Efficient Image and Video Co-localization with Frank-Wolfe Algorithm

Armand Joulin*, Kevin Tang*, and Li Fei-Fei

Computer Science Department, Stanford University

Abstract. In this paper, we tackle the problem of performing efficient co-localization in images and videos. Co-localization is the problem of simultaneously localizing (with bounding boxes) objects of the same class across a set of distinct images or videos. Building upon recent state-of-the-art methods, we show how we are able to naturally incorporate temporal terms and constraints for video co-localization into a quadratic programming framework. Furthermore, by leveraging the Frank-Wolfe algorithm (or conditional gradient), we show how our optimization formulations for both images and videos can be reduced to solving a succession of simple integer programs, leading to increased efficiency in both memory and speed. To validate our method, we present experimental results on the PASCAL VOC 2007 dataset for images and the YouTube-Objects dataset for videos, as well as a joint combination of the two.

1 Introduction

With the rising popularity of Internet photo and video sharing sites like Flickr and YouTube, there is a large amount of visual data uploaded to the Internet. In addition to pixels, these images and videos are often tagged with the visual concepts they contain, leading to a natural source of weakly labeled data. Recent research has studied ways of leveraging this data, such as weakly supervised localization [14], co-segmentation [21, 23, 38], and co-localization [37, 44].

In this paper, we address the problem of co-localization in images and videos. Co-localization is the problem of localizing (with bounding boxes) the common object in a set of images or videos. Recent work has studied co-localization in images with potentially noisy labels [44], and co-localization in videos [37] for learning object detectors. Building upon the success of a recent state-of-the-art method [44], we propose a formulation for co-localization in videos that can take advantage of temporal consistency with temporal terms and constraints, while still maintaining a standard quadratic programming formulation. We also show how we can combine both models to perform joint image-video co-localization, the logical way of utilizing all of the weakly supervised data we have available.

To efficiently perform co-localization in both images and videos, we show how our optimization problems can be reduced to a succession of simple integer problems using the Frank-Wolfe algorithm (also known as conditional gradient) [17].

* Indicates equal contribution.

D. Fleet et al. (Eds.): ECCV 2014, Part VI, LNCS 8694, pp. 253–268, 2014.

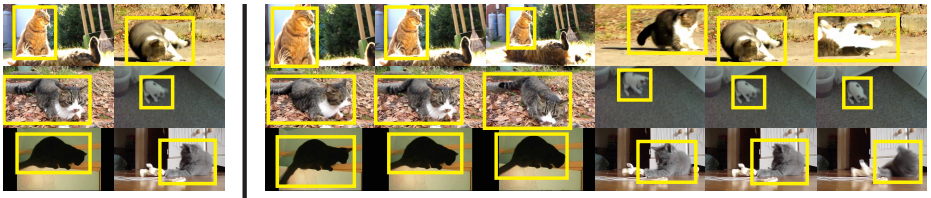

Fig. 1. In the co-localization problem, our goal is to simultaneously localize the common object of the same class in a set of images or videos. In videos, we have additional temporal consistency information we can leverage to make the problem easier.

For image co-localization, this results in simply taking the maximum of a set of values. For video co-localization, this results in the shortest path algorithm, which can be efficiently solved using dynamic programming.

To re-iterate, we make two key contributions in this paper.

- **Formulation for video co-localization.** We present a novel formulation for video co-localization, extending [44] with temporal terms and constraints.
- **Frank-Wolfe algorithm for efficient optimization.** We show how the Frank-Wolfe algorithm [17] can be used to efficiently solve our optimization problems by solving a succession of simple integer problems.

We present convincing experiments on two difficult datasets: PASCAL VOC 2007 for images [16], YouTube-Objects for videos [37]. We also show results for joint image-video co-localization by combining our models.

2 Related Work

The co-localization problem is similar to co-segmentation [21–24, 38, 39, 47] and weakly supervised localization [14, 31, 32, 43, 46]. Compared to co-segmentation, we seek to localize objects with bounding boxes rather than segmentations, which allows us to greatly decrease the number of variables in our problem. Compared to weakly supervised localization, we are more flexible because we do require any negative images for which we know do not contain our object.

Our work builds upon the formulation introduced in [44] for co-localization in images, which defines an optimization objective that draws inspiration from works in image segmentation [42] and discriminative clustering [3, 21, 49, 51]. Extending their work, we introduce a formulation for co-localization in videos that incorporates constraints and terms that capture temporal consistency, a key property in videos. We also show how the formulation in [44], as well as our video extension, are able to be efficiently solved using the Frank-Wolfe algorithm [17, 26]. Also similar is [14], which generates candidate bounding boxes and tries to select the correct box within each image. However, while they utilize a conditional random field, we adopt a quadratic programming formulation that can be relaxed and efficiently solved. Similar discrete optimization approaches have been shown to work well in various computer vision applications [6, 12, 13].

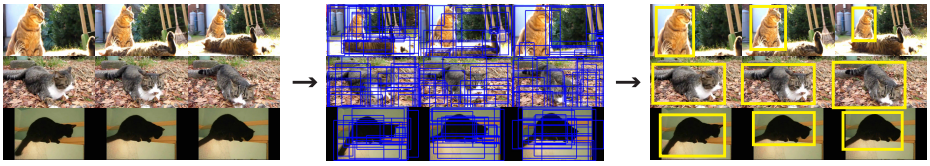

Fig. 2. Our co-localization approach starts by generating candidate bounding boxes for each image/video frame. We then jointly select the correct box in each image/video frame that contains the common object.

Our work is also closely related to Chari et al. [9] where they efficiently solve a quadratic program for multi-object tracking using the Frank-Wolfe algorithm.

For video co-localization, most similar is [37], which also tackles the problem of co-localization in videos by proposing candidate regions and selecting the correct one from each video. In [37], the authors try to leverage temporal information by proposing candidate tubes, which suffers from poor performance even with an optimal learning algorithm. In our formulation, we consider the temporal information directly in our model. Co-localization in video also shares similarities to co-segmentation in video, which has recently been studied in [10].

3 Our Approach

We start by briefly describing the co-localization model we use for images [44], and then show how it can be extended to videos. In both models, we take the approach of generating a set of candidate bounding boxes in each image/frame, and then formulating an optimization problem to jointly select the box from each image/frame that contains the common object, as shown in Figure 2.

3.1 Image Model

Given a set of n images $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$, our goal is to localize the common object in each image. Using objectness [1], we generate m candidate boxes for each image that could potentially contain an object, resulting in a set of boxes $\mathcal{B}_j$ for each image $I_j \in \mathcal{I}$. Our goal then is to jointly select the box from each image that contains the common object. To simplify notation, we define the set of all boxes as $\mathcal{B} = \mathcal{B}_1 \cup \mathcal{B}_2 \ldots \cup \mathcal{B}_n$ and $n_b = nm$ the total number of boxes.

Feature Representation. For each box $b_k \in \mathcal{B}$, we compute a feature representation of the box as $x_k \in \mathbb{R}^d$, and stack the feature vectors to form a feature matrix $X \in \mathbb{R}^{n_b \times d}$. We densely extract SIFT features [29] at every pixel and vector quantize each descriptor into a 1,000 word codebook. For each box, we pool the SIFT features within the box using 1×1 and 3×3 SPM pooling regions [28] to generate a $d = 10,000$ dimensional feature descriptor for each box.

Model Formulation. We associate with each box $b_{j,k} \in \mathcal{B}_j$ a binary label variable $z_{j,k}$, which is equal to 1 if $b_{j,k}$ contains the common object and 0 otherwise. We denote by z the n_b dimensional vector obtained by stacking the $z_{j,k}$.

Making the assumption that in each image there is only one box that contains the common object, we then solve the following optimization problem to select the best box from each image:

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & z^T(L + \mu A)z - z^T \lambda \log(m) \\ \text{subject to} \quad & z \in \{0,1\}, \forall I_j \in \mathcal{I} : \sum_{k=1}^{m} z_{j,k} = 1. \end{aligned} \tag{1}$$

The parameter μ controls the tradeoff between the quadratic terms, while the parameter λ controls the tradeoff between the linear and quadratic terms. The constraints enforce that only a single box is selected in each image. We briefly describe the terms in the objective below, but more details can be found in [44].

Box Prior. The vector m is a prior for each box computed from a saliency map [35] that represents our belief that a box contains the common object given only information within the image.

Box Similarity. The matrix $L = I - D^{-\frac{1}{2}} S D^{-\frac{1}{2}}$ is the normalized Laplacian matrix [42], where D is the diagonal matrix composed of the row sums of S, the $n_b \times n_b$ pairwise χ^2-similarity matrix computed from X. We set the similarity between boxes from the same image/video to 0. This matrix encourages boxes with similar appearances from different images/videos to have the same label.

Box Discriminability. The matrix $A = \frac{1}{n_b}(\Pi_{n_b}(I_{n_b} - X_{box}(X_{box}^T \Pi_{n_b} X_{box} + n_b \kappa I)^{-1} X_{box}^T)\Pi_{n_b})$ is the discriminative clustering term [3, 49], where $\Pi_{n_b} = I_{n_b} - \frac{1}{n_b} 1_{n_b} 1_{n_b}^T$ is the centering projection matrix. This term allows us to utilize a discriminative objective function to penalize the selection of boxes whose features are not easily linearly separable from other boxes. Note that since the matrices L and A are each positive semi-definite, the objective function is convex.

3.2 Video Model

Given a set of n videos $\mathcal{V} = \{V_1, V_2, \ldots, V_n\}$, our goal is to localize the common object in each frame of each video. We approach this problem by considering each video V_i as a collection of temporally ordered frames $\mathcal{I}_i = \{I_{i1}, I_{i2}, \ldots, I_{il_i}\}$, where l_i is the length of video V_i and I_{ij} corresponds to frame j of video V_i. Similar to the image model, we generate a set of m candidate boxes $\mathcal{B}_{ij}$ for each frame of each video using objectness [1]. Our goal then is to select the box from each frame that contains the common object. Similar to the image model, we associate with each box $b_{i,j,k} \in \mathcal{B}_{i,j}$ a binary label variable $z_{i,j,k}$, and stack the variables to obtain z, the $n_b = \sum_{i=1}^{n} l_i m$ dimensional vector.

Defining $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n\}$ as the set of all frames, we can apply the same objective function and constraints from the image model to $\mathcal{I}$. The image model constraints enforce selecting a single box in each frame, and the image model objective function captures the box prior, similarity, and discriminability within and across different videos.

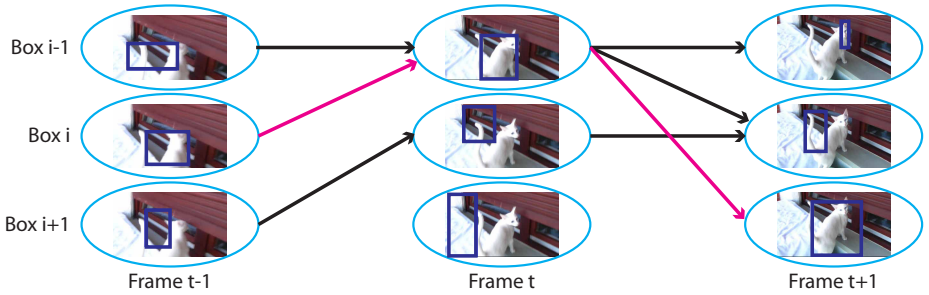

Fig. 3. Given consecutive frames of video, we build a graph between adjacent frames. Each node in the graph (blue circle) represents a candidate bounding box, and the directed edges between boxes are defined by a temporal similarity metric that measures how well the boxes agree in size and position. Note that some edges are removed, effectively limiting the possible paths through the graph from first to last frame. The magenta edges represent the optimal path through the frames.

Incorporating Temporal Consistency. In video data, temporal consistency tells us that between consecutive frames, it is unlikely for objects to undergo drastic changes in qualities such as appearance, position, and size. This is a powerful prior that is often leveraged in video tasks such as tracking [2, 4, 19, 33, 36, 45, 50]. In our framework, if two boxes from consecutive frames differ greatly in their size and position, it should be unlikely that they will be selected together. Using this intuition, we can define a simple temporal similarity measure between two boxes b_i and b_j from consecutive frames as follows:

$$s_{\text{temporal}}(b_i, b_j) = \exp\left(- \|b_i^{center} - b_j^{center}\|_2 - \left\| \frac{|b_i^{area} - b_j^{area}|}{\max(b_i^{area}, b_j^{area})} \right\|_2 \right), \quad (2)$$

where b_i^{area} is the pixel area of box b_i and b_i^{center} are the center coordinates of box b_i, normalized by the width and height of the frame.

With this similarity metric for all pairs of boxes between adjacent frames, we obtain a weighted graph G_i for each video V_i that connects the boxes within the video based on temporal similarity, as shown in Figure 3. We threshold small values of similarity so that dissimilar edges have a weight of 0 and are thus disconnected. Note that as long as we can obtain a weighted graph, any similarity metric between two boxes from adjacent frames can be used. This makes our temporal framework extremely flexible, and allows us to potentially leverage state-of-the-art methods in object tracking [2, 4, 19, 33, 36, 50].

We collect all the pairwise similarities s_{temporal} between boxes in adjacent frames into a similarity matrix S_t, where $S_t(i, j) = s_{\text{temporal}}(b_i, b_j)$ if b_i and b_j are boxes in adjacent frames, and $S_t(i, j) = 0$ otherwise. With this matrix, we can compute the normalized Laplacian $U = I - D^{-\frac{1}{2}} S_t D^{-\frac{1}{2}}$, where D is the diagonal matrix composed of the row sums of S_t. This matrix encourages us to select boxes that are similar based on the temporal similarity metric.

Intuitively, the boxes we select from each video V_i should respect the corresponding graph G_i, in that the solution should follow a valid path through the graph from the first frame to the last. For each edge (a, b) in the graph G_i, we define a binary variable $y_{i,a,b}$ equal to 1 if both a and b are boxes containing the object and 0 otherwise. More precisely, we require $y \in \{0, 1\}$ to follow the linear constraints for each video V_i and every box b_k (associated with binary label variable z_k) in V_i: $z_k = \sum_{l \in p(k)} y_{i,l,k} = \sum_{l \in c(k)} y_{i,k,l}$, where $p(k)$ and $c(k)$ are the parents and children of box b_k in the graph G_i, respectively.

Model Formulation. Combining the temporal terms and constraints together with the original image model, we obtain the following optimization problem to select the box containing the common object from each frame of video:

$$\begin{aligned} \underset{z,y}{\text{minimize}} \quad & z^T(L + \mu A + \mu_t U)z - z^T \lambda \log(m) \\ \text{subject to} \quad & z \in \{0,1\},\ y \in \{0,1\}, \\ & \forall I_j \in \mathcal{I} : \sum_{k=1}^{m} z_{j,k} = 1, \\ & \forall V_i \in \mathcal{V},\ \forall k \in V_i,\ z_k = \sum_{l \in p(k)} y_{i,l,k} = \sum_{l \in c(k)} y_{i,k,l}, \end{aligned} \tag{3}$$

where z_i are the binary label variables associated with the boxes in video V_i, and μ_t weights the temporal Laplacian matrix. The additional constraint forces us to choose solutions that respect the edges defined by the underlying graphs for each video, and the additional Laplacian term in the objective function weights these edges. Note that the additional constraint is required to constrain our solutions, as the terms in the objective from the image model can still lead us to select invalid paths if we only had the temporal Laplacian matrix. This formulation allows us to incorporate temporal consistency into the image model. In the rest of this paper, we denote by $\mathcal{P}$ the set of constraints defined in Eq. (3).

In the next section, we present a tight convex relaxation which can be efficiently optimized using the Frank-Wolfe algorithm [17].

4 Optimization

A standard way of dealing with quadratic programs such as Eq. 3 is to relax the discrete non-convex set $\mathcal{P}$ to its convex hull, $\text{conv}(\mathcal{P})$. Standard algorithms such as interior point methods can be applied but leads to a complexity of $O(N^3)$ which cannot deal with hundreds of videos. We show how it is possible to design an efficient algorithm by using the specificities of our problem.

A key observation towards designing an efficient algorithm for our problem is that the constraints defining the set $\mathcal{P}$ are separable in each video and are equivalent, for each video, to the constraints used in the shortest-path algorithm. This means that if our cost function was linear, we could solve our problem efficiently using dynamic programming.

4.1 Frank-Wolfe Algorithm

Given a convex cost function f and a convex set $\mathcal{D}$, the Frank-Wolfe algorithm [17] finds the global minimum of f over $\mathcal{D}$ by solving a succession of linear problems [15, 20]. More precisely, at each iteration k it solves:

$$\begin{aligned} \underset{y}{\text{minimize}} \quad & y^T \nabla f(z_{k-1}) \\ \text{subject to} \quad & y \in \mathcal{D}. \end{aligned} \tag{4}$$

The solution y_k is then used in Frank-Wolfe updates given by:

$$z_k = z_{k-1} + \lambda(y_k - z_{k-1}), \tag{5}$$

where $\lambda > 0$ is found using a line search (see Algorithm 1 for details). Essentially, the Frank-Wolfe algorithm considers a linear approximation of the objective function at each iteration. Although not appropriate for all convex optimization problems, Frank-Wolfe applied to our optimization formulations results in very simple linearizations with integer solutions that are easily solved.

Frank-Wolfe Algorithm on Convex Hull. This algorithm does not need an explicit form for $\mathcal{D}$ as long as it is possible to find the solution of a linear program over $\mathcal{D}$. This is particularly interesting when $\mathcal{D}$ is the convex hull of a set of points $\mathcal{C}$ on which it is possible to solve an integer program. Solving a linear program on $\mathcal{D}$ is then equivalent to solving an integer program over $\mathcal{C}$. This is a particularity of the Frank-Wolfe algorithm that we will exploit in our video setting.

Video Model. For the video model, the Frank-Wolfe algorithm solves the following problem at each iteration:

$$\begin{aligned} \underset{y}{\text{minimize}} \quad & y^T H z_{k-1} \\ \text{subject to} \quad & y \in \text{conv}(\mathcal{P}). \end{aligned} \tag{6}$$

where $H = L + \mu A + \mu_t U$. The cost function and constraints are separable for each video, and optimizing Eq. (6) results in the standard shortest path problem (see supplementary material for details) for each video, which can be solved efficiently using dynamic programming.

Image Model. For the image model, the linearized cost function is separable for each image, and we can efficiently find the best integer solution for this problem by computing the score for each box, $(L + \mu A)z_{k-1}$, and then simply selecting the argmin. Note that there is a trade-off between this algorithm and projected gradient descent in the case of images. While projected gradient descent requires less iterations to converge, each iteration requires a projection over the simplex, which is $O(N \log N)$ whereas each of our updates is only $O(N)$ (or folded into the gradient computation with almost no running time cost).

Since the Frank-Wolfe algorithm for images utilizes the same framework as for videos, an additional advantage is that we can easily learn a shared image/video model with a single algorithm.

Data: $y_0 \in \mathcal{D}$, $\varepsilon > 0$
Result: y^*
Initialization: $k = 0$, $z = y_0$, $\mathcal{S}_0 = \{y_0\}$, $\alpha^0 = \{1\}$;
while *duality_gap*$(z) \geq \varepsilon$ **do**
  $k \leftarrow k + 1$;
  $y_k \leftarrow \text{argmin}_{y \in \mathcal{D}} \langle y, \nabla f(z) \rangle$ (FW direction);
  $x_k \leftarrow \text{argmax}_{y \in \mathcal{S}_{k-1}} \langle y, \nabla f(z) \rangle$ (away direction);
  if $\langle y_k - z, \nabla f(z) \rangle \leq \langle z - x_k, \nabla f(z) \rangle$ **then**
    $d_k = y_k - z$;
    $\gamma_{max} = 1$;
  else
    $d_k = z - x_k$;
    $\gamma_{max} = \alpha_k(x_k)$;
  end
  Line search: $\gamma_k = \min_{\gamma \in [0, \gamma_{max}]} f(z + \gamma d_k)$;
  $\mathcal{S}_k$, $\alpha_k \leftarrow$ *update_active_set*(d_k, γ_k);
  Update $z \leftarrow z + \gamma_k d_k$;
  if $f(y_k) < f(y^*)$ **then**
    $y^* \leftarrow y_k$ (rounding 1);
  end
end
$y_r \leftarrow \text{argmax}_{y \in \mathcal{D}} \langle y, z \rangle$ (rounding 2);
if $f(y_r) < f(y^*)$ **then**
  $y^* \leftarrow y_r$ (combining rounding);
end

Algorithm 1. Frank-Wolfe algorithm with away step and rounding

4.2 Implementation Details

In this section, we present some details on our implementation of the Frank-Wolfe algorithm.

Away Step. We use an accelerated version termed Frank-Wolfe with away step [48]. The details of this algorithm are given in Algorithm 1. The algorithm keeps a set of previously seen integer solutions (called active corners) $\mathcal{S}_k$ at each iteration such that the current update z is the sum of the corners in $\mathcal{S}_k$ re-weighted by α_k. The set $\mathcal{S}_k$ is used to find potentially better directions by moving "away" from an active corner (away step). This version of Frank-Wolfe has been shown to have better convergence rates [18, 25]. The definition of the function *update_active_set* from Algorithm 1 is given in the supplementary material.

Line Seach and Duality Gap. In the case of a quadratic function, both the line search and the duality gap are in closed form (see supplementary material), which significantly improves the speed of our algorithm [26].

Parallel Computation. Our constraints are separable for each image and video, allowing efficient parallel computation of the update. Note that this is a property of any first-order method, including the Frank-Wolfe algorithm. In

practice, this allows us to be extremely memory efficient, as we can consider subproblems for each image or video separately.

Rounding. A typical concern with methods based on convex relaxations is obtaining a solution from the relaxed problem that satisfies the non-convex constraints from the original problem. In our case, the rounded solution must belong to the set $\mathcal{P}$. The most natural way of rounding a solution z is to find the closest element in $\mathcal{P}$ given some distance. For the ℓ_2 distance, this means solving $\min_{y \in \mathcal{P}} \|y - z\|_2^2$ which is not possible in general. However, in our case, since the ℓ_2 norm is constant on $\mathcal{P}$ (and equal to the total number of frames/images in the dataset), this projection is equivalent to:

$$\underset{y \in \mathcal{P}}{\text{maximize}} \quad \langle y, z \rangle, \tag{7}$$

which can be solved efficiently using the shortest-path algorithm for the video model, and simply taking the argmax in each image for the image model.

Additionally, the particular form of the Frank-Wolfe updates offers another very natural and inexpensive way of rounding our solution. We can keep track of the solution to the linear problem defined in Eq. (6) that minimizes the cost function defined in Eq. (3). Since this solution is in the original set $\mathcal{P}$, it automatically satisfies the constraints.

In practice, we use both rounding methods and keep the one that results in the lowest value of our cost function, as shown in Algorithm 1.

5 Results

We perform experiments on two challenging datasets, the PASCAL VOC 2007 dataset [16] and the YouTube-Objects dataset [37]. We also combine the two and present results for joint image-video co-localization. Following previous works in weakly supervised localization [14] and co-localization [44], we use the CorLoc evaluation metric, defined as the percentage of images correctly localized according to the PASCAL-criterion: $\frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} > 0.5$, where B_p is the predicted box and B_{gt} is the ground-truth box. All CorLoc results are given in percentages.

Implementation Details. We set the parameters of our method by optimizing over a small set of images/videos. For the image model, we set $\mu = 0.4$ and for the video model, we set $\mu = 0.6$ and $\mu_t = 1.8$. For both models, we found $\lambda = 0.1$ to perform best. Unless otherwise stated, we extracted 20 objectness boxes from each image. For the video model, we sampled each video every 10 frames, since there is typically little change in such a short amount of time.

5.1 Running Time and Rounding Experiments

In this section, we evaluate the running time of our algorithm. Our implementation is coded in MATLAB and we compare to two standard Quadratic Programming (QP) solvers, Mosek and Gurobi, which are coded in C++. All experiments are done on a single core 2.66GHz Intel CPU with 6GB of RAM.

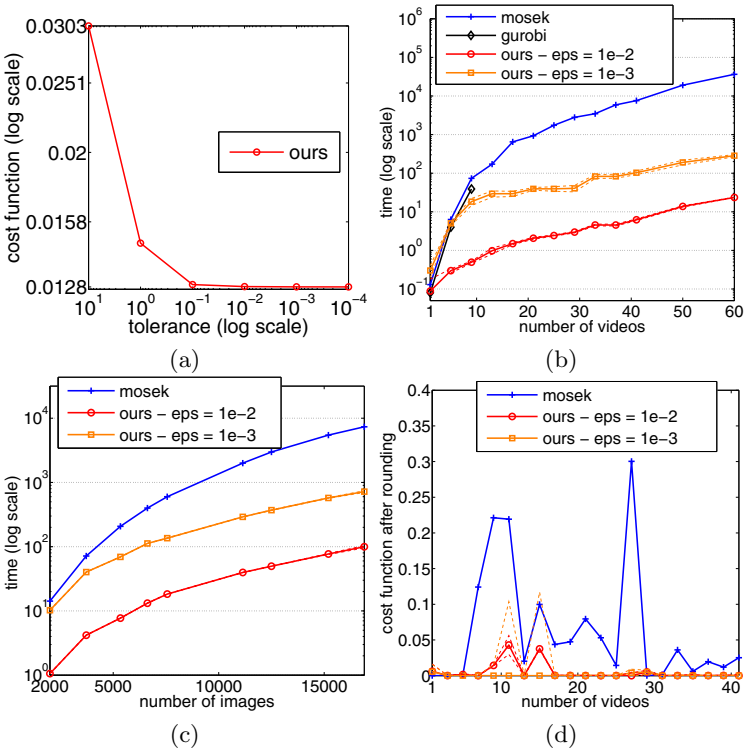

Fig. 4. (a) Value of the cost function for 11 videos as a function of the relative duality gap (log scale). Time comparison between our algorithm and standard QP solvers (time in log scale of second) for (b) video co-localization and (c) image co-localization. (d) Comparison of the value of the cost function obtained with our algorithm and a standard QP solver.

Stopping Criterion. Our stopping criterion is based on the relative duality gap defined as $d = (f - g)/g$, where f is our cost function and g is its dual (see supplementary material for more details). We stop when d is less than some $\varepsilon > 0$. We consider two values for ε, 10^{-2} and 10^{-3}. The choice of these values for ε is motivated by the empirical observation that our cost function remains almost constant for $d < 10^{-2}$, as show in 4(a).

Running Time Analysis. In 4(b)(c), we show how our algorithm scales in the number of videos and images compared to standard QP solvers. For fair running time comparison, we present the time for both standard QP solvers to reach a duality gap less than $\varepsilon = 10^{-2}$. When $\varepsilon = 10^{-3}$, our algorithm runs 100 times faster than standard solvers for more than 20 videos. For $\varepsilon = 10^{-2}$, this factor increases to more than 1000. Typically, for $\varepsilon = 10^{-3}$, solving our problem with 50 videos takes 3 minutes, and 80 videos takes 7 minutes. The gain in speed is mostly due to efficiently computed iterations based on a shortest path algorithm/argmin.

Table 1. Average CorLoc results and upper bound on PASCAL07

Number of objectness boxes [1]	5	10	15	20
Our method	23.96	24.23	24.28	**24.59**
Upper bound	51.04	62.22	67.99	**71.58**

Table 2. CorLoc results on PASCAL07-all compared to previous methods

Method	[40](w/ viewpoint)	[11](w/ viewpoint)	[14](w/ viewpoint)	Our method(w/o viewpoint)
Average CorLoc	14	19	23	22

Rounding Quality. In 4(d), we also compare the quality of the solution obtained after rounding in terms of the original cost function. We compare the relative value of the cost function, $(f - f^*)/f^*$, where f^* is the minimum observed value of the cost function. We round the solutions by solving Eq. (7). For fairness of comparison, we use the solution given by the QP solver for a tolerance of $\varepsilon = 10^{-10}$. Compared to the standard QP solver, our algorithm obtains a significantly better rounded solution in terms of value of the cost function.

Despite numerous advantages of our solver for our specific problem, a limitation of the Frank-Wolfe algorithm with away step in the case of an exponential number of corner points (as is the case in our problem) is that it converges with no guarantee of a linear convergence rate.

5.2 Image Co-localization: PASCAL VOC 2007

In [44], the authors show improved co-localization performance on PASCAL07-6x2, a small subset of PASCAL VOC 2007 divided into specific viewpoints. To illustrate the benefits of the Frank-Wolfe algorithm, which allows us to efficiently consider many more images and boxes per image, we co-localize all images not labeled as difficult for all classes in the PASCAL VOC 2007 dataset [16], which we denote as PASCAL07. This makes the problem much more difficult as we now have to co-localize differing viewpoints together and a much larger set of images. To emphasize the difference in size, the "bicycle-right" class in the PASCAL07-6x2 dataset has the largest number of images at 50, whereas the "person" class in the PASCAL07 dataset has 2,008 images. In all experiments performed in [44], the authors only co-localize a maximum of 100 images at a time due to efficiency concerns. Results for our method varying the number of extracted candidate boxes are given in Table 1, and visualizations are shown in Figure 5. We also show the upper bound on the performance that can be achieved with the candidate boxes, computed by selecting the box in each image with the highest CorLoc.

Number of Candidate Boxes. As we can see, the performance of our model increases when we increase the number of candidate boxes. We can also see that the upper bound becomes much better due to the better recall obtained with more boxes. This helps to validate the importance of efficient methods for co-localization, as they allow us to take advantage of more data in our model.

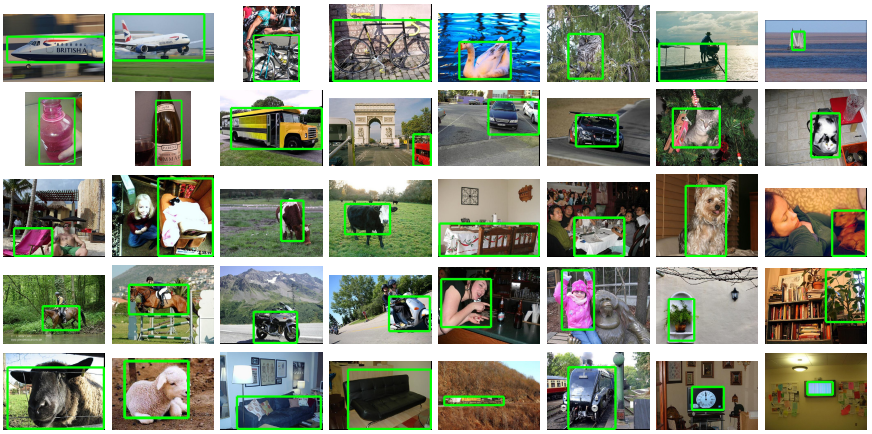

Fig. 5. Example co-localization results on PASCAL07. From left-right, every two images belong to the same class. Note that we can see a wide variety of viewpoints because we consider all images jointly.

Table 3. CorLoc results for video co-localization on the YouTube-Objects dataset

Method	aeroplane	bird	boat	car	cat	cow	dog	horse	motorbike	train	Average
[37]	**51.7**	17.5	**34.4**	34.7	22.3	17.9	13.5	26.7	**41.2**	25.0	28.5
Our method (image)	18.36	19.35	28.57	32.97	32.77	25.68	38.26	30.14	15.38	21.43	26.29
Our method (image) w/ smoothing	21.26	21.51	30.95	36.26	35.29	25.68	38.26	35.62	15.38	23.21	28.34
Our method (video)	25.12	**31.18**	27.78	**38.46**	**41.18**	**28.38**	33.91	35.62	23.08	25.00	**30.97**
[37] - Upper bound	53.9	19.6	38.2	37.8	32.2	21.8	27.0	34.7	45.4	37.5	34.8
Our method - Upper bound	**95.17**	**70.97**	**91.27**	**93.41**	**73.11**	**89.19**	**80.00**	**64.38**	**83.52**	**76.79**	**81.78**

Comparisons to Previous Methods. We also show results compared to state-of-the-art co-localization methods in Table 2 for the PASCAL07-all dataset [14], which does not consider the "bird", "car", "cat", "cow", "dog", and "sheep" classes. Note that all previous methods utilize additional viewpoint annotations by dividing the images for each class into separate viewpoints, and co-localizing each viewpoint separately with viewpoint-specific priors. On the other hand, our method is run on all of the viewpoints simultaneously, which is a much more difficult problem. Even in this more difficult scenario, we are able to obtain comparable results to previous methods.

5.3 Video Co-localization: YouTube-Objects

The YouTube-Objects dataset [37] consists of YouTube videos collected for 10 classes from PASCAL [16]: "aeroplane", "bird", "boat", "car", "cat", "cow", "dog", "horse", "motorbike", "train". For each class, bounding box annotations for the object are annotated in one frame per shot for 100-290 different shots. We perform video co-localization on all shots with annotations. Results are given in Table 3, where we compare to the co-localization method of [37], our image model with and without smoothing, as well as the upper bounds that can be

Fig. 6. Example co-localization results on YouTube-Objects for our video model (green boxes) and our image model (red boxes). Each column corresponds to a different class, and consists of frame samples from a single video.

obtained using both our methods for candidate box generation. Note that better results are obtained in [34] using unsupervised motion segmentation, which works particularly well for this dataset where objects of interest are moving. In contrast, our method focuses on trying to leverage appearance information across different videos in conjunction with temporal consistency. Thus, we believe that our approaches are orthogonal but complementary, and will likely result in even better performance if combined.

Comparisons to [37]. From our results, we see that we outperform the previous method of [37] for most classes. In addition, we see that the upper bound we can achieve with objectness boxes compared to the candidate tube generation of [7, 37] is much better for all classes. This is likely because object proposals in the image domain have received a great deal of attention and study [1, 8, 30, 41] compared to video [5, 7]. For the "aeroplane" and "motorbike" classes however, we perform much worse. This is likely because the candidate tube extraction algorithm used in [37] is able to effectively track simple and non-deformable objects. However, note that our method is actually agnostic to the underlying candidate region generation algorithm, and we could easily replace our objectness boxes with candidate tubes.

Comparisons to Image Model. Our video model outperforms the image model, which illustrates the importance of leveraging temporal consistency. From the visualizations in Figure 6, we see that the image model often jumps around throughout a single video. For the "dog" class however, our image model actually performs much better than our video model. This is likely due to large amounts of sporadic movement in the "dog" videos caused by both camera movement and object movement. The simple similarity metric we use for temporal consistency may not be invariant to such difficult types of motion, and thus the image model is able to perform better in this case. As noted previously, we can substitute any similarity metric into our framework, and thus potentially take advantage of methods in object tracking [2, 4, 19, 33, 36, 50] to further improve performance.

Table 4. CorLoc results for joint image-video co-localization on YouTube-Objects

Method	aeroplane	bird	boat	car	cat	cow	dog	horse	motorbike	train	Average
Video only	25.12	31.18	27.78	**38.46**	41.18	28.38	33.91	35.62	**23.08**	25.00	30.97
Joint Image+Video	**27.54**	**33.33**	27.78	34.07	**42.02**	28.38	**35.65**	35.62	21.98	25.00	**31.14**

5.4 Joint Image-Video Co-localization

Since the classes in the YouTube-Objects dataset are a subset of the PASCAL07 classes, we can combine the images from the corresponding classes in PASCAL07 with the videos in YouTube-Objects to perform joint image-video co-localization. Results for CorLoc performance on the YouTube-Objects dataset are given in Table 4. We can see that our performance increases slightly for several classes, such as "aeroplane", "bird", "cat" and "dog". It is not unexpected that performance becomes worse for several classes, as there is an inherent domain adaptation problem between images and videos [37, 45]. However, our preliminary results show that with efficient algorithms for image and video co-localization, the problem of jointly considering the two domains is viable, and may present an effective way of taking advantage of all the weakly labeled data available.

6 Conclusions

In this paper, we introduce a formulation for video co-localization that is able to naturally incorporate temporal consistency in a quadratic programming framework. In addition, we show how the image and video co-localization models that are presented can be efficiently optimized using the Frank-Wolfe algorithm. Our experiments on the PASCAL07 and YouTube-Objects datasets illustrate the benefits of our approach for image, video, and joint image-video co-localization.

For future work, we would like to consider jointly performing domain adaption to address the joint image-video co-localization problem with dimensionality reduction techniques [27]. It would also be interesting to consider ways of handling multiple objects per image/frame, multiple object classes, and occlusions.

Acknowledgments. We especially thank Simon Lacoste-Julien for his helpful comments on many technical aspects of this paper. We also thank Vignesh Ramanathan for helpful comments and suggestions. This research is partially supported by an ONR MURI grant, the DARPA Mind's Eye grant, and a NSF GRFP under grant no. DGE-114747 (to K.T.).

References

1. Alexe, B., Deselaers, T., Ferrari, V.: Measuring the objectness of image windows. IEEE T-PAMI 34(11), 2189–2202 (2012)
2. Babenko, B., Yang, M.H., Belongie, S.: Robust object tracking with online multiple instance learning. IEEE T-PAMI 33(8), 1619–1632 (2011)
3. Bach, F., Harchaoui, Z.: Diffrac: a discriminative and flexible framework for clustering. In: NIPS (2007)

4. Berclaz, J., Fleuret, F., Türetken, E., Fua, P.: Multiple object tracking using k-shortest paths optimization. IEEE T-PAMI 33(9), 1806–1819 (2011)
5. Bergh, M.V.D., Roig, G., Boix, X., Manen, S., Gool, L.V.: Online video seeds for temporal window objectness. In: ICCV (2013)
6. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE T-PAMI 23(11), 1222–1239 (2001)
7. Brox, T., Malik, J.: Object segmentation by long term analysis of point trajectories. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 282–295. Springer, Heidelberg (2010)
8. Carreira, J., Sminchisescu, C.: Constrained parametric min-cuts for automatic object segmentation. In: CVPR (2010)
9. Chari, V., Lacoste-Julien, S., Sivic, J., Laptev, I.: On pairwise cost for multi-object network flow tracking. Tech. rep., arXiv (2014)
10. Chiu, W.C., Fritz, M.: Multi-class video co-segmentation with a generative multi-video model. In: CVPR (2013)
11. Chum, O., Zisserman, A.: An exemplar model for learning object classes. In: CVPR (2007)
12. Delong, A., Gorelick, L., Veksler, O., Boykov, Y.: Minimizing energies with hierarchical costs. IJCV 100(1), 38–58 (2012)
13. Delong, A., Osokin, A., Isack, H.N., Boykov, Y.: Fast approximate energy minimization with label costs. IJCV 96(1), 1–27 (2012)
14. Deselaers, T., Alexe, B., Ferrari, V.: Weakly supervised localization and learning with generic knowledge. IJCV 100(3), 275–293 (2012)
15. Dunn, J.C.: Convergence rates for conditional gradient sequences generated by implicit step length rules. SIAM Journal on Control and Optimization 18(5), 473–487 (1980)
16. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC 2007) Results (2007)
17. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval Research Logistics Quarterly 3(1-2), 95–110 (1956)
18. Guelat, J., Marcotte, P.: Some comments on wolfe's away step. Mathematical Programming 35(1), 110–119 (1986)
19. Hare, S., Saffari, A., Torr, P.H.S.: Struck: Structured output tracking with kernels. In: ICCV (2011)
20. Jaggi, M.: Revisiting frank-wolfe: Projection-free sparse convex optimization. In: ICML, pp. 427–435 (2013)
21. Joulin, A., Bach, F., Ponce, J.: Discriminative clustering for image co-segmentation. In: CVPR (2010)
22. Joulin, A., Bach, F., Ponce, J.: Multi-class cosegmentation. In: CVPR (2012)
23. Kim, G., Xing, E.P., Fei-Fei, L., Kanade, T.: Distributed cosegmentation via submodular optimization on anisotropic diffusion. In: ICCV (2011)
24. Kuettel, D., Guillaumin, M., Ferrari, V.: Segmentation propagation in imagenet. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VII. LNCS, vol. 7578, pp. 459–473. Springer, Heidelberg (2012)
25. Lacoste-Julien, S., Jaggi, M.: An affine invariant linear convergence analysis for frank-wolfe algorithms. arXiv preprint arXiv:1312.7864 (2013)
26. Lacoste-Julien, S., Jaggi, M., Schmidt, M., Pletscher, P.: Block-coordinate frank-wolfe optimization for structural svms. In: ICML, vol. 28, pp. 1438–1444 (2012)
27. Lampert, C.H., Krömer, O.: Weakly-paired maximum covariance analysis for multimodal dimensionality reduction and transfer learning. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 566–579. Springer, Heidelberg (2010)

28. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR (2006)
29. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV 60(2), 91–110 (2004)
30. Manén, S., Guillaumin, M., Van Gool, L.: Prime Object Proposals with Randomized Prim's Algorithm. In: ICCV (2013)
31. Nguyen, M.H., Torresani, L., de la Torre, F., Rother, C.: Weakly supervised discriminative localization and classification: a joint learning process. In: ICCV (2009)
32. Pandey, M., Lazebnik, S.: Scene recognition and weakly supervised object localization with deformable part-based models. In: ICCV (2011)
33. Pang, Y., Ling, H.: Finding the best from the second bests - inhibiting subjective bias in evaluation of visual tracking algorithms. In: ICCV (2013)
34. Papazoglou, A., Ferrari, V.: Fast object segmentation in unconstrained video. In: ICCV (2013)
35. Perazzi, F., Krähenbühl, P., Pritch, Y., Hornung, A.: Saliency filters: Contrast based filtering for salient region detection. In: CVPR (2012)
36. Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color-based probabilistic tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002, Part I. LNCS, vol. 2350, pp. 661–675. Springer, Heidelberg (2002)
37. Prest, A., Leistner, C., Civera, J., Schmid, C., Ferrari, V.: Learning object class detectors from weakly annotated video. In: CVPR (2012)
38. Rubinstein, M., Joulin, A., Kopf, J., Liu, C.: Unsupervised joint object discovery and segmentation in internet images. In: CVPR (2013)
39. Rubio, J.C., Serrat, J., López, A.: Video co-segmentation. In: Lee, K.M., Matsushita, Y., Rehg, J.M., Hu, Z. (eds.) ACCV 2012, Part II. LNCS, vol. 7725, pp. 13–24. Springer, Heidelberg (2013)
40. Russell, B.C., Efros, A.A., Sivic, J., Freeman, W.T., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: CVPR (2006)
41. van de Sande, K.E.A., Uijlings, J.R.R., Gevers, T., Smeulders, A.W.M.: Segmentation as selective search for object recognition. In: ICCV (2011)
42. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE T-PAMI 22(8), 888–905 (2000)
43. Siva, P., Russell, C., Xiang, T., de Agapito, L.: Looking beyond the image: Unsupervised learning for object saliency and detection. In: CVPR (2013)
44. Tang, K., Joulin, A., Li, L.J., Fei-Fei, L.: Co-localization in real-world images. In: CVPR (2014)
45. Tang, K., Ramanathan, V., Fei-Fei, L., Koller, D.: Shifting weights: Adapting object detectors from image to video. In: NIPS (2012)
46. Tang, K., Sukthankar, R., Yagnik, J., Fei-Fei, L.: Discriminative segment annotation in weakly labeled video. In: CVPR (2013)
47. Vicente, S., Rother, C., Kolmogorov, V.: Object cosegmentation. In: CVPR (2011)
48. Wolfe, P.: Convergence theory in nonlinear programming. In: Integer and Nonlinear Programming, pp. 1–36 (1970)
49. Xu, L., Neufeld, J., Larson, B., Schuurmans, D.: Maximum margin clustering. In: NIPS (2004)
50. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Comput. Surv. 38(4) (2006)
51. Zhou, G.T., Lan, T., Vahdat, A., Mori, G.: Latent maximum margin clustering. In: NIPS (2013)