[Supplementary Material]
On Multiple Foreground Cosegmentation

Gunhee Kim Eric P. Xing
School of Computer Science, Carnegie Mellon University
{gunhee,epxing}@cs.cmu.edu

1. The Algorithmic Details of Tractable MFC

In this section, we first present the algorithm that infers the most probable tree $\mathcal{T}_i^*$ from the generated foreground candidates $\mathcal{B}_i$. Then, we discuss the dynamic programming based search algorithm to obtain the optimal solution to Eq.(1) in the main draft. Refer to Section 3.3 of the main draft for the background of discussion in this section.

1.1. Inferring the Tree from Candidate Set

Given candidate set $\mathcal{B}_i$ (*i.e.* a set of subtrees) of image i, our objective here is to infer the most probable tree $\mathcal{T}_i^*$. It can be formulated as the following maximum likelihood estimation (MLE) in a similar way to tree structure learning (*e.g.* Chow-Liu tree [1]).

$$\mathcal{T}_i^* = \underset{\mathcal{T}\in\mathcal{T}(\mathcal{G}_i)}{\operatorname{argmax}} P(\mathcal{B}_i|\mathcal{T}) \tag{1.1}$$

where $P(\mathcal{B}_i|\mathcal{T})$ is the data likelihood of the given $\mathcal{B}_i$ and $\mathcal{T}(\mathcal{G}_i)$ is the set of all possible spanning trees on $\mathcal{G}_i$. The probability of a candidate set $\mathcal{B}_i$ in tree $\mathcal{T}$ is

$$P(\mathcal{B}_i|\mathcal{T}) = \prod_{B_l\in\mathcal{B}_i} P(B_l|\mathcal{T}) \tag{1.2}$$

where we assume the conditional independence between candidates given the tree $\mathcal{T}$. The probability of observing a candidate in a particular tree structure is defined as:

$$P(B_l|\mathcal{T}) = \prod_{(u,v)\in\mathcal{T}} \exp\left(P_{B_l}(u,v)\right) \tag{1.3}$$

$$= \prod_{(u,v)\in\mathcal{T}} \exp\left(P(u,v)\cdot\delta((u,v)\in B_l)\right) \tag{1.4}$$

where the $P(u,v)$ is the probability of an edge between u and v and $\delta((u,v)\in B_l)$ is an indicator whether the edge (u,v) is in the B_l or not. Hence, from eq.(1.2) and eq.(1.4),

Algorithm 1: Infer the most probable $\mathcal{T}_i^*$ from $\mathcal{B}_i$

Input: (1) Candidate set $\mathcal{B}_i$ ($B_l = \langle k_l, C_l, w_l\rangle$ where C_l is a subtree of $\mathcal{G}_i$ and w_l is the value to its foreground).
Output: (1) Candidate tree $\mathcal{T}_i^*$ and (2) Pruned $\mathcal{B}_i^*$ ($\subset \mathcal{B}_i$).

1: Set $\mathbf{A}$ be an $N\times N$ zero matrix where $N=|\mathcal{S}_i|$. Set $\mathcal{B}_i^*\leftarrow\emptyset$.
foreach $B_l = \langle k_l, C_l, w_l\rangle \in \mathcal{B}_i$ **do**
  foreach $s \in \mathcal{C}_l$ **do**
    foreach $t \in \mathcal{C}_l, t \neq s$ **do** $\mathbf{A}(s,t) \leftarrow \mathbf{A}(s,t) + w_l$ **end**
2: Let $\mathcal{T}_i^*$ be the maximum spanning tree of $\mathbf{A}$.
foreach $B_l = \langle k_l, C_l, w_l\rangle \in \mathcal{B}_i$ **do**
  if all edges $(u,v) \in C_l$ is in $\mathcal{T}_i^*$ **then** $\mathcal{B}_i^* \leftarrow B_l$. **end**

the log-likelihood $\mathcal{L}$ is defined as follows.

$$\mathcal{L} = \sum_{B_l\in\mathcal{B}_i} \log P(B_l|\mathcal{T}) \tag{1.5}$$

$$= \sum_{B\in\mathcal{B}_i} \sum_{(u,v)\in\mathcal{T}} \delta((u,v)\in B_l)\log P(u,v) \tag{1.6}$$

Note that the sample proportions are the maximum likelihood estimates of the parameters of discrete distributions.

$$\widehat{P}_{ML}(u,v) = \widetilde{P}(u,v) \propto \exp\left(\frac{\sum_{B_l\in\mathcal{B}_i} w_l\delta((u,v)\in B_l)}{\sum_{B_l\in\mathcal{B}_i} w_l}\right) \tag{1.7}$$

Therefore, from eq.(1.6) and eq.(1.7), the maximum likelihood is as follows by maximizing over the parameters for a fixed structure:

$$\mathcal{L}^* = \sum_{(u,v)\in\mathcal{T}} \sum_{B\in\mathcal{B}_i} \delta((u,v)\in B_l)\log\widetilde{P}(u,v) \tag{1.8}$$

$$\propto \sum_{(u,v)\in\mathcal{T}} \sum_{B_l\in\mathcal{B}_i} w_l\delta((u,v)\in B_l)$$

As shown in Eq.(1.8), the likelihood of tree is the sum of the weight values associated with the edges of each candidate $B_l \in \mathcal{B}_i$. According to [1], we see that the maximum likelihood tree is a maximal spanning tree (MST). The above whole steps are summarized in Algorithm 1, which

Algorithm 2: Solve region assignment (Eq.(1)) from $\mathcal{B}_i^*$

Input: (1) The pruned candidate set $\mathcal{B}_i^*$ (for each $B_l = \langle k_l, C_l, w_l \rangle$ where k_l is the index of the foreground, C_l is a subtree of $\mathcal{G}_i$ and w_l is the value to its foreground). (2) T_i^*.

Output: (1) Foreground assignment $\{\mathcal{F}_1^k, \cdots, \mathcal{F}_1^{K+1}\}$.

1: Randomly choose the root r of the tree T_i^*.
2: Assign each node of T_i^* a *level*, which is its distance from the root r. (*e.g.* The level of r is 0).
3: Compute the level of each candidate B_l in $\mathcal{B}_i^*$, where $level(B_l)$ is the smallest level of any item in B_l.
foreach *A node i in T_i^* in a decreasing order of level* **do**
  4: Let $\mathcal{C}_i$ be the set of candidates B_l in $\mathcal{B}_i$, each of which includes i and whose level is the same as the level of i.
  5: Let $opt(i)$ be the optimal solution for the problem considering only those candidates that contain items in the subtree below i.
  6: Compute $opt(i)$ recursively as follows

$$opt(i) = \max\left(\max_{B_l \in \mathcal{C}_i}\left(w_l + \sum_{j \in \mathcal{U}_B} opt(j)\right), \sum_{j \in ch(i)} opt(j)\right) \tag{1.9}$$

  where w_l is the value of candidate B_l and $ch(i)$ is the children nodes of i. $\mathcal{U}_B$ be the set of the roots of the forest of subtrees that are obtained by removing all nodes in B_l from T_i^*, while ignoring the subtree containing r.
7: The final solution is a set of candidates $\mathcal{B}_i^{opt}$ associated with $opt(r)$. Finally, $\mathcal{F}_i^K \leftarrow \forall B_l \in \mathcal{B}_i^{opt}$ where k_l of B_l is k.

computes the most likely tree $\mathcal{T}_i^*$ given $\mathcal{B}_i$. Once we obtain $\mathcal{T}_i^*$, we retain only the candidates $\mathcal{B}_i^*$ ($\subset \mathcal{B}_i$) that are subgraphs of $\mathcal{T}_i^*$. It is easy to see that Algorithm 1 runs in $O(|\mathcal{B}_i||\mathcal{S}|^2)$ time. Algorithm 1 first generates a complete undirected graph over $\mathcal{S}_i$ in which each edge (u, v) has the sum of values of $B_l \in \mathcal{B}_i$ such that $(u, v) \in B_l$. Its running time is $O(|\mathcal{B}_i||\mathcal{S}|^2)$. Then, the maximum spanning tree is obtained in $O(|\mathcal{S}|^2)$, and the final pruning step is performed in $O(|\mathcal{B}_i||\mathcal{S}|)$ at worst.

1.2. The DP based Search Algorithm

According to Theorem 1 in the main draft, given the $\mathcal{B}_i^*$ that are organized in the tree $\mathcal{T}_i^*$, the optimal solution to Eq.(1) can be achieved in $O(|\mathcal{B}_i^*||\mathcal{S}_i|)$. It can be solved by Algorithm 2, which is a dynamic programming (DP) based search algorithm by modifying the CABOB [3].

2. The FlickrMFC Dataset

We introduce a new dataset called FlickrMFC for the benchmark purpose of multiple foreground cosegmentation. They are sampled images from Flickr photostreams, each of which is taken by a single user for a specific event in a single day. Therefore, a fixed number of subjects frequently occur across the photostream, but an unknown subset of them appears in every single image. We also provide hand-labeled pixel-level ground truths.

It consists of 14 groups, each of which contains 12∼20 images. Fig.1 shows all images of four selected groups. The collected groups are as follows. (The group names are identical to the search keywords): {*dolphin+aquarium* (18), *apple+picking+fall* (20), *cow+pasture* (20), *liberty+statue+cruise* (18), *fishing+salmon+alaska* (12), *cheetah+safari* (20), *butterfly+blossom* (18), *stonehenge+england* (20), *baseball+kids* (18), *parrot+zoo+bird* (18), *dog+family+park* (20), *swan+zoo* (20), *zoo+gorilla+birds* (14), *thinker+Rodin* (16)}.

3. Results for ImageNet Dataset

In this section, we present some cosegmentation examples for the ImageNet dataset [2].

Fig.2 illustrates six segmented images for following synsets: *orangutan*, *snail*, *lion*, *garden+spider*, *otter*, *kit+fox*, *Australian terrier*, and *Nymphalid butterfly*. Most ImageNet images contain only a single object class of a significant size, and thus we sample some challenging images that contain a relatively small foreground in cluttered background. The results show that the proposed approach has been also successful to the single foreground cosegmentation, which is a simper task than the multiple foreground cosegmentation proposed in our paper.

References

[1] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE trans. Information Theory*, 14:462–467, 1968. 1

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009. 2

[3] T. Sandholm and S. Suri. BOB: Improved Winner Determination in Combinatorial Auctions and Generalizations. *Artificial Intelligence*, 145:33–58, 2003. 2

Figure 1. The FlickrMFC dataset. We show all images of four groups (*apple+picking+fall*, *cow+pasture*, *stonehenge+england*, and *parrot+zoo+bird*) from top to bottom. In each group, the first row shows input images, and the second row illustrates hand-labeled pixel-level ground truths.

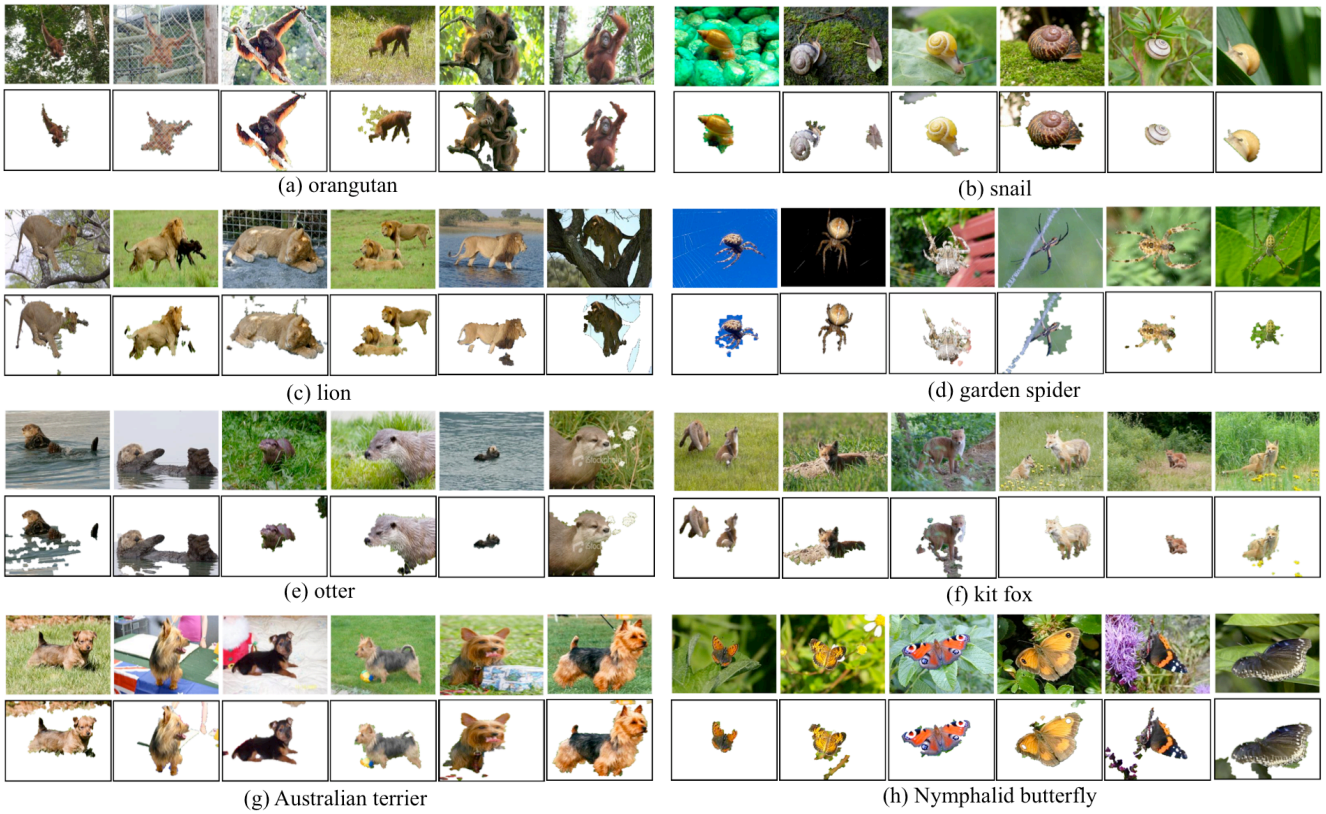

Figure 2. Examples of scalable cosegmentation on the ImageNet dataset. We sample six images from each synset. In each set, the first row shows input images, and the second row illustrates segmented foregrounds.