar objects closer to each other, while dissimilar objects are pushed away. In the triplet hinge loss $l(\boldsymbol{b}_r, \boldsymbol{b}_p, \boldsymbol{b}_n)$, frequently used similarity metrics include dot-product (the linear kernel) and the Euclidean distance. But the output ranges of these metrics are not bounded, and this may invalidate the margin threshold λ in the loss function, as observed in [5]. In addition, more complex metrics can be also used here, such as the polynomial kernel and the Gaussian kernel (the RBF kernel). But there are a few more parameters in these kernel functions and they have to be chosen wisely. For simplicity, we define $s(\boldsymbol{b}_i, \boldsymbol{b}_j)$ as the cosine similarity between two CNN feature vectors $f(\boldsymbol{b}_i; \boldsymbol{w})$ and $f(\boldsymbol{b}_j; \boldsymbol{w})$, namely

$$A_{ij} = s(\boldsymbol{b}_i, \boldsymbol{b}_j) = \frac{f(\boldsymbol{b}_i; \boldsymbol{w})^T f(\boldsymbol{b}_j; \boldsymbol{w})}{\|f(\boldsymbol{b}_i; \boldsymbol{w})\| \|f(\boldsymbol{b}_j; \boldsymbol{w})\|}, \tag{8}$$

since it is already neatly bounded and parameter free. The parameters $\boldsymbol{w}$ in the overall loss function (7) can be updated via the standard Stochastic Gradient Descent (SGD) method. An intuitive description of our triplet network can be found in Figure 2.

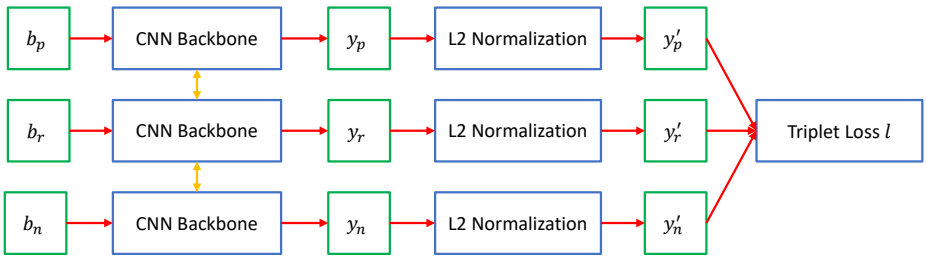

Fig. 2. The architecture of our triplet network. The weights in the CNN backbones are shared in the three branches. The goal is to learn a feature embedding such that similar objects are closer to each other in the metric space while dissimilar objects are pushed away.

5 The MWC Problem Solver

We use a local search based method to solve the MWC problem (4). The local search usually moves from one clique to another until it reaches the cutoff, then the best clique found is kept as the solution. The pipeline of our MWC solver is summarized in Algorithm 1.

Algorithm 1: Our MWC Problem Solver

Input : An undirected weighted graph $G = (V, E)$ and a time limit t
Output: A clique C^* with the maximum clique weight

$C^* \leftarrow C \leftarrow \emptyset$; $lastStepImproved \leftarrow true$;
$step \leftarrow 1$; $confChange(v) \leftarrow 1, \forall v \in V$;
while *elapsed time* $< t$ **do**
  if $C = \emptyset$ **then** add a random vertex into C;
  $v \leftarrow \text{argmax}_v\ score(v, C), v \in S_{add}(C)$, s.t. $confChange(v) = 1$;
  $(u, u') \leftarrow \text{argmax}_{(u,u')}\ score(u, u', C), (u, u') \in S_{swap}(C)$, s.t. $confChange(u') = 1$;
  if $v \neq null$ **then**
    if $(u, u') =$ *(null, null)* or $score(v) > score(u, u')$ **then**
      $C \leftarrow C \cup \{v\}$;
    else
      $C \leftarrow C \backslash \{u\} \cup \{u'\}$;
    $lastStepImproved \leftarrow true$;
  else
    if $(u, u') = (null, null)$ *or* $score(u, u') < 0$ **then**
      if $lastStepImproved =$ true **then**
        if $w(C) > w(C^*)$ **then** $C^* \leftarrow C$;
        if $hash(C)$ *is already marked* **then**
          Drop all the vertices in C;
          continue;
        Label $hash(C)$ as marked;
      $lastStepImproved \leftarrow false$;
    else
      $lastStepImproved \leftarrow true$;
    $v' \leftarrow \text{argmax}_{v'}\ score(v', C), v' \in C$;
    if $(u, u') =$ *(null, null)* or $score(v', C) > score(u, u', C)$ **then**
      $C \leftarrow C \backslash \{v'\}$;
    else
      $C \leftarrow C \backslash \{u\} \cup \{u'\}$;
  Apply the Strong Configuration Checking (SCC) strategy;
  $step$++;
return C^*;

Compared to RSL and RRWL in [11], our algorithm starts from a random single-vertex clique, while they start with a random maximal clique. This is particularly useful when the run-time is restricted. Besides, while RSL and RRWL restart when a solution is revisited in the so-called *first growing step*, our algorithm simply restarts when a local optimum is revisited. In this way, our solver spends less time on searching the local area that has been visited intensively.

5.1 Detecting Revisiting via a Hash Table

In recent methods, the local search typically moves in a deterministic way, *i.e.*, no randomness exists in this process. Thus, a sequence of steps from a previously visited local optimum would be simply repeated, and it may not improve the best clique found so far. Hence, we improve this kind of methods by introducing a cycle elimination based restart strategy, where a hash table is used to approximately detect the revisiting of a local optimum. Given a candidate solution $\mathcal{B}_c^*$ and a prime number p, we define the hash value of $\mathcal{B}_c^*$ as

$$hash(\mathcal{B}_c^*) = (\sum_{\boldsymbol{b}_i \in \mathcal{B}_c^*} 2^i) \mod p, \tag{9}$$

where $i \in \{1, 2, \ldots, n\}$ is the index of $\boldsymbol{b}_i$ in the entire object candidate set $\mathcal{B}$. If p is large enough, the chance of collision is negligible. The parameter p can be set according to the memory capacity of the machine. In the proposed algorithm, the revisiting of a local optimum is detected by checking whether the respective hash entry has been visited. If the local optimum was not visited before, the local search continues. Otherwise, the solver will be restarted and try to look for a better solution.

5.2 Scoring Functions and Candidate Nodes

Given an undirected weighted graph $G = (V, E)$, we describe our approach to finding the MWC in Algorithm 1. To begin with, we first introduce some notations used in our algorithm. In the local search for the MWC, the `add` operation adds a new node to the current clique C. The `drop` operation drops an existing node from the current clique C. The `swap` operation swaps two nodes from inside and outside the current clique C. Each operation returns a new clique as the current solution, which maximizes the gain of the clique weight. Suppose $w(C)$ is the weight of a clique C defined in Equation (1), then for the `add` and `drop` operation, the gain of adding and dropping a node v is computed as

$$score(v, C) = \begin{cases} w(C \cup \{v\}) - w(C) & \text{if } v \notin C; \\ w(C \backslash \{v\}) - w(C) & \text{if } v \in C. \end{cases} \tag{10}$$

For `swap` operation, the gain of clique weight when swapping two nodes (u, v) is

$$score(u, v, C) = w(C \backslash \{u\} \cup \{v\}) - w(C), u \in C, v \notin C, (u, v) \notin E. \tag{11}$$

We denote the set of neighbors of a vertex v as $\mathcal{N}(v) = \{u|(u,v) \in E\}$. To ensure that the local search always maintains a clique, we define two operand sets. Firstly for a clique C, we define the set of candidate nodes for the `add` operation as

$$S_{add}(C) = \begin{cases} \{v|v \notin C, v \in \mathcal{N}(u), \forall u \in C\} & \text{if } |C| > 0; \\ \emptyset, \text{ otherwise.} \end{cases} \tag{12}$$

Secondly, the set of candidate node pairs for the `swap` operation is defined as

$$S_{swap}(C) = \begin{cases} \{(u,v)|u \in C, v \notin C, (u,v) \notin E, v \in \mathcal{N}(w), \forall w \in C \backslash \{u\}\} \text{ if } |C| > 1; \\ \emptyset, \text{ otherwise.} \end{cases} \tag{13}$$

To maximize the gain of clique weight in each step, the `add` operation adds a node v^* to the current clique C such that $v^* = \text{argmax}_v \, score(v, C), v \in S_{add}(C)$. The `drop` operation drops a node $v^* = \text{argmax}_v \, score(v, C), v \in C$ from the current clique C. The `swap` operation swaps two nodes (u^*, v^*) such that $(u^*, v^*) = \text{argmax}_{(u,v)} \, score(u, v, C), (u, v) \in S_{swap}(C)$.

5.3 The Strong Configuration Checking Strategy

We apply the Strong Configuration Checking (SCC) strategy [35] to avoid revisiting a solution too early. The main idea of the SCC strategy works as follows. After a vertex v is dropped or swapped from a clique C, it can be added or swapped back into C only if one of its neighbors is added into C. Suppose $confChange(v)$ is an indicator function of node v, where $confChange(v) = 1$ means v is allowed to be added or swapped into the candidate solution and $confChange(v) = 0$ means v is forbidden to be added or swapped into the candidate solution, then the SCC strategy specifies the following rules:

1. Initially $confChange(v)$ is set to 1 for each vertex v;
2. When v is added, $confChange(u)$ is set to 1 for all $u \in \mathcal{N}(v)$;
3. When v is dropped, $confChange(v)$ is set to 0;
4. When $(u, v) \in S_{swap}(C)$ are swapped, $confChange(u)$ is set to 0.

6 Experiments

To evaluate the performance of our method in comparison to other approaches, experiments are conducted on the PASCAL VOC 2007 image dataset [8] and the YouTube-Objects video dataset [16]. The standard PASCAL criterion Intersection over Union (IoU) is adopted for evaluation. Namely, a predicted bounding box $\boldsymbol{b}^p$ is correct if $IoU(\boldsymbol{b}^p, \boldsymbol{b}^{gt}) = \frac{area(\boldsymbol{b}^p \cap \boldsymbol{b}^{gt})}{area(\boldsymbol{b}^p \cup \boldsymbol{b}^{gt})} > 0.5$, where $\boldsymbol{b}^{gt}$ is a ground-truth annotation of the bounding box. Finally, the percentage of images with correct object localization (CorLoc) [18] is used as the evaluation protocol. Our method is denoted as LSMWC for local search MWC solver.

6.1 Implementation Details

Our experiments are carried out on a desktop machine with two Intel(R) Core(TM) i7 CPUs (2.80GHz) and 64 GB memory. A GeForce GTX Titan X GPU is used for training and testing related deep neural networks. The proposed MWC solver is implemented in C/C++. The deep learning framework Caffe and MatConvNet are utilized as carriers for building the Region Proposal Network and the Triplet Network. The pipeline of the system is organized in MATLAB with some utilities written as MEX files, due to the efficiency for high level data management and visualization. The default parameters are used to learn RPN and generate the object candidates. A threshold of 0.5 is used for the NMS process to remove redundant object proposals. The best $K = 20$ object candidates are selected in each image. We set $\lambda = 0.25$ in the hinge loss (6) of a triplet. The prime number p in the hash function (9) is set to $10^9 + 7$, thus the hash table consumes around 1 GB memory. The RPN and Triplet Network in our method are built upon the VGG-f model [30] as well as the VGG-16 model [4]. Compared to the VGG-16 model, the structure of the VGG-f model is much simpler thus more computationally efficient. The VGG-f and VGG-16 models are pre-trained on the ImageNet dataset [29] and fine-tuned on the Microsoft COCO dataset [19]. All parameters are fixed the same in the experiments unless explicitly stated otherwise.

6.2 Experiments on the PASCAL07 Dataset

The PASCAL VOC 2007 dataset [8] is used to evaluate the performance of object co-localization in images. The dataset is split as a training-validation set and a test set, each with about 5,000 images in 20 classes. We follow [15] to construct a collection of images for object co-localization from the training-validation set and denote it as PASCAL07. This is fine in our framework, since our RPN and Triplet networks are not trained on this dataset but on the ImageNet and COCO datesets as stated in Section 6.1.

We first compare the co-localization accuracy of different MWC problem solvers on the PASCAL07 dataset in Table 1. The graph instances of these MWC problems are constructed based on the VGG-16 model. Since the PASCAL07 dataset has images from 20 different classes, we construct 20 different graphs, one graph for each image class. For the experiments on the PASCAL07 dataset, the average number of nodes in the constructed graphs is 6081.67, and the average number of edges is 2.16×10^7. The average density of the graphs is 0.9962. Different solvers are evaluated on exactly the same MWC problem instances constructed by our co-localization framework. As randomized processes may exist in different methods, the reported accuracy is taken as the average over 10 runs with different seeds for the random number generator.

For the method [20], it solves the MWC problem in the relaxed continuous domain and a modified Frank-Wolfe algorithm is proposed to attack the problem. Similar to our approach, the solver TBMA [1] also solves the MWC problem directly in the discrete domain. Compared to their solver, our solver will restart

Method	Aero	Bike	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	Motor	Person	Plant	Sheep	Sofa	Train	TV	Avg
MC [20]	63.0	45.7	56.1	51.9	14.3	47.8	71.9	61.1	36.2	**72.3**	**46.0**	57.2	61.3	79.6	62.3	34.3	69.8	39.3	59.4	9.8	52.0
LSCC [35]	61.3	68.3	61.2	48.1	16.8	67.7	76.9	58.5	41.8	**72.3**	24.5	63.9	68.3	75.5	69.0	28.6	**76.0**	47.2	62.1	68.4	57.8
[35] + BMS	63.9	68.3	60.9	50.3	49.2	66.7	76.9	59.3	41.1	**72.3**	23.0	**65.1**	68.3	77.1	**69.8**	27.8	**76.0**	45.9	62.8	68.4	59.7
TBMA [1]	62.2	**69.5**	**62.1**	52.5	18.4	**71.5**	78.5	61.1	**49.2**	70.9	30.0	62.0	69.7	80.8	66.0	**49.4**	70.8	50.2	63.6	68.0	60.3
LSMWC	**64.7**	58.4	60.3	**54.1**	**52.0**	71.0	**79.2**	**63.8**	43.1	71.6	40.5	64.4	**72.1**	**84.9**	69.5	45.3	75.0	**51.1**	**66.3**	**71.1**	**62.9**

Table 1. Co-localization CorLoc (%) of different MWC solvers on the PASCAL07 dataset.

CNN Backbone	RPN Objectness	Object Proposal Re-ranking	Triplet Loss Fine-tuning
Pre-trained VGG-f Model	31.2	53.8	59.3
Pre-trained VGG-16 Model	33.1	56.1	62.9

Table 2. Co-localization CorLoc (%) of different strategies on the PASCAL07 dataset.

if a local optimum is revisited, while TBMA will restart if the solution quality has not been improved for a specified number of steps. As for the solver LSCC [35], originally it is dedicated to solve the MWC problems where the edge weights are absent. Namely, the add, swap or drop operations change the weight of a clique considering related vertex weights only. Here we modify it so that the edge weights are taken into account in these operations. Two versions of the LSCC solver in the original paper are evaluated, and they serve as the baseline results in our experiment. The experiments justify our choice of the MWC problem solver, which improves the accuracy of object co-localization.

The co-localization accuracy of different object candidate generation and feature embedding methods on the PASCAL07 dataset is compared in Table 2. Different CNN models are used to extract the object candidate features, then the cosine similarity is applied on these deep neural network features. It shows that re-ranking the object proposals in each image according to the entropy of the class distribution of each object proposal leads to significantly better results than directly using the RPN objectness score of each proposal for ranking. With the involvement of the Triplet Network learning framework, the co-localization performance improves further. The experiments validate that the performance of the object co-localization is benefited from the proper choice of the object candidate generation and feature embedding scheme.

The co-localization accuracy of different object co-localization methods on the PASCAL07 dataset is reported in Table 3. The results of the compared methods are directly taken from the corresponding literature. Among these methods using deep CNN features as visual descriptors [18, 34, 3, 26, 36, 7, 27], our method demonstrates superior results over recent state-of-the-art methods. The experiments confirm the effectiveness of the proposed object co-localization framework.

6.3 Experiments on the YouTube-Objects Dataset

The Youtube-Objects dataset [16] is used for object co-localization in videos. The dataset contains videos collected from YouTube with 10 object classes. There are

Method	Aero	Bike	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	Motor	Person	Plant	Sheep	Sofa	Train	TV	Avg
Joulin *et al* [15]	32.8	17.3	20.9	18.2	4.5	26.9	32.7	41.0	5.8	29.1	34.5	31.6	26.1	40.4	17.9	11.8	25.0	27.5	35.6	12.1	24.6
Cho *et al* [6]	50.3	42.8	30.0	18.5	4.0	62.3	64.5	42.5	8.6	49.0	12.2	44.0	64.1	57.2	15.3	9.4	30.9	34.0	61.6	31.5	36.6
Li *et al* [18]	73.1	45.0	43.4	27.7	6.8	53.3	58.3	45.0	6.2	48.0	14.3	47.3	69.4	66.8	24.3	12.8	51.5	25.5	65.2	16.8	40.0
Wang *et al* [34]	37.7	58.8	39.0	4.7	4.0	48.4	70.0	63.7	9.0	54.2	33.3	37.4	61.6	57.6	30.1	31.7	32.4	52.8	49.0	27.8	40.2
Bilen *et al* [3]	66.4	59.3	42.7	20.4	21.3	63.4	74.3	59.6	21.1	58.2	14.0	38.5	49.5	60.0	19.8	39.2	41.7	30.1	50.2	44.1	43.7
Ren *et al* [26]	79.2	56.9	46.0	12.2	15.7	58.4	71.4	48.6	7.2	69.9	16.7	47.4	44.2	75.5	41.2	39.6	47.4	32.2	49.8	18.6	43.9
Wei *et al* [36]	67.3	63.3	61.3	22.7	8.5	64.8	57.0	**80.5**	9.4	49.0	22.5	72.6	73.8	69.0	7.2	15.0	35.3	**54.7**	75.0	29.4	46.9
Wang *et al* [33]	**80.1**	63.9	51.5	4.9	21.0	55.7	74.2	43.5	26.2	53.4	16.3	56.7	58.3	69.5	14.1	38.3	58.8	47.2	49.1	60.9	48.5
Cinbis *et al* [7]	67.1	66.1	49.8	34.5	23.3	68.9	**83.5**	44.1	27.7	71.8	**49.0**	48.0	65.2	79.3	37.4	42.9	65.2	51.9	62.8	46.2	54.2
Rochan *et al* [27]	78.5	63.3	**66.3**	**56.3**	19.6	**82.2**	74.7	69.1	22.4	72.3	31.0	62.9	**74.9**	78.3	48.6	29.3	64.5	36.2	**75.8**	69.5	58.8
LSMWC (VGG-f)	59.7	**67.1**	60.3	46.4	51.2	68.8	75.9	57.9	40.4	**77.3**	21.5	**64.6**	65.2	74.7	67.3	41.6	**77.1**	48.0	60.9	60.9	59.3
LSMWC (VGG-16)	64.7	58.4	60.3	54.1	**52.0**	71.0	79.2	63.8	**43.1**	71.6	40.5	64.4	72.1	**84.9**	**69.5**	**45.3**	75.0	51.1	66.3	**71.1**	**62.9**

Table 3. Co-localization CorLoc (%) of different methods on the PASCAL07 dataset.

Method	Aeroplane	Bird	Boat	Car	Cat	Cow	Dog	Horse	Motorbike	Train	Average
Prest *et al* [23]	51.7	17.5	34.4	34.7	22.3	17.9	13.5	26.7	41.2	25.0	28.5
Joulin *et al* [15]	25.1	31.2	27.8	38.5	41.2	28.4	33.9	35.6	23.1	25.0	30.9
Papazoglou *et al* [22]	65.4	67.3	38.9	65.2	46.3	40.2	65.3	48.4	39.0	25.0	50.1
Zhang *et al* [37]	**75.8**	60.8	43.7	71.1	46.5	54.6	55.5	54.9	42.4	35.8	54.1
Rochan *et al* [27]	56.0	30.1	39.6	**85.7**	24.7	**87.8**	55.6	60.2	61.8	51.7	55.3
LSMWC (VGG-f)	48.5	**74.4**	52.8	61.6	**59.4**	69.3	**71.4**	68.5	73.6	43.0	62.3
LSMWC (VGG-16)	44.3	68.6	**56.7**	63.5	50.0	70.7	71.2	**75.9**	**73.8**	**55.5**	**63.0**

Table 4. Co-localization CorLoc (%) of different methods on the YouTube-Objects dataset.

about 570,000 frames with 1,407 annotations in the first version of the dataset [23]. According to our knowledge, it is the largest available video dataset with bounding-box annotations on multiple classes. The individual video frames after decompression are used in our experiments to avoid possible confusion when applying different video decoders. We only perform object co-localization on video frames with ground-truth annotations, following the practice in [15]. No additional spatial-temporal information is utilized in our method. The Youtube-Objects dataset comes with the test videos divided in 10 classes according to which dominant object is mostly present in them. Hence we construct 10 different graphs for this dataset. The co-localization accuracy of different methods on the YouTube-Objects dataset are summarized in Table 4. Among all the methods, [37, 27] also utilize deep networks for visual representation. The experiments justify that the proposed object co-localization framework is also very effective for mining common objects in videos.

7 Conclusion

In this paper, we present a novel framework to address the problem of object co-localization. It provides a practical and general solution for research and applications related to the MWC problem. Besides, deep learning based methods are utilized to localize the candidates of the common objects and describe their visual characteristics. This makes it possible to better discriminate the inter-class similarities and identify the intra-class variations. Finally, a cycle elimination based restart strategy is proposed to guide the local search for the MWC. It suc-

cessfully resolves the cycling issue in the optimization process. The experimental results on the object co-localization tasks demonstrate that our MWC solver is particularly suitable for graphs with high density. The proposed method shows significant improvements over several strong baselines.

Acknowledgements

This work was supported in part by NSF grants IIS-1814745 and IIS-1302164. Yi Fan is supported by the Natural Science Foundation of China (Nos. U1711263, U1811264, 61463044, 61572234, 61672441, 61603152) and the Shenzhen Basic Research Program (No. JCYJ20170818141325209). We also acknowledge donations of Titan X GPU cards by NVIDIA corporation.

References

1. Adamczewski, K., Suh, Y., Mu Lee, K.: Discrete tabu search for graph matching. In: ICCV. pp. 109–117 (2015)
2. Alidaee, B., Glover, F., Kochenberger, G., Wang, H.: Solving the maximum edge weight clique problem via unconstrained quadratic programming. European Journal of Operational Research **181**(2), 592–597 (2007)
3. Bilen, H., Pedersoli, M., Tuytelaars, T.: Weakly supervised object detection with convex clustering. In: CVPR. pp. 1081–1089 (2015)
4. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. In: BMVC (2014)
5. Chen, W., Chen, X., Zhang, J., Huang, K.: Beyond triplet loss: a deep quadruplet network for person re-identification. arXiv (2017)
6. Cho, M., Kwak, S., Schmid, C., Ponce, J.: Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In: CVPR. pp. 1201–1210 (2015)
7. Cinbis, R.G., Verbeek, J., Schmid, C.: Weakly supervised object localization with multi-fold multiple instance learning. TPAMI (2016)
8. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results (2007)
9. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results (2012)
10. Fan, Y., Li, C., Ma, Z., Wen, L., Sattar, A., Su, K.: Local search for maximum vertex weight clique on large sparse graphs with efficient data structures. In: Australasian Joint Conference on Artificial Intelligence. pp. 255–267 (2016)
11. Fan, Y., Li, N., Li, C., Ma, Z., Latecki, L.J., Su, K.: Restart and random walk in local search for maximum vertex weight cliques. In: IJCAI (2017)
12. Gouveia, L., Martins, P.: Solving the maximum edge-weight clique problem in sparse graphs with compact formulations. EURO Journal on Computational Optimization **3**(1), 1–30 (2015)
13. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: International Workshop on Similarity-Based Pattern Recognition. pp. 84–92 (2015)
14. Hosseinian, S., Fontes, D.B., Butenko, S.: A nonconvex quadratic optimization approach to the maximum edge weight clique problem. Journal of Global Optimization pp. 1–22 (2018)

15. Joulin, A., Tang, K., Fei-Fei, L.: Efficient image and video co-localization with frank-wolfe algorithm. In: ECCV. pp. 253–268 (2014)
16. Kalogeiton, V., Ferrari, V., Schmid, C.: Analysing domain shift factors between videos and images for object detection. TPAMI **38**(11), 2327–2334 (2016)
17. Kwak, S., Cho, M., Laptev, I., Ponce, J., Schmid, C.: Unsupervised object discovery and tracking in video collections. In: ICCV. pp. 3173–3181 (2015)
18. Li, Y., Liu, L., Shen, C., van den Hengel, A.: Image co-localization by mimicking a good detectors confidence score distribution. In: ECCV. pp. 19–34 (2016)
19. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755 (2014)
20. Ma, T., Latecki, L.J.: Maximum weight cliques with mutex constraints for video object segmentation. In: CVPR. pp. 670–677 (2012)
21. Mascia, F., Cilia, E., Brunato, M., Passerini, A.: Predicting structural and functional sites in proteins by searching for maximum-weight cliques. In: AAAI (2010)
22. Papazoglou, A., Ferrari, V.: Fast object segmentation in unconstrained video. In: ICCV. pp. 1777–1784 (2013)
23. Prest, A., Leistner, C., Civera, J., Schmid, C., Ferrari, V.: Learning object class detectors from weakly annotated video. In: CVPR. pp. 3282–3289 (2012)
24. Pullan, W.: Approximating the maximum vertex/edge weighted clique using local search. Journal of Heuristics **14**(2), 117–134 (2008)
25. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. TPAMI **39**(6), 1137–1149 (2017)
26. Ren, W., Huang, K., Tao, D., Tan, T.: Weakly supervised large scale object localization with multiple instance learning and bag splitting. TPAMI **38**(2), 405–416 (2016)
27. Rochan, M., Wang, Y.: Weakly supervised localization of novel objects using appearance transfer. In: CVPR. pp. 4315–4324 (2015)
28. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: ECCV. pp. 430–443 (2006)
29. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV **115**(3), 211–252 (2015)
30. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv (2014)
31. Tang, K., Joulin, A., Li, L.J., Fei-Fei, L.: Co-localization in real-world images. In: CVPR. pp. 1464–1471 (2014)
32. Wang, C., Zhang, H., Yang, L., Cao, X., Xiong, H.: Multiple semantic matching on augmented n-partite graph for object co-segmentation. TIP **26**(12), 5825–5839 (2017)
33. Wang, C., Ren, W., Huang, K., Tan, T.: Weakly supervised object localization with latent category learning. In: ECCV. pp. 431–445 (2014)
34. Wang, X., Zhu, Z., Yao, C., Bai, X.: Relaxed multiple-instance svm with application to object discovery. In: ICCV. pp. 1224–1232 (2015)
35. Wang, Y., Cai, S., Yin, M.: Two efficient local search algorithms for maximum weight clique problem. In: AAAI. pp. 805–811 (2016)
36. Wei, X.S., Zhang, C.L., Li, Y., Xie, C.W., Wu, J., Shen, C., Zhou, Z.H.: Deep descriptor transforming for image co-localization. arXiv (2017)
37. Zhang, Y., Chen, X., Li, J., Wang, C., Xia, C.: Semantic object segmentation via detection in weakly labeled video. In: CVPR. pp. 3641–3649 (2015)