Object Cosegmentation

Sara Vicente
University College London

Carsten Rother
Microsoft Research Cambridge

Vladimir Kolmogorov
University College London

Abstract

Cosegmentation is typically defined as the task of jointly segmenting "something similar" in a given set of images. Existing methods are too generic and so far have not demonstrated competitive results for any specific task. In this paper we overcome this limitation by adding two new aspects to cosegmentation: (1) the "something" has to be an object, and (2) the "similarity" measure is learned. In this way, we are able to achieve excellent results on the recently introduced iCoseg dataset, which contains small sets of images of either the same object instance or similar objects of the same class. The challenge of this dataset lies in the extreme changes in viewpoint, lighting, and object deformations within each set. We are able to considerably outperform several competitors. To achieve this performance, we borrow recent ideas from object recognition: the use of powerful features extracted from a pool of candidate object-like segmentations. We believe that our work will be beneficial to several application areas, such as image retrieval.

1. Introduction

This paper addresses the task of unsupervised pixel-accurate segmentation of "similarly looking objects" in a small number of images, where no *a priori* information about the images (such as object class) is available. Such a problem was considered in recent papers under the name **image cosegmentation** [4, 14, 15, 19, 21, 23]. We improve on previous cosegmentation approaches by making the following contributions:

1) We observe that in most applications of cosegmentation the regions of interest are **objects**, i.e. "things" (such as a bird or a car) as opposed to "stuff" (such as grass or sky). Although this assumption was implicit in [4, 14, 15, 19, 21, 23], it was not directly incorporated in the models. We propose to add a measure of "objectness" explicitly, by exploiting object-like segmentation proposals from [11], and using features that were found to be important for scoring hypotheses in the case of single image segmentation.
We show that this improves results considerably compared to previous cosegmentation approaches. In fact, using just a single image as input already outperforms methods in [15, 23].
We use the term **object cosegmentation** to emphasize the fact that we are interested in segmenting "objects" rather than "stuff".

2) In general, the "objectness" assumption alone is not sufficient if, for example, an image contains multiple objects. We show that the performance can often be boosted further by learning a "similarity" measure between the segmentations of two images. More precisely, we train a Random Forest classifier for scoring a pair of segmentation proposals using both single image features and "pairwise" features such as differences of histograms.

Cosegmentation scenarios To the best of our knowledge, the task of cosegmentation has not previously been clearly defined. We argue that there is no "generic" cosegmentation problem. Indeed, "similarly looking object" can refer to different scenarios and each will have different degrees of variability of object appearances; cosegmentation algorithms should ideally take this into account.

In some previous work, "similarly looking object" referred to objects of the same class and the task was called *unsupervised class segmentation* (we discuss this in more detail in the next section).

Cosegmentation can also refer to the case where the images depict the same physical object. This scenario can be very challenging if, for example, the images capture different physical parts of the object (viewpoint or zoom change) or the object is deformable. Examples of such variations are the Stonehenge, Statue and Alaskan bear classes in Fig. 5. In fact, we believe that this may be even more challenging than segmenting different object instances from the same class, e.g. different spoons.

Our approach can be adapted to these different scenarios by training the system on an adequate dataset, with the appropriate variability of appearance. If the variability is too high then the method would learn to rely on single image features only.

Concerning the image background, it is important for our approach (and all other cosegmentation methods) that at least in some images the background varies. This avoids the trivial solution that the whole image is segmented as the object.

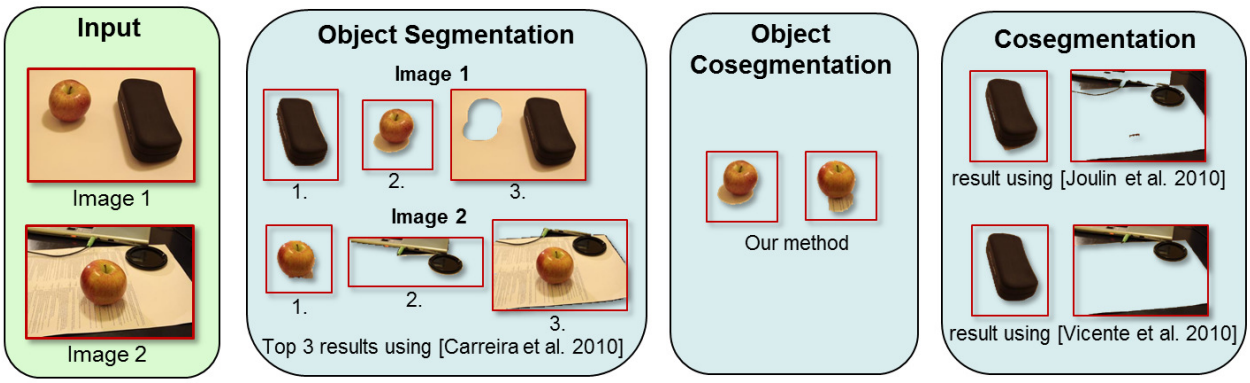

Figure 1. **Motivation for object cosegmentation.** The task is to extract the common object (apple) from the two input images. One possibility would be to use single image segmentation techniques that generate multiple object-like proposals. Taking the 3 top-ranked proposals for each image is enough to get an accurate segmentation of the apple. However, it is less clear how to choose the correct segmentation from that pool of proposals. Typical cosegmentation methods fail due to the similarities in terms of color between the black object in the first image and the background of the second image, preferring an arbitrary shaped segmentation for the second image. This failure in imposing object-like segmentations and the success of object segmentation methods in generating good proposals motivates our approach to cosegmentation: a system that chooses jointly for both images the best available proposal.

2. Related work

Single image segmentation The task of segmenting the foreground object in a single image without side information is ill-posed, since multiple objects might be present. Therefore, in the last two decades, a lot of research has focused on the problem of partitioning the image into multiple areas belonging to different objects and background.

Recently, a slightly different task has been suggested: instead of segmenting the image into multiple regions, the goal is to extract a set of binary segmentations ranked according with their object likeness. An example is [11], which is the work that we build on. Multiple segmentations are computed using parametric maxflow. Then, a scoring function is learned and used to select the best proposals. This scoring function encodes properties expected in all objects independent of the class, e.g. convexity, curvilinear continuity, contrast with the background, alignment of boundaries with image edges and the location in the image. The work in [13] follows a closely related approach with similar results.

Although these approaches have been successfully used as a building block for object segmentation and recognition systems (in particular [11] was part of the system that won the segmentation competition of the VOC Pascal Challenge 2009), it is not clear how to use them as a standalone method for image segmentation. Indeed, their accuracy relies on the use of multiple binary segmentations for each image, which leaves the question of choosing the optimal one.

Another related approach is the measure of "objectness" proposed in [2] and applied to bounding boxes. The motivation of this work is to reduce the search space of sliding window methods for object detection, by pre-computing an "objectness" measure for each bounding box, i.e. a probability of the box containing an object. Then, the object detectors specific to each class are only applied to the best scoring boxes in terms of "objectness".

Cosegmentation As mentioned in the introduction, there are several recent articles which address the generic task of cosegmenting multiple images [4, 14, 15, 19, 21, 23]. The methods in [15, 23] can be considered as state-of-the-art. The models used in these approaches do not explicitly encode the "objectness" assumption, which can lead to failure cases shown in Fig. 1: arbitrarily shaped regions (with similar appearance in both images) are segmented. Numerical results in our experimental section confirm that object-aware approaches can indeed outperform methods in [15, 23]. Another important difference is that previous systems have an objective function where each pixel is a random variable. In contrast, our objective works with a pool of proposal segmentations, hence we cannot represent all possible segmentations. We will confirm experimentally that in rare cases this limits our performance. Finally, another difference in our work is that previous cosegmentation systems did not train the weighting of features.

Unsupervised class segmentation Different generative models have been proposed for the task of Unsupervised class segmentation. Examples are the LOCUS model [24], which learns a shape model for the class, and the use of a topic model over image segments [10], assigning segments to topics depending on their visual words.

Both [10, 24] model separately what is common in all images (the shape of the object, sift features) and what is image specific (the appearance of the object).

Although [24] reports significantly better error rates than [10], it uses shape features, assuming a rough alignment of all the objects, given a reference frame. For example, [24]

reports results for a subset of the Weizmann horse database [6]. The horses in this dataset are all facing left. In [10] there is no modeling of shape information which makes it a more generic model, applicable to other scenarios.

Recently, [1] proposes a method inspired by interactive image segmentation. Similarly to LOCUS, the method alternates between learning a class model and updating the image segmentations, thus suffering from the same drawback of requiring rough alignment and similar pose.

We note that building a model for an object class typically requires a large number of training images, and therefore the methods above are not suitable for our scenario of only 2-10 input images.

3D reconstruction approaches In recent decades there has been significant progress on building 3D models from multiple images. While such approaches could be feasible for some of our examples, we believe that most of our data is too challenging for an automatic 3D reconstruction. In particular, it may be difficult to establish reliable point-to-point matches due to large changes of viewpoint and/or zoom. Some techniques, such as [9], work with silhouettes rather than point-to-point correspondences. However, correspondences are still needed for camera calibration. Note that, silhouettes correspond to object segmentations, and thus our approach can be seen as complementary to silhouette-based 3D reconstruction methods.

2.1. Applications of cosegmentation

In this section we discuss potential application areas of our system. Although we do not address these specific applications in this paper, they are worthy of mention.

One interesting scenario is the use of our system to rerank images in an image retrieval system. For example, if the input to the retrieval system is one image, our system may provide a ranking which focuses on the similarity of the common object as opposed to the similarity of the full image. If the input is a text query, e.g. "animals", our system can be used to provide object sensitive image pair-distances within the retrieved set. Clustering the images based on these distances can help visualize the variety of images within the results retrieved (see Fig. 2).

A possible related application is motivated by the recent trend of solving recognition problems by associating an image with a large database of images, e.g. [17], where the images in the database have meta-data attached (e.g. class label, motion direction). Our approach will help to find an object sensitive association. Another scenario is motivated by [4], where a user provides the system with a set of photographs from a photo collection which contain the same object, for example the iCoseg dataset in Fig. 5. Our system automatically provides a solution before any user interaction is done.

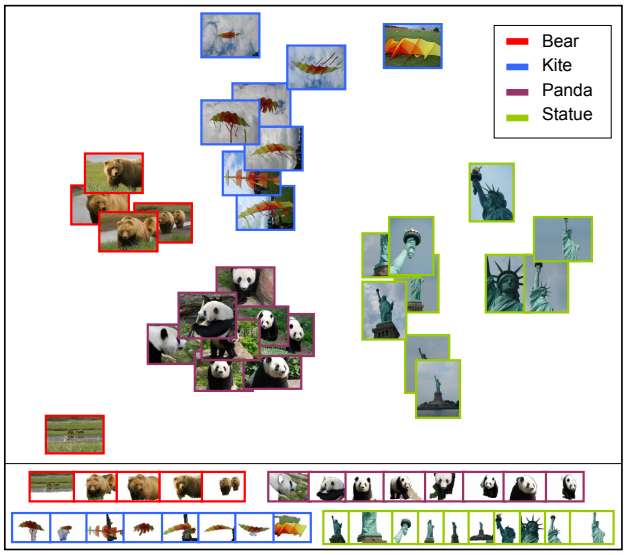

Figure 2. **Illustrating the idea of an object-sensitive clustering system**. We selected 31 images from 4 classes of the iCoseg dataset. All possible pairwise distances between the images were computed with our method, and used to map the images onto a 2D map (using multi-dimensional scaling) - the corresponding segmentations are in the bottom part. We see that the 4 classes are nicely separated and images with very similar foreground objects are close (e.g. top 2 bears). A retrieval system could visualize only the cluster means to illustrate the variability in the dataset. For each image we choose the closest image using our pairwise similarity measure and also show the segmentation corresponding to segmenting that pair.

3. Our approach

We are given L images containing the same object and the goal is to segment the common object. For each image $I_l, l = 1, ..., L$, we retrieve 200 proposal segmentations using the implementation of [11] and retain the 50 highest scoring ones. We denote by $\mathcal{S}_l = \{S_l^1, ..., S_l^{50}\}$ the set of proposal segmentations for image I_l. A proposal segmentation S_l^k is a binary labeling assigning to each pixel in the image I_l a label 0 for background, and label 1 for foreground. For all the objects in the image, we expect that one of these proposals contains only the full object. Fig. 3 shows examples of proposals obtained with [11].

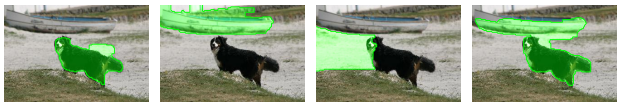

Figure 3. **Top scoring proposals obtained with [11].** Each proposal corresponds to a binary segmentation of the image and they are obtained by running parametric maxflow with different seeds. The score measures their object-likeness and it is computed based on several features: graph partition (e.g. value of the cut), region (e.g. area and perimeter) and gestalt (e.g. convexity). The result of our method for this image is shown in Fig. 8.

We formulate the task of cosegmentation as a labeling problem in a complete graph. Each image corresponds to a

node in the graph and each proposal segmentation to a label. The goal is to find a labeling $\boldsymbol{x} = (x_l | l = 1, ..., L; x_l \in \{1, ..., 50\})$ that maximizes the scoring function:

$$E(\boldsymbol{x}) = \sum_{l,p \leq L, l \neq p} P(x_l, x_p). \quad (1)$$

Assigning label $x_l = k$ corresponds to selecting the proposal S_l^k as the segmentation for image I_l.

The pairwise term $P(x_l, x_p)$ is learned using a Random Forest regressor [7]. This term encodes both how similar and how close to the ground truth the two proposals are, and is described in detail in section 3.1. Since the problem is defined in a complete graph, we compute the pairwise term for all pairs of proposals for all pairs of images.

Inference We use for inference an exact *A**-search algorithm introduced in [3, 5]. The use of an exact inference algorithm limits the number of images that can be jointly segmented. Alternatively we could use an approximate inference method, like loopy belief propagation. For the simplest case, where L=2, the inference reduces to choosing the pair of proposals with the highest score.

3.1. Learning the pairwise term between proposals

At training time, our method requires ground truth segmentations of pairs of images depicting similar objects. The test images belong to different classes than the ones used to train the system.

For each training image, we start by extracting proposal segmentations using [11]. Then, for each pair of proposals, we extract features depending both on the proposals and on the corresponding images. We extract a total of 33 features. There are two different types of features. The first type (section 3.1.1) takes into account the two proposals and images simultaneously. The second type (section 3.1.2) only considers one of the images.

We train a Random Forest regressor based on these features. For the two proposals being considered, we compute the overlap of each proposal with ground truth and regress on the sum of the overlaps, where the overlap is given by: $Overlap(S_l^k, GT_l) = (S_l^k \cap GT_l)/(S_l^k \cup GT_l)$. At test time, the score of the Random Forest regressor is used as a pairwise term between proposals.

3.1.1 Features including both images

Most of the features discussed in this section are based on histogram similarity.

Given two normalized histograms h_1 and h_2 with b bins, we use as histogram similarity the χ^2-distance measure: $\chi^2(h_1, h_2) = \sum_b (h_1^b - h_2^b)^2/(h_1^b + h_2^b)$.

We consider a total of seven features that take into account both images. The first three features depend on the foreground segment of both images and the last features only depend on the proposal segmentations.

Similarity between the foreground color histograms of both proposals. The color histograms are computed by fitting a Gaussian Mixture Model (GMM) to the RGB color of both images simultaneously, where each Gaussian in the mixture model corresponds to a bin.

Similarity between the foreground histograms of textons. We use the implementation of [12] to compute a patch codebook with 100 clusters for each pair of images. The foreground histogram of each proposal is obtained from this codebook.

Similarity between the foreground histograms of SIFT descriptors. For each pair of images, we compute SIFT descriptors [18] over a regular grid and cluster them in 100 clusters. We use the code from [16].

Similarity between the curvature histograms of the segmentation (2 features). To compute a histogram over curvature, we use an integral representation of the curvature [8]. For each point in the boundary we compute the number of foreground pixels inside a circle centered at that point. We use two circles of different radius, obtaining two different histograms.

Similarity between the histograms of the boundary orientation. For each point in the boundary we compute the orientation at that point and cluster them into eight bins.

Segmentation overlap. Overlap of both segmentations when constrained to the tightest possible bounding box and reshaped to have 64×64 pixels.

3.1.2 Features independent for each image

Following [11], we also consider features that are computed individually for each image.

Foreground and background similarity (3 features). Distance between the foreground and background histograms of color, textons and SIFT described in the previous section.

Alignment with image edges. Average edge strength on the segmentation boundary.

Centroid (2 features). Coordinates of the center of mass of the foreground region, normalized by each dimension.

Major and minor axis length (2 features). Lengths of the major and the minor axes of the ellipse that has the same normalized second central moments as the segmentation.

Convexity and area (2 features). Ratio of the number of foreground pixels over the area of the convex hull and over the total area of the image.

Bounding box (2 features). Size of the bounding box (2 dimensions), normalized by the size of the image.

Boundary pixels. Percentage of boundary pixels that belong to the segmentation.

3.2. Learning a single image classifier

For completeness, in the experimental section (section 4), we report the results of training a regression Random Forest for single images, using the features discussed in section 3.1.2.

This method is similar to [11], differing only in some of the features and in the dataset used for training. The results of a single image classifier, trained with the same features used for the pairwise classifier, provide information of how much gain in performance, comparing with existing cosegmentation methods, comes from single image measures alone.

3.3. Possible extensions

The score of the single image classifier can be directly included in the scoring function as a unary term. However, this requires an extra parameter that weights the unary and the pairwise parts of the model.

Another possible extension is to address the case when the variability between pairs of objects in the set is high. Consider the following example scenario, where the set has three images A, B and C. The objects in images A and B are very similar, but the object in C is quite different to both A and B. We expect that $\mathcal{P}(A, B)$ is large, and both $\mathcal{P}(A, C)$ and $\mathcal{P}(B, C)$ are small, where $\mathcal{P}(A, B) = P(x_A^*, x_B^*)$, i.e. the similarity between the two selected proposals in each image. Currently, the segmentation of the object in image B is influenced *equally* by the pairwise term $\mathcal{P}(A, B)$ and $\mathcal{P}(B, C)$. The idea is to down-weight the importance of the term $\mathcal{P}(B, C)$. To achieve this we plan to replace the sum over P in (1) by a sum over $f(P)$, where f is some robust function, e.g. truncated linear. Learning f is left as future work.

4. Experiments

We report results on three different datasets: a dataset with 20 pairs of images of the same object, the iCoseg dataset [4] and the MSRC dataset [22].

We show quantitative and qualitative results. The measure used for the quantitative results is the accuracy, i.e. the percentage of pixels in the image (both foreground and background) correctly classified. Since the performance of our method varies substantially for different classes, we separate the results per class. Note, our algorithm does not use any information about the class of the object.

4.1. Images with the same object

The *cosegmentation dataset* contains 20 image pairs with the exact same object in similar poses and typically with very different backgrounds. i.e. the ideal setting for cosegmentation. Most of these images have been used before in cosegmentation [14, 21]. For this dataset, the accuracy using leave-one-out cross validation ,i.e. using 19 pairs of images for training and testing in the remaining pair, is 91.9% for our single image implementation and 91.8% for our joint method applied to pairs of images.

The results shown in Fig. 4 are visually comparable to the ones presented in previous work on cosegmentation [14, 15, 19, 21]. The accuracy for our method is slightly lower compared with the best accuracy previously reported

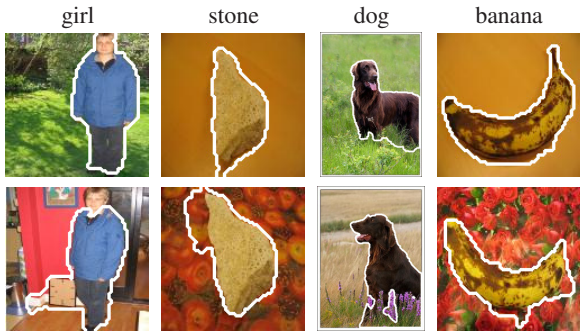

Figure 4. **Results for images with the same object.** For each pair of images we report the accuracy for our method and, in brackets, the upper bound.

for each of the images: "girl" 98% [14], "stone" 99% [15], "dog" 96% [14] and "banana" 97% [19]. However, the performance of our method is upper bounded by the accuracy of the best segmentation in the pool of proposals (this accuracy is reported in brackets in Fig. 4). A post-processing step, using e.g. [20], could further improve our results, by recomputing a pixel-accurate segmentation. A similar post-processing step was used in [15]. Note that, in [14, 19] the authors used prior information about the object's color, incorporated in the model as unary terms.

4.2. The iCoseg dataset

The iCoseg dataset introduced in [4] contains several groups of images. In [4] the authors used the iCoseg dataset in an interactive cosegmentation framework and, to the best of our knowledge, this dataset was never used in a fully automatic setting.

Each group contains images of the same object instance or similar objects from the same class. iCoseg is a challenging dataset because the objects are deformable, change considerably in terms of viewpoint and illumination, and in some cases, only a part of the object is visible. This contrasts significantly with the images typically used to test cosegmentation systems, like the ones in Fig. 4. The diversity of the dataset can be seen in the Fig. 5, 6 and 7.

We resized the images to half the size and excluded images in some of the groups to make it feasible to use A^*-search for maximizing function (1).

We use the *cosegmentation dataset* (discussed in section 4.1) for training. This shows that training our model on a totally distinct dataset gives good performance.

Table 1 shows the segmentation accuracy of different methods for the iCoseg dataset. We compare our results with three previously proposed methods. The method of [11] was designed for single image segmentation and we select the highest scoring segmentation as the result.

The second method [23] extends GrabCut [20] to pairs of images by jointly modeling the foreground color of both images. Since this method was formulated for pairs of images,

	Our method		Competitors			Baselines	
	1 image	All images	1 image [11]	Pairs [23]	All images [15]	Upper bound	Uniform
Alaskan bear (9/19 images)	79.0	**90.0**	60.4	58.2	74.8	96.4	79.0
Balloon (8/24 images)	79.5	90.1	**97.5**	89.3	85.2	99.3	86.8
Baseball (8/25 images)	84.5	**90.9**	74.6	69.9	73.0	96.5	88.8
Bear (5/5 images)	78.2	**95.3**	83.5	87.3	74.0	97.5	68.4
Elephant (7/15 images)	**75.4**	43.1	74.3	62.3	70.1	96.5	82.9
Ferrari (11/11 images)	84.8	**89.9**	71.8	77.7	85.0	97.1	73.9
Gymnastics (6/6 images)	82.1	**91.7**	72.2	83.4	90.9	96.4	83.4
Kite (8/18 images)	89.3	**90.3**	81.5	87.0	87.0	96.7	83.5
Kite panda (7/7 images)	80.2	**90.2**	87.7	70.7	73.2	97.8	68.7
Liverpool (9/33 images)	87.4	**87.5**	83.2	70.6	76.4	92.7	76.0
Panda (8/25 images)	87.8	**92.7**	79.5	80.0	84.0	96.3	62.0
Skating (7/11 images)	78.4	77.5	73.4	69.9	**82.1**	85.8	62.7
Statue (10/41 images)	92.9	**93.8**	91.5	89.3	90.6	97.8	73.7
Stonehenge (5/5 images)	**84.2**	63.3	83.3	61.1	56.6	96.1	78.2
Stonehenge 2 (9/18 images)	**88.9**	88.8	79.7	66.9	86.0	93.8	64.4
Taj Mahal (5/5 images)	80.7	**91.1**	82.2	79.6	73.7	96.5	82.2

Table 1. **Segmentation accuracy for the iCoseg dataset**

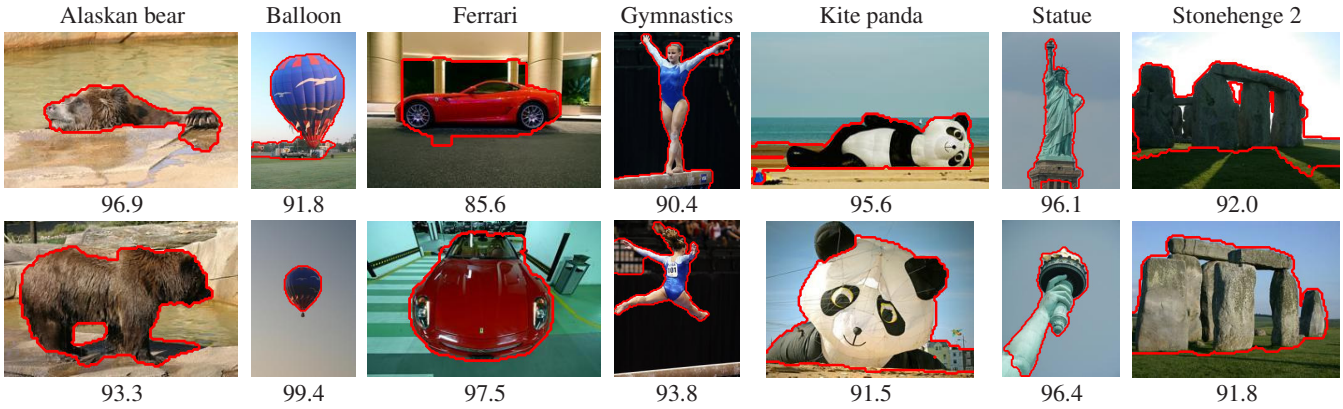

Figure 5. **Results for the iCoseg dataset.** Our method is robust to changes in object size (Balloon), viewpoint (Ferrari and Gymnastics) and partial occlusions of the object (Alaskan bear, Kite panda and Statue). Below each image we report the accuracy of the segmentation.

we apply it to all possible pairs in each class. We chose two different ways of initializing the algorithm (using the histogram intersection proposed in [23] and using the best scoring segmentations from [11]), and then selected the result with lower energy. The accuracy reported is the average of accuracies for all pairs.

We also compare with [15]. We use the reference implementation of the method and set the only free parameter to 0.001. Since the superpixel code used in [15] is not freely available, we use mean shift to compute the superpixels. For each class, we tested this method using SIFT and color features, with and without graph cut post-processing and report results for the best of the 4 settings.

The last two columns show two different baselines. First, we report the accuracy upper bound for our method. This is given by choosing the best segmentation according to ground truth from the 50 used proposals. Finally, we report results considering a uniform segmentation, i.e. for each image, we take the full and the empty segmentations and choose the one with the best error rate.

Our method using all images is best for 11 out of 16 classes. For 4 classes (Balloon, Elephant, Skating and Stonehenge) other methods are clearly better, which we discuss below.

Fig. 5 shows results of jointly segmenting all the images in a class. The dataset contains considerable variation within each class and our method is robust to that variation. For example, the Alaskan bear and Statue classes have images with significant object occlusion while the Ferrari images have great variations in terms of view point.

Fig. 6 compares the result of segmenting the images individually and jointly. For both classes, segmenting jointly

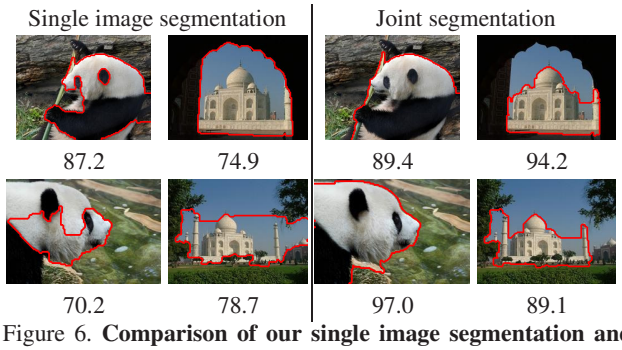

Figure 6. **Comparison of our single image segmentation and joint segmentation.** Single image segmentation fails to correctly segment the object due to strong internal edges (Panda) and strong edges in the background (Taj Mahal).

significantly increases the accuracy of the segmentations. For example, for the Panda images, the single image method aligns with the strong boundaries inside the object, while the joint segmentation, correctly retrieves the full panda.

Table 1 shows that for some classes the joint method is outperformed by single image segmentation, e.g. Elephant and Skating. Fig. 7 shows segmentations for some images in those classes. In both the Elephant and the Stonehenge classes the object is depicted in very similar backgrounds, which increases the ambiguity of the cosegmentation task.

Note that, for the other group with Stonehenge images (Stonehenge 2), some of the images have very different lighting conditions which helps disambiguate the object. This can be seen in the last column of Fig. 5.

For the Skating class, the object is quite complex since all the skaters are considered foreground. Since the proposal segmentations considered by our algorithm are connected, there is a considerable amount of background incorrectly labeled foreground.

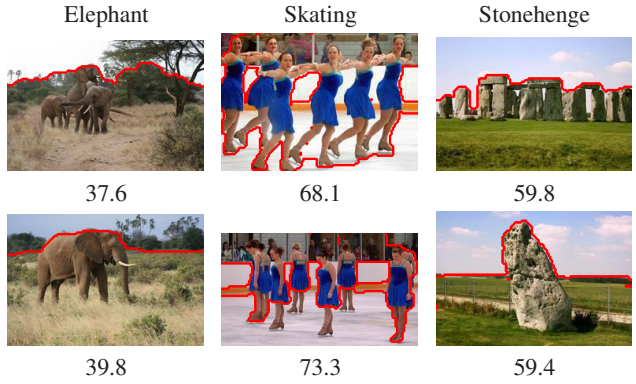

Figure 7. **Failure cases for the iCoseg dataset.** Joint segmentation fails for these classes due to the high similarity of the background in all the images (Elephant and Stonehenge) and the complexity of the object (Skating).

4.3. Images with objects of the same class

The MSRC dataset was first introduced in the context of supervised class segmentation [22]. We select a subset of its classes and 10 images for each of these classes such that there is a single object in each image.

For each class we train a regression Random Forest using the images of the remaining classes. We take all possible pairs of images inside a class as training examples. We show results for this experiment in Fig. 8.

Since, as an intermediate step for our model we need to compute the pairwise scores for all pairs in the class, we also report the average accuracy of the top scoring pair of proposals, i.e. for each pair of images, we take as the solution the segmentations with the highest pairwise term.

For the MSRC dataset the use of joint segmentation does not improve scores considerably compared with using our single image classifier. We believe that this is due to the characteristics of the dataset, where objects tend to be centered in the image, have a good contrast with background, be homogeneous in terms of color. Another reason may be the larger intra-class variability, compared with iCoseg.

5. Discussion

From the results reported for the different datasets we conclude that our method outperforms existing methods for cosegmentation.

For unsupervised object class segmentation (section 4.3) using single image methods (e.g. [11]) already outperforms existing methods for cosegmentation. Intuitively, using multiple images should provide more information and make the problem easier to solve; however, that is not the case for this dataset due to the amount of intra-class variation and the fact that the objects are usually very distinct from the rest of the image. Although our method uses multiple images, it is capable of adapting to such situation by weighting the importance of single image features accordingly.

We also showed that for the MSRC dataset our single image version outperforms [11]. This is probably due to the fact that they use the Pascal VOC dataset for training, which contains high variability in terms of object properties, while we perform leave-one-out cross validation in the MSRC dataset, i.e. while testing in one of the classes we train in the remaining classes. This further supports our claim that adapting the training set to the task is important for achieving good performance.

In the experiments for the iCoseg dataset, we showed that our method considerably outperforms both state-of-the-art methods for cosegmentation and single image approaches. It has, however some limitations, especially when the background is very similar in all of the images.

In summary, our method presents several advantages compared to existing methods for the same or similar tasks: (1) it can be applicable to sets with a small number of images (in contrast to generative methods for unsupervised object segmentation); (2) it does not require images of the

	Our method			Competitors		
	1 image	Pairs	All	1 image [11]	Pairs [23]	All [15]
Bird	90.8	94.5	**95.3**	90.7	88.0	62.2
Car	80.2	**80.7**	79.6	72.3	64.9	78.6
Cat	91.9	92.0	**92.3**	87.8	77.5	80.8
Cow	93.9	93.5	**94.2**	92.9	91.9	80.8
Dog	92.9	**93.1**	93.0	88.7	86.7	75.6
Plane	82.7	82.8	**83.0**	78.2	65.7	80.3
Sheep	**94.6**	93.7	94.0	94.3	89.8	92.5

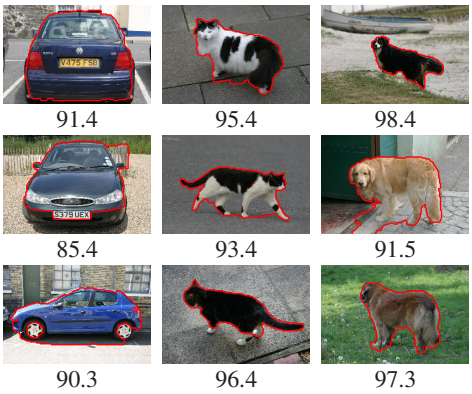

Figure 8. **Results for the MSRC dataset.** For these dataset, the results of our joint method are comparable with single image segmentation.

same class for training (as opposed to supervised object segmentation methods) and (3) it can be adapted to different cosegmentation scenarios, by using a different training set (as opposed to cosegmentation methods that have a "fixed" concept of distance between foreground segments).

6. Conclusions and Future Work

We presented a novel method for cosegmentation of images. The main contribution is to show that requiring the foreground segment to be an *object* significantly improves on existing methods. We include this constraint by building upon methods that generate a pool of object-like proposal segmentations. In practice, for many applications, imposing this constraint does not reduce the usefulness of the method.

We show state-of-the-art results in a recently introduced challenging dataset. In this dataset, the objects present large variations of viewpoint, scale and illumination.

As future work, we believe that the ideas listed in section 3.3 may be fruitful. We are also excited about applying our system in the future to application scenarios mentioned in section 2.1.

Acknowledgements We thank João Carreira for providing the implementation of [11] and for helpful discussions.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. Classcut for unsupervised class segmentation. In *ECCV*, 2010.

[2] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *CVPR*, 2010.

[3] B. Andres, J. H. Kappes, U. Koethe, C. Schnörr, and F. A. Hamprecht. An empirical comparison of inference algorithms for graphical models with higher order factors using OpenGM. In *DAGM 2010*, 2010.

[4] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen. iCoseg: Interactive co-segmentation with intelligent scribble guidance. In *CVPR*, 2010.

[5] M. Bergtholdt, J. Kappes, S. Schmidt, and C. Schnörr. A study of parts-based object class detection using complete graphs. *IJCV*, 87(1-2):93–117, 2010.

[6] E. Borenstein and S. Ullman. Class-specific, top-down segmentation. In *ECCV*, 2002.

[7] L. Breiman. Random forests. In *Machine Learning*, 2001.

[8] J. W. Bullard, E. J. Garboczi, W. C. Carter, and E. R. F. Jr. Numerical methods for computing interfacial mean curvature. *Computational Materials Science*, 4:103–116, 1995.

[9] N. D. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. Automatic 3D object segmentation in multiple views using volumetric graph-cuts. *Image and Vision Computing*, 28.

[10] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent object segmentation and classification. In *ICCV*, 2007.

[11] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *CVPR*, 2010.

[12] T. Deselaers and V. Ferrari. Global and efficient self-similarity for object classification and detection. In *CVPR*, 2010.

[13] I. Endres and D. Hoiem. Category independent object proposals. In *ECCV*, 2010.

[14] D. S. Hochbaum and V. Singh. An efficient algorithm for co-segmentation. In *ICCV*, 2009.

[15] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *CVPR*, 2010.

[16] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[17] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. Sift flow: Dense correspondence across different scenes. In *ECCV*, 2008.

[18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91, 2004.

[19] L. Mukherjee, V. Singh, and C. R. Dyer. Half-integrality based algorithms for cosegmentation of images. In *CVPR*, 2009.

[20] C. Rother, V. Kolmogorov, and A. Blake. Grabcut - interactive foreground extraction using iterated graph cuts. *SIGGRAPH*, August 2004.

[21] C. Rother, V. Kolmogorov, T. Minka, and A. Blake. Cosegmentation of image pairs by histogram matching - incorporating a global constraint into MRFs. In *CVPR*, 2006.

[22] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

[23] S. Vicente, V. Kolmogorov, and C. Rother. Cosegmentation revisited: models and optimization. In *ECCV*, 2010.

[24] J. Winn and N. Jojic. Locus: learning object classes with unsupervised segmentation. In *ICCV*, 2005.