Segmentation Propagation in ImageNet

Daniel Kuettel[1,2⋆] Matthieu Guillaumin[2⋆] Vittorio Ferrari[1]

[1] University of Edinburgh, UK [2] ETH Zürich, Switzerland

Abstract. ImageNet is a large-scale hierarchical database of object classes. We propose to automatically populate it with pixelwise segmentations, by leveraging existing manual annotations in the form of class labels and bounding-boxes. The key idea is to recursively exploit images segmented so far to guide the segmentation of new images. At each stage this propagation process expands into the images which are easiest to segment at that point in time, e.g. by moving to the semantically most related classes to those segmented so far. The propagation of segmentation occurs both (a) at the image level, by transferring existing segmentations to estimate the probability of a pixel to be foreground, and (b) at the class level, by jointly segmenting images of the same class and by importing the appearance models of classes that are already segmented. Through an experiment on 577 classes and 500k images we show that our technique (i) annotates a wide range of classes with accurate segmentations; (ii) effectively exploits the hierarchical structure of ImageNet; (iii) scales efficiently; (iv) outperforms a baseline GrabCut [1] initialized on the image center, as well as our recent segmentation transfer technique [2] on which this paper is based. Moreover, our method also delivers state-of-the-art results on the recent iCoseg dataset for co-segmentation.

1 Introduction

Foreground-background segmentation is the fundamental task of producing a binary segmentation of an image, separating the foreground object from the background [1, 3, 4]. Segmentation is useful in many higher-level applications such as object recognition, as it provides the spatial support for extracting texture and shape descriptors on objects [5, 6]. It is also valuable for human pose estimation, where silhouettes have been shown to reliably convey pose [7], and for 3D reconstruction from silhouettes. However, manually annotating images with segmentations is tedious and very time consuming. This prevents the above applications to scale both in the number of training images and the number of classes. On the other hand, we have witnessed the advent of very large scale datasets for other computer vision applications, including image search [8] and object classification [9].

In this paper, we want to bridge the gap between these domains by automatically populating the large-scale ImageNet [10] database with foreground segmentations (fig. 1). ImageNet[1] contains millions of images annotated by the class label of the main object.

⋆ These authors contributed equally to this work.

[1] http://www.image-net.org/

Fig. 1. *Example segmentations produced by our method on ImageNet at stage 2 (left) and stage 3 (right). Each row of a stage shows 5 images of a class with the segmentations overlaid in red. The rightmost column of each class shows segmentation faults.*

However, only a small fraction of the images is annotated with bounding-boxes, and *none* with foreground segmentation. Our method leverages these existing annotations while exploiting the semantic hierarchy of ImageNet to populate its images with segmentations of their main objects, see fig. 1. Our work weaves together and extends several recent developments including Grabcut [1], segmentation transfer [11, 2], efficient binary codes [8], cosegmentation [3, 12] and structured output learning [13, 14] into a fully automatic, computationally efficient and reliable large scale segmentation framework. We jointly segment groups of semantically related images by sharing appearance models, and help the process by importing appearance models from related classes that were segmented in previous stages of our segmentation propagation process.

In an extensive experimental evaluation, we show that our process accurately segments 500k images over 577 classes. To our knowledge, this is the largest segmentation experiment to date. Moreover, we validate the components of our approach on the smaller iCoseg dataset, where we outperform the state-of-the-art [3, 12, 15, 16].

2 Related Work

Object segmentation. Fully supervised segmentation techniques aim at separating instances of an object class from their background (e.g. horses, faces, cars [17–19]). They are supervised in that the training set shows images of other instances of the class along with their binary segmentations. Two main directions have been explored to reduce the burden of annotating images with ground-truth segmentations. The first is to reduce the degree of supervision [20–22] by either annotating only a fraction of the pixels [22] or by providing only the names of the object class appearing the image [20,

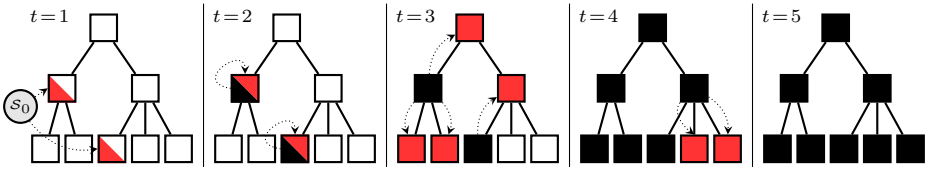

Fig. 2. *Illustration of segmentation progapation on ImageNet. The stage of propagation is marked by t. Nodes are classes and edges represent the class hierarchy. Node colors indicate the state of a class: white = "unsegmented", red = "currently being segmented" ($\mathcal{T}_t$), and black = "already segmented" ($\mathcal{S}_{t-1}$). Diagonally split nodes are classes partially annotated with bounding-boxes (bottom-left corner). Segmentation transfer is shown by arrows.*

21]. Our work is related to this, as most of the images in ImageNet are only labeled by class names. A second, recent trend is to guide the segmentation process using generic object localization tools [23, 24], as in [2, 15]. We build on the segmentation transfer scheme of [2], but make it computationally much more efficient to scale up to ImageNet.

Interactive segmentation [1, 25, 26] has been thoroughly researched since the very popular GrabCut [1]. Most of these approaches minimize a binary pairwise energy function whose unary potentials are determined by appearance models estimated based on user input on the test image. Our approach builds on their energy formulation, but is fully automatic.

Our work is also related to co-segmentation, where the task is to segment multiple images at the same time [3, 12, 15, 16, 27]. Similar to [3, 12], we share appearance models when segmenting many images of the same class. This sharing helps identifying which image regions belong to the foreground object.

Knowledge transfer. There is also a trend towards transferring knowledge to help learning a new class from a few training examples by leveraging examples from related classes [28–30]. Most of these works aim at object recognition or detection, not segmentation. Knowledge transfer is typically done by regularizing model parameters [28], through an intermediate attribute layer [29] or by sharing parts [30]. For segmentation, we propose to use appearance models of previously segmented classes to help segmenting a new class. Moreover, our segmentation propagation scheme automatically determines which classes to segment next.

3 Overview of our approach: Segmentation Propagation

Our goal is to derive a binary segmentation for each image in ImageNet, accurately delineating its main object. A key idea is to employ the images segmented so far to help segmenting new images. At any stage t, we employ a source pool $\mathcal{S}_{t-1}$ of segmented images to *transfer* segmentations to a target set $\mathcal{T}_t$ of new unsegmented images. The idea is to transfer segmentations masks from windows in a subset of $\mathcal{S}_{t-1}$ to visually similar windows in $\mathcal{T}_t$ and then use GrabCut to refine the segmentation (sec. 4). The subset of $\mathcal{S}_{t-1}$ is chosen based on semantic similarity between classes. The newly segmented images in $\mathcal{T}_t$ are then added to the source pool, forming the pool $\mathcal{S}_t$, which is used as source in the next stage. Since no segmented images are available in ImageNet, we start this recursive process from the PASCAL VOC 2010 segmentation challenge

images ($\mathcal{S}_0$). The process is like a wave spreading through ImageNet, gradually segmenting more and more images (fig. 2). In stage $t = 1$, the wave propagates from $\mathcal{S}_0$ to ImageNet images annotated with ground-truth bounding-boxes. We start from these images because here the segmentation task is the easiest as the bounding-boxes provide a reliable estimate of the object location. Moreover, we jointly segment images in the same class by sharing appearance models across them (sec. 5). This further improves segmentation accuracy. Because of all these factors, the output of stage $t = 1$ are excellent segmentations for tens of thousands of images, which can be used as surrogate ground-truth in the next stages (see sec. 6.2 for a quantitative evaluation).

After the images in $\mathcal{T}_1$ are segmented, they are added to the source pool $\mathcal{S}_1 = \mathcal{S}_0 \cup \mathcal{T}_1$ to support the segmentation of a larger set of images $\mathcal{T}_2$. A key issue is now: which images should be processed next? All remaining images are annotated only with a class label, no bounding-box is left. In general, a good choice for $\mathcal{T}_t$ would be unsegmented images most related to the images in the source pool $\mathcal{S}_{t-1}$, in terms of the kind of objects they contain. Importantly, all images in ImageNet are labeled by class labels and these are organized in a semantic hierarchy. Therefore, we exploit the semantic relation between the class labels to define $\mathcal{T}_t$. Our choice for $\mathcal{T}_2$ is the set of unsegmented images with the *same* class label as any image in $\mathcal{T}_1$ (i.e. 0 semantic distance). Analog to stage 1, we jointly segment images in a class C to improve accuracy, using as source the subset of $\mathcal{S}_1$ consisting of $\mathcal{S}_0$ and the images of C segmented at stage 1.

After stage $t = 2$, all remaining classes are completely unsegmented and contain no image with bounding-boxes. Therefore, we create $\mathcal{T}_t$ from batches containing to entire classes. A new class C is included in $\mathcal{T}_t$ if it is *directly related* to a class C' in $\mathcal{S}_{t-1}$. Two classes are directly related if they are connected by an edge in the ImageNet DAG (*i.e.* they are parent-child). In addition to jointly segmenting all images in a new class C, here we also import appearance models from its related classes C', which further helps accuracy (sec. 5.2). Over the subsequent stages, the wave progressively spreads to siblings, then to cousins, and continues until the whole ImageNet is segmented.

When transferring from $\mathcal{S}_{t-1}$ to a class C in $\mathcal{T}_t$, we restrict the source pool to classes directly related to C and all their respective sources. Hence, the source pool is tailored to a target class to be maximally related to it and always contains $\mathcal{S}_0$. When there is no possible confusion, we will simply denote the source pool as $\mathcal{S}$. Overall, our segmentation propagation scheme balances two opposing forces. On the one hand the source pool rapidly grows and contains exactly those images most highly related to the target ones, which helps segmentation transfer. On the other hand, errors in segmentations generated in a stage degrade the quality of the source pool and risk propagating to later stages. Our scheme balances these forces to make segmentation transfer work at every stage and ultimately produce high quality segmentations for a large subset of ImageNet.

We detail below the components of our approach. In sec. 4 we describe segmentation transfer. Then, sec. 5 details the energy minimization framework used to jointly segment all images of a class. This includes sharing appearance models within the class (sec. 5.1) and importing appearance models of related classes from the source pool (sec. 5.2). In sec. 6, we present experimental results on the iCoseg data set (sec. 6.1), then on 500k images from ImageNet (sec. 6.2) and draw conclusions (sec. 6.3).

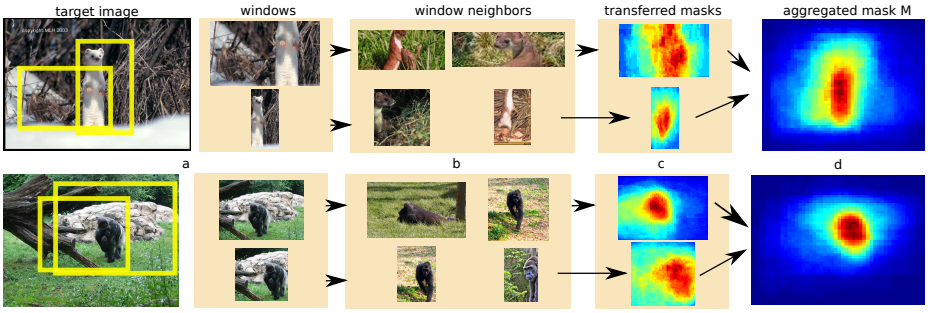

Fig. 3. *Two examples of window-level segmentation transfer at stage 3. (a) two out of 100 windows extracted in a target image; (b) the most similar windows from the source set $\mathcal{S}_2$ transfer their segmentation masks to the windows of the target image, giving (c); (d) the 100 individual window masks are aggregated into a single soft-segmentation mask M for the whole target image.*

4 Large-scale segmentation transfer

We present here the paradigm of *segmentation transfer* [11, 2], and explain how to make it computationally very efficient to scale up to ImageNet. We then describe how the parameters of this transfer mechanism are learnt.

To segment a new image i, the idea is to transfer segmentation masks from the images most similar to i in the source pool $\mathcal{S}$ of pre-segmented images. The transferred masks are then used to derive a unary potential of a pairwise energy function which is minimized to refine the segmentation (sec. 5).

4.1 Window-level segmentation transfer

The basic scheme [11] compares the image i to the source images $\mathcal{S}$ based on global descriptors capturing the image as a whole. The segmentation masks of the most similar source images are averaged into a mask M_i for i. Very recently, [2] improved on this basic scheme by transferring segmentation masks at the level of *windows* (fig. 3a). In each image, we first extract 100 candidate windows using the objectness sampling [23] and then transfer masks from windows in $\mathcal{S}$ (fig. 3b) to visually similar windows in i (fig. 3c). Because of the objectness sampling, some candidate windows are centered on objects (*e.g.* cow, motorbike, telephone) rather than background elements, making them a better spatial support for segmentation transfer. Using windows is preferable because they exhibit less variability than whole images, so they are easier to match. Moreover, they enable to compose novel scenes using local parts of different images from $\mathcal{S}$.

After transferring masks for each window independently (fig. 3c), they are aligned to their corresponding windows in i and aggregated into a single mask M_i (fig. 3d, see sec. 4.3). Hence, $M_{ip} \in [0, 1]$ estimates the probability that the pixel p is foreground in image i (fig. 3d). M_i is then used in two different ways in our energy minimization framework (see sec. 5). First, they automatically setup the unary potentials based on appearance models by estimating their parameters for the foreground and background classes. Second, they are used directly as a location prior unary potential that encourages the final segmentation to be close to M_i (see [2]).

4.2 Efficient segmentation transfer

The quality of the output segmentation depends on the source pool $\mathcal{S}$ containing windows with appearance as similar as possible to windows in i and with segmentation masks truly reflecting the underlying segmentation of i. In the spirit of recent work for recognition [9], we aim at collecting the largest possible pool of segmented windows. When applying this idea to millions of images that contain hundreds of windows, a key requirement is efficiency both in terms of computation and memory.

The first step to reduce computational cost is to describe windows very quickly. Instead of GIST as used in [11, 2], we prefer HOG which are much faster for virtually no loss in retrieval performance. The second step is to speed up the computation of distances between the descriptors of all windows in i to all windows in $\mathcal{S}$. This is in theory the most computationally expensive step in segmentation transfer. With 100 windows per image and $|\mathcal{S}|$ =100k images, 1 billion distance computations are needed to segment a single image! Moreover, storing the HOG descriptors for 10M windows in $\mathcal{S}$ requires 310 GB of memory. This is both too large to fit in the memory of a desktop machine and too slow to read from disk for each new image to segment.

In this paper we employ the efficient binary coding scheme called "Iterative Quantization" (ITQ) [8] to circumvent this issue. The key idea of ITQ is to encode high-dimensional descriptors as short binary vectors so that points close in L2 distance in the original descriptor space are close for the Hamming distance in the binary space. Using 512 bits (*i.e.* 64 bytes) to encode each HOG, 10M windows now account for a mere 640Mb. Moreover, hamming distances are particularly fast to compute on modern CPUs, which can perform a 64-bit XOR in a single operation. Our standard desktop computer achieved a rate of about 40 million distances computations per second (on a single core of an Intel Core i7 CPU 923 2.67GHz). While this is already fast enough for the large-scale experiments in this paper, it could be accelerated even further with fast nearest neighbour techniques dedicated to hamming codes [31].

4.3 Aggregating neighbor masks

As explained above, the key operation in our scheme is to transfer segmentations from the K appearance nearest neighbors $\{s_1, s_2, \ldots, s_K\}$ in $\mathcal{S}$ to the target window w. We model the mask m_w for w as a weighted sum of the masks m_{s_k} of its neighbors: $m_w = \sum_{k=1}^{K} \lambda_k m_{s_k}$, where $\lambda_k \geq 0$, $\sum_k \lambda_k = 1$ and all the masks are normalized to the same size. Using uniform weights λ_k would make the transfer very dependent on K. An excessively large K would simply average the segmentations in the source pool, ignoring image appearance, making K an important parameter to be set. Instead, we propose here to *learn* λ_k using training images from PASCAL VOC10 along with their ground-truth segmentations, which we use to derive ground-truth masks m_w. We train the weights λ_k by minimizing the Frobenius norm $||\cdot||_F$ of the residuals:

$$\min_{\{\lambda_k\}} \sum_{w} \left\| m_w - \sum_{k=1}^{K} \lambda_k m_{s_k} \right\|_F^2 \quad \text{s.t.} \quad \forall k, \lambda_k \geq 0, \quad \text{and} \quad \sum_{k=1}^{K} \lambda_k = 1. \tag{1}$$

We reparametrize this constrained convex quadratic program by

$$\lambda_k = \exp(\hat{\lambda}_k) / \sum_{k=1}^{K} \exp(\hat{\lambda}_k). \tag{2}$$

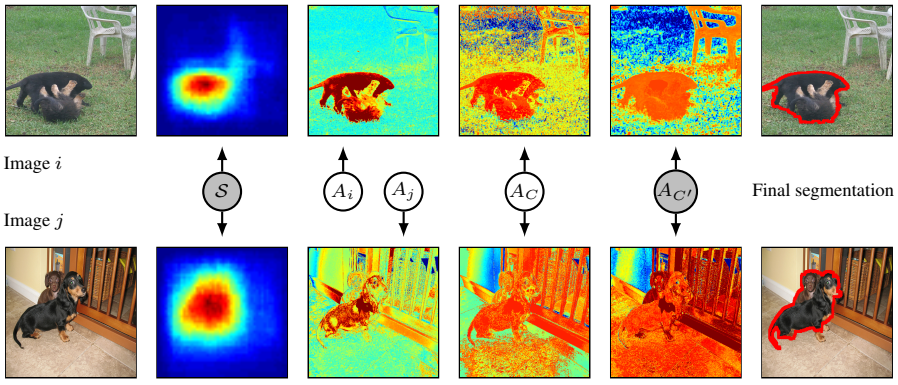

Fig. 4. *Our joint segmentation model. Left: two images i and j of a class to segment. The location priors M_i and M_j are obtained by segmentation transfer from $\mathcal{S}$ (second column). Image models A_i and A_j contribute to an image-specific unary potential (third column). The fourth column shows the class-wide unary potential (A_C) applied to these two images. The fifth column uses the appearance model $A_{C'}$ of a related class C' on these two images. Gray nodes represent fixed models, while white nodes illustrate models that are updated during the iterations of the energy minimization. Unary potential are represented by mapping the most likely background pixel to blue and the most likely foreground pixel to red. Rightmost column: final segmentations produced by our model.*

We solve the resulting unconstrained program using Matlab's `fminunc`, based on an interior-point algorithm. As the weights decrease rapidly, we automatically set K to the rank of the first neighbor with near zero weight (i.e. $K = 10$).

5 Joint segmentation of a set of images

Thanks to the technique of sec. 4, each image i of a class C in the target set $\mathcal{T}_t$ now has a transferred soft-segmentation mask M_i. In this section we focus on the next processing stage, *i.e.* producing a binary segmentation of all images in C jointly. We model this task in a classic energy minimization framework [1, 2] which we extend to multiple unary potentials, including image-specific and class-wide appearance terms (sec. 5.1), a location prior derived from M, and appearance terms imported from semantically related classes (sec. 5.2, used from stage $t = 3$ onward). In sec. 5.3 we learn the optimal weights of all potentials in the model by solving a structured SVM problem [13].

5.1 Sharing appearance within a class

Given a set of images $i \in \mathcal{I}$, let x_{ip} be the label for pixel p in image i and $\mathbf{x}$ be the vector of all x_{ip}. In this paper $\mathcal{I}$ contains all images of a class C of ImageNet. The energy function for jointly segmenting all images in $\mathcal{I}$ using the source pool $\mathcal{S}$ is

$$E(\mathbf{x}; \mathcal{A}, \mathcal{S}) = \sum_i \left(\sum_p E_{ip}(x_{ip}; \mathcal{A}, \mathcal{S}) + \sum_{p,q} E_{ipq}(x_{ip}, x_{iq}) \right) \quad (3)$$

The pairwise potential is

$$E_{ipq}(x_{ip}, x_{iq}) = \delta(x_{ip} \neq x_{iq}) \cdot d(i,p,q)^{-1} \cdot \exp(-\gamma ||c_{ip} - c_{iq}||^2) \quad (4)$$

Analog to [1, 26, 32–34], this potential encourages smoothness by penalizing neighboring pixels taking different labels. The penalty depends on the color contrast between the pixels, being smaller in regions of high contrast (image edges). The summation over (p, q) is defined on a 8-connected pixel grid.

The unary potential is a linear combination of several terms

$$E_{ip}(x_{ip}; \mathcal{A}, \mathcal{S}) = -\alpha_I \log p(x_{ip}; c_{ip}, A_i) - \alpha_C \log p(x_{ip}; c_{ip}, A_C) - \alpha_M \log M_{ip}(x_{ip}; \mathcal{S}) \quad (5)$$

Each potential $p(x_{ip}; c_{ip}, A)$ evaluates how likely a pixel of color c_{ip} is to take label x_{ip}, according to the appearance model A. The set of appearance models $\mathcal{A}$ contains one model A_i specific to each image and one class model A_C common to all images in $\mathcal{I}$. This class model enables to share appearance among the images, so they are jointly segmented. The image-specific models account for visual characteristics unique to an image (*e.g.* the hair color of a soccer player), while the class model accounts for class-wide characteristics (*e.g.* the color of the team's shirt). As in [1], an appearance model A consists of two Gaussian mixture models (GMM), one for the foreground (used when $x_{ip} = 1$) and one for the background (used when $x_{ip} = 0$). Each GMM has 5 components. Each component is a full-covariance Gaussian over the RGB color space.

Because of the probabilistic nature of M_{ip} as obtained from sec. 4, we can directly use $M_{ip}(x_{ip}; \mathcal{S}) = M_{ip}^{x_{ip}}(1 - M_{ip})^{1-x_{ip}}$ as a unary potential in eq. (5). Figure 4 illustrates the various unary potentials.

Our joint segmentation model can be seen as a generalization of both Grabcut [35] and Batra et al. [36]. In Grabcut each image is segmented independently, based on an appearance model for each image: $\mathcal{A} = \{A_i\}_{i \in \mathcal{I}}$. Conversely, Batra et al. [36] uses only a single model shared among all images: $\mathcal{A} = \{A_G\}$. Both [35, 36] and other works using analog energy functions [26, 37] require user interaction to estimate the appearance model, typically a manually drawn bounding-box or scribbles. In our work instead, following [2], the appearance models are *automatically* estimated from the mask M_i. After this initial estimation, we follow [35] and alternate between finding the optimal segmentation $\mathbf{x}$ given the appearance models, and updating the appearance models given the segmentations. The first step is solved globally optimally by minimizing eq. (5) using graph-cuts as our pairwise potentials are submodular. The second step fits GMMs to labeled pixels. Each image model A_i is fitted to the current segmentation of the respective image, while a single global model A_C is fitted to the segmentations of all images at the same time. The benefits of having A_C can be understood in the light of this iterative scheme. The class model can be more robustly estimated from all images, as the errors due to inaccurate segmentations average out. In turn this more accurate appearance model helps improving segmentations in the next iteration.

Propagation scheme. The general scheme above is adapted at each stage t to fit the situation (sec. 3). At stage 1 we constrain the solution of eq. (3) to the available ground-truth bounding-box. At stage 2, when segmenting unannotated images in the

same classes as stage 1, we include the images of stage 1 in eq. (3) but keep their segmentation fixed to the output of stage 1. This way they can improve the segmentation of new images by contributing to the class model A_C.

5.2 Importing appearance from related classes

From stage $t = 3$ onward, the propagation wave reaches new target classes $\mathcal{T}_t$ which are *semantically related* to the source classes in $\mathcal{S}_{t-1}$ (see sec. 3). As these related classes have been already segmented in the previous stage, we propose to import their appearance models to help segmenting the new classes. This idea is related to *knowledge transfer* for object classification [38], localization [39] and detection [40], but we believe it is unexplored for segmentation.

More precisely, when segmenting a new class C, we add to eq. (3) a unary potential for each of its related classes $C' \in \mathcal{R}(C)$, which carries its appearance model $A_{C'}$. We therefore extend the unary potentials in eq. (5) to

$$\begin{aligned} E_{ip}(x_{ip}; \mathcal{A}, \mathcal{R}(C)) = &- \alpha_I \log p(x_{ip}; c_{ip}, A_i) - \alpha_C \log p(x_{ip}; c_{ip}, A_C) \\ &- \alpha_M \log M_{ip}(x_{ip}; \mathcal{R}(C)) - \frac{\alpha_R}{|\mathcal{R}(C)|} \sum_{C' \in \mathcal{R}(C)} \log p(x_{ip}; c_{ip}, A_{C'}) \end{aligned} \tag{6}$$

Note how the related source classes all have the same weight α_R, instead of their own specific weight $\alpha_{C'}$. As the number of related source classes varies for each target class, it is very difficult to learn a weight per related model (sec. 5.3).

Note how in eq. (6) we restrict the source pool used for segmentation transfer to $\mathcal{R}(C)$, to make it maximally related to C (as discussed in sec. 3).

5.3 Learning the weights $\boldsymbol{\alpha}$

We learn the weights $\boldsymbol{\alpha}$ of the unary potentials using some manually segmented images from ImageNet. We train two weight vectors $\boldsymbol{\alpha} = \{\alpha_I, \alpha_C, \alpha_M\}$ specific to stage 1 and 2 respectively, and one generic weight vector $\boldsymbol{\alpha} = \{\alpha_I, \alpha_C, \alpha_M, \alpha_R\}$ for all later stages. This involves a total of only 90 segmented training images.

Let $\mathbf{x}_i$ be the labeling of all pixels in image i. Given n training images $\mathcal{I} = (i_1, \ldots, i_n)$ with associated ground-truth labelings $\mathbf{x}^* = (\mathbf{x}_1^*, \ldots, \mathbf{x}_n^*)$, we seek for the parameters so that the energy of the ground-truth labeling $\mathbf{x}_i^*$ of each image is lower than the energy of any other labeling $\mathbf{x}_i$ of that image, assuming fixed models $\mathcal{A}$ and source pool $\mathcal{S}$. This translates to the following contraints

$$E(\mathbf{x}_i^* | i, \boldsymbol{\alpha}) \leq E(\mathbf{x}_i | i, \boldsymbol{\alpha}), \quad \forall \mathbf{x}_i \neq \mathbf{x}_i^*, \quad \forall i \in \mathcal{I}. \tag{7}$$

where $E(\mathbf{x}|i, \boldsymbol{\alpha})$ is one term in the outermost summation of eq. (3), corresponding to only image i. For simplicity, we omit $\mathcal{A}$ and $\mathcal{S}$ as they are predetermined by the segmentation transfer process and cannot change during inference on eq. (3).

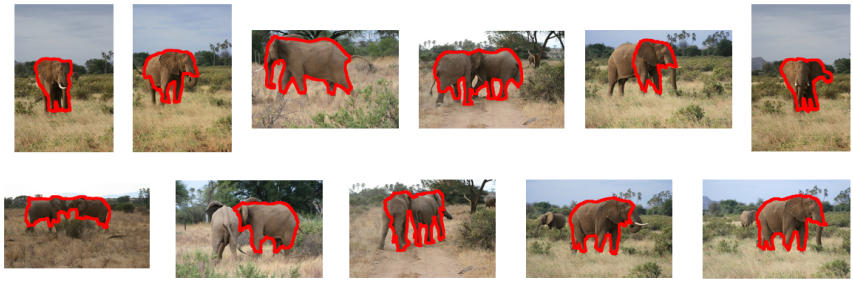

Fig. 5. *Segmentations produced by our image+transfer+class method on the* Elephants *class of the iCoseg dataset.*

To learn the parameters $\boldsymbol{\alpha}$ we solve a generalized support vector machine training problem, following [13]

$$\min_{\boldsymbol{\alpha},\xi} \frac{1}{2}\|\boldsymbol{\alpha}\|^2 + C\sum_{j=1}^{n}\xi_i \tag{8}$$
$$\text{s.t.} \quad E(\mathbf{x}_i|i,\boldsymbol{\alpha}) - E(\mathbf{x}_i^*|i,\boldsymbol{\alpha}) \geq \Delta_i(\mathbf{x}_i^*,\mathbf{x}_i) - \xi_i, \quad \forall \mathbf{x}_i \neq \mathbf{x}_i^*$$
$$\xi_i \geq 0, \quad \forall i \in \mathcal{I}$$

where $C > 0$ is a constant; ξ_i is the slack variable for $\mathbf{x}_i$, which is necessary if no $\boldsymbol{\alpha}$ fulfilling all constraints exists; $\Delta_i(\mathbf{x}_i^*,\mathbf{x})$ is a loss function quantifying the difference between a labeling $\mathbf{x}_i$ and the ground-truth $\mathbf{x}_i^*$.

Our choice for Δ_i is the average number of mislabelled pixels, weighted to account for the ratio of foreground/background pixels in the image: $\Delta_i(\mathbf{x}_i^*,\mathbf{x}_i) = \sum_{p\in i} w_{ip}|x_{ip} - x_{ip}^*|$, where $w_{ip} = 1/n_i^+$ if x_{ip}^* is foreground and $w_{ip} = 1/n_i^-$ otherwise; n_i^+, n_i^- are the number of ground-truth foreground/background pixels in i. In essence, this weighted loss gives equal importance to foreground and background regions, thus avoiding biases towards the background which often occupies most of an image.

As each labeling $\mathbf{x}_i$ corresponds to a constraint, the number of constraints is exponential in the number of pixels. Constraint generation circumvents this issue by iteratively solving (8) while updating a set of most violated constraints. Finding the most violated constraint for an image i involves minimizing $E(\mathbf{x}_i|i,\boldsymbol{\alpha}) - \Delta_i(\mathbf{x}_i^*,\mathbf{x}_i)$. Since Δ_i can be expressed as a unary potential over pixels, this problem can be solved exactly using graph-cut [14].

6 Experiments and conclusion

We validate in sec. 6.1 the components of our approach on the recent iCoseg dataset [36], and then present results on ImageNet in sec. 6.2. We conclude in sec. 6.3.

6.1 Cosegmentation on iCoseg

iCoseg [36] contains 643 images grouped in 38 classes (e.g. stonehenge, brown bear, gymnasts, airplanes). The task, as set out by previously published works [15, 16, 36] is

	[16]	[15]	image only ≈GrabCut [1]	class only ≈Batra [36]	image+class	image+transfer ≈Kuettel [2]	image+transfer +class
Accuracy	78.9	85.4	82.4	83.6	88.2	87.6	**91.4**

Table 1. *Segmentation accuracy on iCoseg. The results for [15, 16] are taken from table 1 in [15]. Columns 3-6 are stripped down versions of our model. The last column is our complete model (see main text).*

to jointly segment the foreground object in all images of a class. Following these works, we measure performance as the percentage of correctly labelled pixels (*accuracy*).

In tab. 1 we compare several stripped down versions of our model (sec. 5.1). The first three use no segmentation transfer (sec. 4) and initialize their appearance models from a window centered on the image taking 25% of its surface. **(1) image only**: using only the image-specific unary potential A_i. This is essentially identical to GrabCut [1]; **(2) class only**: using only the class-wide unary potential A_C. This is very similar to [36], but without user interaction; **(3) image+class**: using both types of unaries; **(4) image+transfer**: using the image-specific unary A_i and segmentation transfer to initialize the appearance models and to add a location prior unary potential M (sec. 4). The source pool is fixed to the PASCAL VOC10 training set. This is a computationally efficient version of [2] using the speedups we proposed in sec. 4.2. As reported in [2], it obtains state-of-the-art figure-ground segmentation performance on PASCAL VOC10. **(5) image+transfer+class**: using image-specific unaries, class-wide unaries, and segmentation transfer with source pool fixed to the VOC10 training set. For the models using multiple unary potentials (3-5), we use the technique in sec. 5.3 to learn their weights $\boldsymbol{\alpha}$ in a leave-one-class-out fashion. When evaluating a class, we use weights learned from two random images from each of the other 37 classes.

As table 1 shows, the baseline GrabCut already shows a good performance (82.4% accuracy). On iCoseg, using class models proves very beneficial, because the object instances in different images of a class have very similar appearance. Class models alone perform better than image models (83.6%), and greatly improve the performance when combined with other models: $+5.8\%$ with image models, $+3.8\%$ with image models and segmentation transfer (fig. 5). Segmentation transfer [2] also proves very useful: it improves by $+5.2\%$ over GrabCut using image models only, and by $+3.2\%$ with both image and class models. This shows that segmentation transfer is a very effective way to automatically initialize GrabCut, confirming what observed by [2] on other datasets (PASCAL VOC10, Graz-02, Weizmann horses). Here, we cannot evaluate the idea of recursively updating the source pool (sec. 3) nor of importing appearance models from related classes (sec. 5.2), as classes in iCoseg are not organized in a hierarchy.

Table 1 also reports the average accuracy of two recent state-of-the-art works [15, 16] as reported in [15]. In a comparable setting using only iCoseg images, our image+class method outperforms them both. Our image+transfer+class method performs best by a considerable margin, but it uses manually segmented PASCAL VOC10 images as training data. Importantly, our method is also computationally much more efficient than [15, 16]. It takes only minutes to segment a class, in contrast to several hours for [15, 16]. Hence, we can apply our technique to the much larger ImageNet dataset.

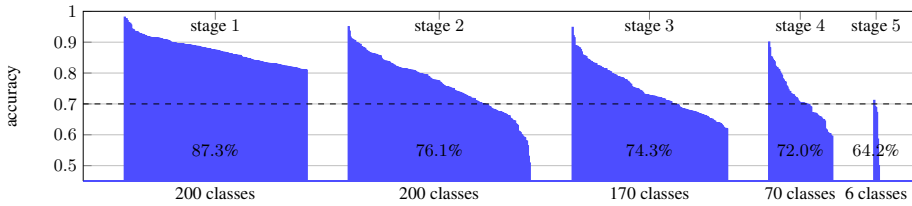

Fig. 6. *ImageNet: segmentation accuracy for all classes of each stage of our propagation pipeline. For each stage we show the classes sorted by their accuracy averaged over images. Note how stage 1 and stage 2 operate on the same classes, but on different images (images in stage 1 have annotated bounding-boxes, whereas images in stage 2 do not). Each subsequent stage operates on new classes containing only images without bounding-boxes (fig. 2).*

6.2 Segmentation propagation on ImageNet

We have run our full segmentation propagation method on two subtrees of ImageNet containing about 500k images over 577 classes. We selected the classes automatically to ensure that about half of the classes have some images annotated by bounding-boxes, while half of the classes have none. In total, there are 60k images with bounding-boxes and 440k images with only class labels. On this subset of ImageNet, segmentation propagation runs for 5 stages to completion. To quantitatively evaluate our approach, we annotated segmentations via Amazon Mechanical Turk for 10 random images from 446 classes, for a total of 4460 images. These annotations enable to reliably estimate the segmentation performance of our method on a wide range of classes. We have also annotated a small separate set of 90 images to estimate $\boldsymbol{\alpha}$, as discussed in sec. 5.3.

The segmentation accuracy of our full method averaged over all images in the evaluation set is 77.1%. In comparison, the baseline GrabCut delivers 71.0% (as in 'image only' in sec. 6.1). Fig. 6 reports per-class accuracy for all 446 classes, divided by the stage by which they are reached by our propagation wave (sec. 3). Interestingly, accuracy degrades gracefully over the stages. The average accuracies for stage 1 to 5 are 87.3%, 76.1%, 74.3%, 72.0%, and 64.2% (see fig. 1 for examples). Note how stage 5 contains very few images and so has little impact of the overall average performance.

Stages 2-5 are interesting because their source pools contain many (imperfect) segmentations produced by earlier stages rather than only the ground-truth masks $\mathcal{S}_0$ from VOC10. This enables to test the effect of the full segmentation propagation idea, compared to segmentation transfer from the fixed $\mathcal{S}_0$ pool. For this we compare to our image+transfer+class method, including also class-wide appearance models (sec. 6.1). The accuracy of our full segmentation propagation method differs from image+transfer+class by +3.6% in stage 2, by +0.7% in stage 3, by 0% in stage 4 and by -5.9% in stage 5. On average over all images in stages 2-5, the improvement is +1.2% (from 72.9% to 74.1%). This demonstrates the value of recursively employing images segmented before to help segmenting new images. However, the progressive decay of improvement over stages indicates that errors in early stages propagate to later stages.

Finally, we notice that the visual variability in ImageNet classes is huge. As a consequence, the weights $\boldsymbol{\alpha}$ learned on ImageNet are quite different from the ones learned on iCoseg. Typically, class models in iCoseg perform very well and have high weight. On the contrary, class and related models have lower weights in ImageNet. This stresses the value of learning these weights automatically rather than setting them manually.

6.3 Conclusion

We have presented *segmentation propagation*: a computationally efficient technique to recursively segment images in ImageNet. It successfully combines ideas from segmentation transfer, cosegmentation, structured output learning, efficient binary codes, and GrabCut. The technique was shown to segment 500k images over 577 ImageNet classes with good accuracy. We have shown how accuracy degrades gracefully as the propagation waves moves from easier images with bounding-box annotations, to unannotated images in the same classes, to images in completely unannotated classes. We have also demonstrated the value of the various components of our method on the smaller iCoseg dataset [36], where it outperforms the state-of-the-art in cosegmentation [15].

In future work, we plan on exploiting the fact that classes in ImageNet are very diverse. Some have more images than others and some have much larger variations in appearance than others. This suggests to adapt the segmentation technique to each target class, and to propagate segmentations based on *visual* similarity between classes, rather than only based on *semantic* similarity. To improve robustness we plan to automatically detect bad segmentations in early stages, to avoid propating errors to later stages. This could be achieved, e.g. by analysing the entropy of the transfer mask M, as a measure of the confidence of the method (fig. 3d).

References

1. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. In: SIGGRAPH. (2004)
2. Kuettel, D., Ferrari, V.: Figure-ground segmentation by transferring window masks. In: CVPR. (2012)
3. Chai, Y., Lempitsky, V., Zisserman, A.: Bicos: A bi-level co-segmentation method for image classification. In: ICCV. (2011) 2579–2586
4. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV **59** (2004) 167–181
5. Tu, Z., Chen, X., Yuille, A., Zhu, S.: Image parsing: Unifying segmentation, detection, and recognition. IJCV **63** (2005) 113–140
6. Shotton, J., Blake, A., Cipolla, R.: Contour-Based Learning for Object Detection. In: ICCV. (2005)
7. Jiang, H.: Human pose estimation using consistent max-covering. In: ICCV. (2009)
8. Gong, Y., Lazebnik, S.: Iterative quantization: A procrustean approach to learning binary codes. In: CVPR. (2011)
9. Torralba, A., Fergus, R., Weiss, Y.: Small codes and large image databases for recognition. In: CVPR. (2008)
10. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-fei, L.: ImageNet: A large-scale hierarchical image database. In: CVPR. (2009) `http://image-net.org/`.
11. Rosenfeld, A., Weinshall, D.: Extracting foreground masks towards object recognition. In: ICCV. (2011)
12. Batra, D., Kowdle, A., Parikh, D., Luo, J., Chen, T.: Interactively co-segmentating topically related images with intelligent scribble guidance. IJCV (2011)
13. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. JMLR **6** (2005) 1453–1484
14. Szummer, M., Kohli, P., Hoiem, D.: Learning CRFs using graph cuts. In: ECCV. (2008)

15. Vicente, S., Rother, C., Kolmogorov, V.: Object cosegmentation. In: CVPR. (2011) 2217–2224
16. Joulin, A., Bach, F., Ponce, J.: Discriminative clustering for image co-segmentation. In: CVPR. (2010) 1943–1950
17. Borenstein, E., Sharon, E., Ullman, S.: Combining top-down and bottom-up segmentation. In: CVPR. (2004)
18. Jojic, N., Perina, A., Cristani, M., Murino, V., Frey, B.: Stel component analysis: Modeling spatial correlations in image class structure. In: CVPR. (2009)
19. Bertelli, L., Yu, T., Vu, D., Gokturk, S.: Kernelized structural SVM learning for supervised object segmentation. In: CVPR. (2011)
20. Winn, J., Jojic, N.: LOCUS: learning object classes with unsupervised segmentation. In: ICCV. (2005)
21. Arora, H., Loeff, N., Forsyth, D., Ahuja, N.: Unsupervised segmentation of objects using efficient learning. In: CVPR. (2007)
22. Verbeek, J., Triggs, B.: Region classification with Markov field aspect models. In: CVPR. (2007)
23. Alexe, B., Deselaers, T., Ferrari, V.: What is an object? In: CVPR. (2010)
24. Carreira, J., Sminchisescu, C.: Constrained parametric min cuts for automatic object segmentation. In: CVPR. (2010)
25. Schoenemann, T., Cremers, D.: Introducing curvature into globally optimal image segmentation: Minimum ratio cycles on product graphs. In: ICCV. (2007)
26. Blake, A., Rother, C., Brown, M., Perez, P., Torr, P.: Interactive image segmentation using an adaptive GMMRF model. In: ECCV. (2004)
27. Kim, G., Xing, E., Fei-Fei, L., Kanade, T.: Distributed cosegmentation via submodular optimization on anisotropic diffusion. In: ICCV. (2011) 169–176
28. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: CVPR Workshop of Generative Model Based Vision. (2004)
29. Lampert, C., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: CVPR. (2009)
30. Ott, P., Everingham, M.: Shared parts for deformable part-based models. In: CVPR. (2011)
31. Norouzi, M., Punjani, A., Fleet, D.J.: Fast search in hamming space with multi-index hashing. In: CVPR. (2012)
32. Vicente, S., Kolmogorov, V., Rother, C.: Graph cut based image segmentation with connectivity priors. In: CVPR. (2008)
33. Shotton, J., Winn, J., Rother, C., Criminisi, A.: TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: ECCV. (2006)
34. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: ICCV. (2001)
35. Rother, C., Kolmogorov, V., Minka, T., Blake, A.: Cosegmentation of image pairs by histogram matching - incorporating a global constraint into MRFs. In: CVPR. (2006)
36. Batra, D., Kowdle, A., Parikh, D., Luo, J., Chen, T.: iCoseg: Interactive co-segmentation with intelligent scribble guidance. In: CVPR. (2010) 3169–3176
37. Wang, J., Cohen, M.: An iterative optimization approach for unified image segmentation and matting. In: ICCV. (2005)
38. Tommasi, T., Orabona, F., Caputo, B.: Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In: CVPR. (2010)
39. Guillaumin, M., Ferrari, V.: Large-scale knowledge transfer for object localization in ImageNet. In: CVPR. (2012)
40. Salakhutdinov, R., Torralba, A., Tenenbaum, J.: Learning to share visual appearance for multiclass object detection. In: CVPR. (2011)