ImageNet Auto-Annotation with Segmentation Propagation

ImageNet Auto-annotation with Segmentation Propagation

Matthieu Guillaumin · Daniel Küttel · Vittorio Ferrari

Abstract ImageNet is a large-scale hierarchical database of object classes with millions of images. We propose to automatically populate it with pixelwise object-background 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 experiments 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, especially when implemented on superpixels; (iv) outperforms a baseline GrabCut [52] initialized on the image center, as well as segmentation transfer from a fixed source pool and run independently on each target image [37]. Moreover, our method also delivers state-of-the-art results on the recent iCoseg dataset for co-segmentation.

Keywords figure-ground segmentation · ImageNet · knowledge transfer · object localization · large-scale computer vision

Matthieu Guillaumin, Daniel Küttel
ETH Zürich, D-ITET, Computer Vision Laboratory
Sternwartstrasse 7, CH-8092 Zürich
E-mail: {guillaumin,kuettel}@vision.ee.ethz.ch

Vittorio Ferrari
University of Edinburgh, IPAB, School of Informatics
Crichton street 10, Edinburgh EH8 9AB, UK
E-mail: vferrari@staffmail.ed.ac.uk

1 Introduction

Foreground-background segmentation is the fundamental task of producing a binary segmentation of an image, separating the foreground object from the background [52,14]. Segmentation is useful in many higher-level applications such as object recognition, as it provides a spatial support for extracting texture and shape descriptors on objects [66,58]. It is also valuable for human pose estimation, where silhouettes have been shown to reliably convey pose [32], and for 3D reconstruction from silhouettes. However, manually annotating images with segmentations is tedious and very time consuming. This prevents the above applications from scaling 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 [27] and object classification [64].

In this paper, we want to bridge the gap between these domains by automatically populating the large-scale ImageNet [19] database with foreground segmentations (fig. 12). ImageNet[1] contains millions of images annotated by the class label of the main object. 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. 12. Our work weaves together and extends several recent developments including Grabcut [52], segmentation transfer [51,37], efficient binary codes [27], cosegmentation [14,8] and structured output learning [65, 61] 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 appear-

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

ance models from related classes that were segmented in previous stages of our segmentation propagation process.

1.1 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. 3). 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. 1). 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 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. 5.3). 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 contains perfect, manual foreground-background segmentations, but of potentially irrevelant object classes from PASCAL VOC. On the other hand, semantically related classes are relevant sources for segmentation transfer, but the corresponding segmentations are automatically generated by the propagation and are thus imperfect. 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.

1.2 Plan of the paper and overview of experiments

We review related work in sec. 2 and then detail the components of our approach in sections 3 to 5. In sec. 3 we describe the segmentation transfer paradigm and how we extend it to make it suitable for large-scale applications. Then, sec. 4 describes how to employ the transferred mask to guide the segmentation of each image independently by minimizing an energy function analog to GrabCut. Section 5 extends the energy function to segment all the images in a class *jointly*. This include sharing appearance models within the class (sec. 5.1) and importing appearance models of related classes from the source pool (sec. 5.3).

In sec. 6, we present experimental results. First, we validate the components of our approach on the smaller iCoseg dataset, we compare it to several existing works [8,34,71, 45] and achieve state-of-the-art performance (sec. 6.1). Next, we show that our process accurately segments 500k images over 577 classes of ImageNet (sec. 6.2). To our knowledge, this is the largest segmentation experiment to date. We compare our results to several relevant alternatives, including: (a) a baseline GrabCut [52] initialized on the image center; this was shown to be a competitive baseline on several datasets, such as Weizman horses [2], CalTech 101 [2] and iCoSeg (sec. 6.1); (b) a simpler segmentation transfer technique based on global image similarity instead of windows; (c) our recent segmentation transfer technique [37] on which

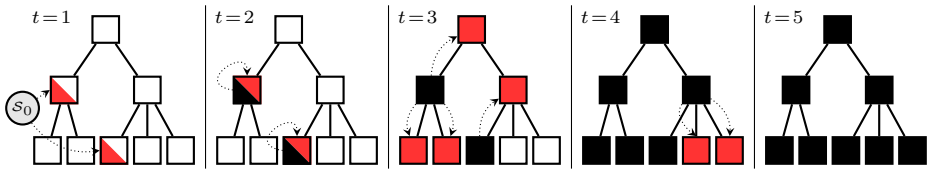

Fig. 1 *Illustration of segmentation propagation on ImageNet. The stage of propagation is marked by t. Nodes are classes and edges represent the class hierarchy. Node colors indicate the state of a class: white = "unsegmented", red = "currently being segmented" ($\mathcal{T}_t$), and black = "already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

this work is based. It keeps the source pool $\mathcal{S}_0$ fixed to PASCAL VOC 2010 and does not include any propagation element nor sharing appearance models between images. Finally, we draw conclusions in sec. 7.

To promote applications, we have released all our ImageNet segmentations online[2]. This paper is an extension of our preliminary works [37,38]. It includes an accelerated segmentation model based on superpixels, additional experimental results for in-depth analysis, and more detailed explanations of the method.

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 [11,33,9]). They are supervised in that the training set shows images of other instances of the class along with their binary segmentations. Several works have attempted to reduce the burden of annotating images with ground-truth segmentations. The degree of supervision is typically reduced by providing only the class names of the object appearing in the image [73, 4], and sometimes by annotating only a fraction of the pixels [69]. Our work is related to this, as most of the images in ImageNet are only labeled by class names.

Another related recent trend is to guide the segmentation process with class-generic techniques to propose candidate regions likely to contain objects of any class [3,13,56,24], as in [37,71,41]. As spatial support for our segmentation transfer operations, we use the candidate windows detected by the 'objectness' technique of [3]. However, other methods to obtain such candidates [56] could form a valid alternative, as long as they are fast to compute so they can be applied at the ImageNet scale.

Interactive segmentation [52,57,10] has been thoroughly researched since the very popular GrabCut [52]. Most of these approaches minimize a binary pairwise energy function whose unary potentials are determined by appearance models, in the form of pixel color distribution, 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 [14,15, 8,71,34,35,45]. Similar to [14,15,8], we share appearance models when segmenting many images of the same class. This sharing helps to identify which image regions belong to the foreground object.

Annotation transfer by nearest neighbours. Our method transfers segmentation masks from windows in the source pool to visually similar windows in a new target image. This is related to works that transfer annotations between images based on their global similarity, [51,43,30,31,62,54] as done in inpainting [31], image tagging [30], object class detection [54], and scene parsing [43,62]. Malisiewicz et al. [44] proposes to employ per-exemplar SVMs to find neighbours for transfer, instead of simply measuring appearance similarity. Rosenfeld et al. [51], transfers segmentation masks between images based on their global similarity, for the task of figure-ground segmentation. Recently we [37] improved on their scheme by transferring segmentation masks at the level of windows (using [3] to define windows likely to be centered on objects). We build our work on this segmentation transfer scheme, but make it computationally much more efficient to scale up to ImageNet. As we recap in sec. 3.1, (object) windows offer better spatial support for segmentation transfer than whole images.

Transfer learning. Our work is related to previous works on transfer learning in computer vision, where learning a new class (target) is helped by labeled examples of other related classes (sources) [5,6,25,40,48–50,55,60,63,22,28]. Most of these works try to reduce the number of examples necessary to learn the target, improving generalization from a few examples. Many methods use the parameters of the source classifiers as priors for the target model [5,6,25,55, 63]. Other works [40,50] transfer knowledge through an intermediate attribute layer, which captures visual qualities shared by many object classes (e.g. 'striped', 'yellow'), or through prototypes [49]. A third family of works transfer object parts between classes [6,48,60], such as wheels between cars and bicycles or legs between cows and horses.

[2] Website: `http://www.vision.ee.ethz.ch/~mguillau/imagenet.html?calvin`

Finally, [22,28] employ the knowledge transferred from the source classes to reduce the degree of supervision necessary to learn object class detectors from bounding-boxes to just image labels.

The above works aim at image classification or object detection, not segmentation. For segmentation, we propose to use appearance models of previously segmented classes to help segmenting a new class. Our segmentation propagation scheme automatically determines which classes to segment next.

ImageNet. ImageNet [19] is a large-scale hierarchical database of images. ImageNet forms a directed acyclic graph (DAG) where the classes are vertices linked by directed edges that represent parent-child relations: *Aircraft* is a parent of *Airplane* because an airplane *is an* aircraft, along with helicopters, etc. Currently, ImageNet contains about 15 million images of 22.000 classes. Its large scale, accurate annotation of all images by the class of the main object they contain, and the connections in the semantic hierarchy, make ImageNet a great resource for computer vision research and the ideal playground for experimenting with knowledge transfer ideas. However, currently only a small fraction of the images is annotated with bounding-boxes, and none with foreground segmentation.

There is a growing body of work which uses ImageNet. Several works tackle image classification [18,42,21,20,36, 17] or object detection in the fully supervised setting [17]. Deselaers and Ferrari study the relation between appearance similarity and semantic similarity [23]. Guillaumin and Ferrari [28] populate about 500k images of ImageNet with object bounding-boxes automatically derived by transferring knowledge from images with ground-truth bounding-box annotations. To our knowledge, ours is the first work trying to automatically populate ImageNet with object *segmentations*.

3 Large-scale segmentation transfer

We present here the paradigm of *segmentation transfer* [37, 51], 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 similar images in the source pool $\mathcal{S}$ of pre-segmented images. The transferred masks are then used to derive the unary potentials of an energy function which is minimized to refine the segmentation (sec. 5).

Fig. 3 *An example to demonstrate the advantage of window-level segmentation transfer over global transfer. In both cases the transferred mask M is used to guide a GrabCut-like segmentation process of section sec. 4.1. The two methods differ in how M is obtained.*

3.1 Window-level segmentation transfer

The basic scheme [51] 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 for i. However, often the most similar source images have quite different figure-ground segmentations than i. This happens because there is too much variability at the level of the whole image, so typically there are no source images which are globally similar and have similar objects at the correct position and size.

Recently, we have improved on the basic scheme by transferring segmentation masks at the level of *windows* [37] (fig. 2a). In each image, we first extract 100 candidate windows using the 'objectness' technique of [3], and then transfer masks from windows in $\mathcal{S}$ (fig. 2b) to visually similar windows in i (fig. 2c). The objectness sampling tends to return more windows centered on an object with a well-defined boundary in space, such as cows and cars, rather than amorphous background elements, such as grass and sky. These windows make a better spatial support for segmentation transfer, as they exhibit less variability than whole images, while at the same time containing enough distinctive information. This leads to retrieving much better neighbours, whose segmentation masks better match the target image. As another important advantage, window-level segmentation transfer enables to compose novel scenes using local parts from different source images (e.g. the source images have either a cow or a dog, while the target image has a cow *and* a dog). Finally, as the objectness window sampling is covariant to translation and scale, the segmentation transfer process can relocate objects to the appropriate position in the target image (e.g. all source images have a dog in the center, while the target image has a dog in the top-left corner).

After transferring masks for each window independently (fig. 2c), they are aligned to their corresponding windows in i and aggregated into a single mask M_i (fig. 2d, see sec. 3.3). The window masks are first translated and rescaled to their appropriate image location and then M_i is defined as their pixelwise mean. Hence, $M_{ip} \in [0,1]$ estimates the prob-

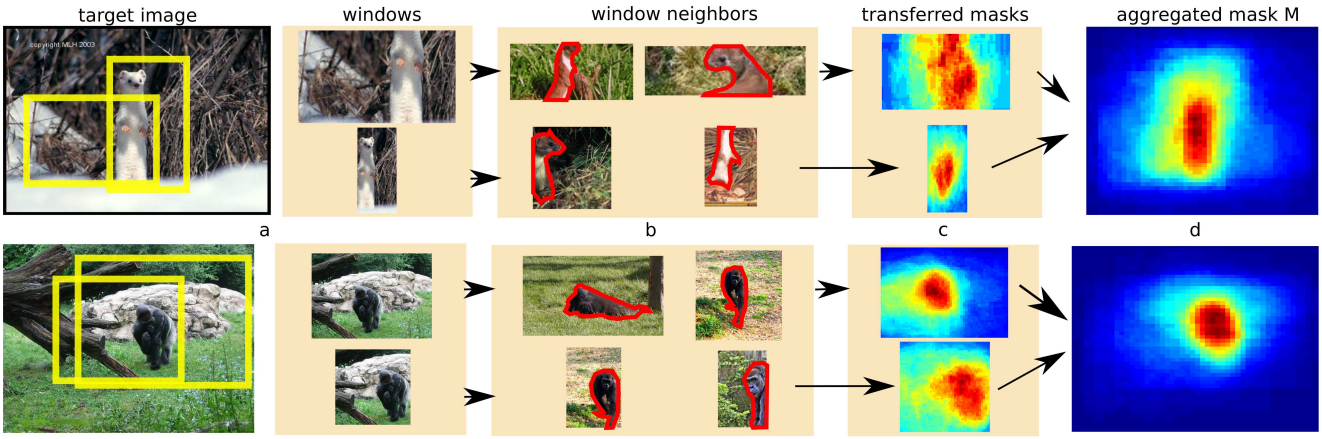

Fig. 2 *Two examples of window-level segmentation transfer at stage 3. (a) two out of 100 windows extracted in a target image; (b) the most similar windows from the source set $\mathcal{S}_2$ transfer their segmentation masks (outlined in red) to the windows of the target image, giving (c); (d) the 100 individual window masks are aggregated into a single soft-segmentation mask M for the target image.*

ability that the pixel p is foreground in image i (fig. 2d). M_i is then used in two different ways in our energy minimization framework (sec. 4). First, they automatically set 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. In this fashion, while segmentation transfer operates on individual windows, the energy minimization step integrates local evidence from all windows into a coherent global segmentation of the target image (sec. 4.1). Figure 3 shows the benefit of our segmentation transfer based on windows, compared to based on global image neighbours.

3.2 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 [64], 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[47] as used in [51, 37], we use HOG[16]. In our experiments, it as accurate while being much faster to compute. 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 a typical source pool $\mathcal{S}$ containing 10k images, 100M distance computations are needed to segment a single target image! Moreover, storing the HOG descriptors for all 100M windows in the 1M images in an ImageNet scale experiment would require 3.1 TB of disk space. This cannot fit the memory of a computer, and reloading the part of it corresponding to the source pool of each target image is even slower than computing the distances. This makes window-level large-scale segmentation transfer essentially infeasible.

In this paper we employ the efficient binary coding scheme called "Iterative Quantization" (ITQ) [27] 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 128 bits (*i.e.* 16 bytes) to encode each HOG, 100M windows now account for a mere 1.6 GB, *i.e.* about $2000\times$ less memory. 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 70 million distances computations per second (on a single core of an Intel Core i7 CPU 923 2.67GHz). This is about $350\times$ faster than directly computing the distance between the original HOG descriptors. In practice, it takes only about 1.5 seconds to do segmentation transfer for a typical target image, which has 10k images in its source pool. 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 [46].

A natural question is whether the binary encoding causes any loss in segmentation transfer performance. We investigated this on the PASCAL VOC10 dataset, using as a source the training subset of the challenge, and as target images the validation set. Fig. 4 shows the intersection-over-union segmentation performance when describing windows in the original HOG space, and as a function of the number of bits in the encoding. As the plot shows, the performance is essentially unchanged when using 1024 bits, and there is only

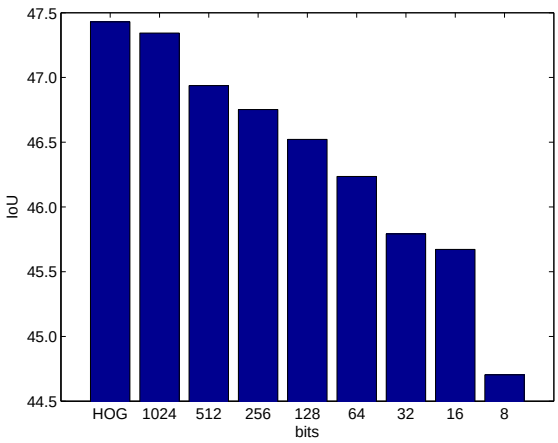

Fig. 4 *We conducted experiments on the pascal VOC10 challenge with varying binary code sizes. We measure the IoU score of the final segmentation. Even for relatively short code sizes the score does not suffer much compared to using the full HOG features.*

a very small loss when using 512 bits (-0.5%) or 128 bits (-1%). Therefore, we can safely use binary encodings with 512 bits and enjoy the tremendous computational and memory advantages they bring.

3.3 Aggregating neighbour masks

As explained above, the key operation in our scheme is to transfer segmentations from the K most visually similar windows $\{s_1, s_2, \ldots, s_K\}$ in $\mathcal{S}$ to the target window w, where $s_1, \ldots, s_K$ are sorted from the most similar s_1 to the K-th most similar s_K. We then model the mask m_w for w as a weighted sum of the masks m_{s_k} of its neighbours:

$$m_w = \sum_{k=1}^{K} \lambda_k m_{s_k}, \tag{1}$$

where $\lambda_k \geq 0$, $\sum_k \lambda_k = 1$ and all the masks are normalized to the same size (50×50 in our experiments).

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. At the opposite end of the spectrum, $K = 1$ would only use the segmentation of the single nearest neighbour, making the transfer process sensitive to errors in individual source segmentations and reduce the ability to generalize from the source set. With uniform weights, it is therefore crucial to carefully set K.

To avoid having to manually set K, we propose instead to *learn* λ_k using training images from PASCAL VOC10 along with their ground-truth segmentations. For each training window w, we use the ground-truth segmentation of the full image to derive its ground-truth mask m_w. We train the weights λ_k by minimizing the sum of the Frobenius norms $||\cdot||_F$ of the residuals:

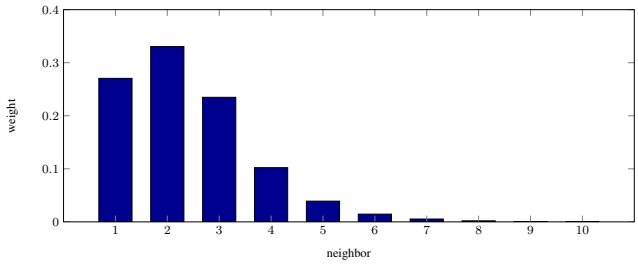

Fig. 5 *Weights learned on the pascal VOC10 dataset for the first 10 neighbours. The weights rapidly decrease and after neighbour 10 they are almost zero.*

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

We reparametrized this constrained convex quadratic program using

$$\lambda_k = \exp(\hat{\lambda}_k) / \sum_{k=1}^{K} \exp(\hat{\lambda}_k) \tag{3}$$

to obtain an unconstrained problem in $\{\hat{\lambda}_k\}$, which we then solved using Matlab's `fminunc` optimization function, based on an interior-point algorithm.

Observing the first 10 $\{\lambda_k\}$ in fig. 5, we see that the weights decrease rapidly. Learning the weights therefore serves two purposes. First, it improves the accuracy of segmentation transfer, over simply using uniform weights, as the residuals to ground-truth masks are minimized. Second, it allows to automatically determine the number K of neighbours needed to reach good accuracy. Since neighborus beyond rank 10 have near-zero weights, we set $K = 10$ in the rest of our experiments. As this K is small, the computation of segmentation transfer by eq. (1) is also sped up.

4 Models for image segmentation

Thanks to the technique of sec. 3, each image i of a class C in the target set $\mathcal{T}_t$ now has a transferred soft-segmentation mask M_i (fig. 2). This mask provides a rough initial indication of the position of the object. The next step is to refine it into a binary segmentation that delineates the object's spatial extent accurately. We model this task in an energy minimization framework analog to GrabCut [52, 37], where M_i is used to replace the user interaction, resulting in a fully automatic process.

In this section we describe how to segment each image independently, and explore extensions of the traditional

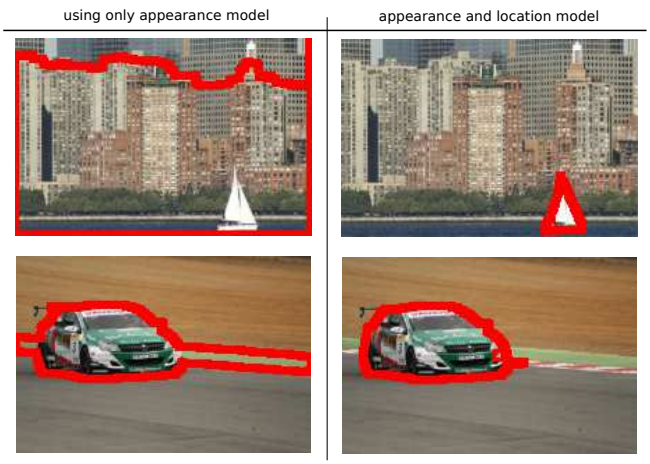

Fig. 6 *Our segmentation model uses the transferred mask M in two ways. The left column uses it only to initialize the appearance models. The right column uses it also as an additional location prior term. This improves the final segmentation considerably.*

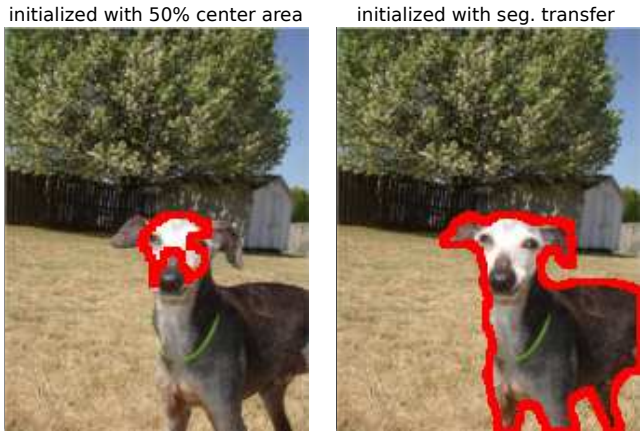

Fig. 7 *An example to illustrate the advantage of our segmentation transfer scheme. Left: segmentation produced by GrabCut when initialized by a rectangle in the image center. Right: GrabCut guided by our segmentation transfer scheme of sec. 4.1.*

GrabCut energy function (i) to incorporate the information given by M_i (sec. 4.1); (ii) to share labels among neighbouring pixels to improve computational and memory efficiency (sec. 4.2). In sec. 5, we further extend the framework to segment all the images in a class C *jointly*. This includes additional unary potentials for sharing appearance models between all images in C (sec. 5.1), and for importing appearance models from semantically related classes which have been segmented before in the propagation wave (sec. 5.3).

4.1 Iterative Graph-cuts guided by segmentation transfer

Let $x_{ip} \in \{0,1\}$ be the label and $c_{ip} \in [0,1]^3$ the color of pixel p in image i. Let $\mathbf{x}_i$ and $\mathbf{c}_i$ be the vectors of all x_{ip} and c_{ip}, respectively. The following energy function evaluates a binary foreground-background segmentation $\mathbf{x}_i$

$$E(\mathbf{x}_i; \mathbf{c}_i, M_i, A_i) = \sum_p E^A_{ip}(x_{ip}; c_{ip}, A_i) + \sum_p E^L_{ip}(x_{ip}; M_{ip}) + \sum_{(p,q)\in G} E_{ipq}(x_{ip}, x_{iq}) \quad (4)$$

This function is an extension of the traditional GrabCut energy [52]. It consists of two unary potentials E_{ip} for each pixel and a pairwise term E_{ipq} for each pair of neighbouring pixels in a 8-connected grid G. The pairwise potential is

$$E_{ipq}(x_{ip}, x_{iq}) = \delta(x_{ip} \neq x_{iq}) \cdot d(p,q)^{-1} \cdot \exp(-\gamma ||c_{ip} - c_{iq}||^2). \quad (5)$$

Analog to [52,10,70,59,12], this potential encourages smoothness by penalizing neighbouring pixels taking different labels. The penalty depends on the color contrast $||c_{ip} - c_{iq}||^2$ between the pixels, being smaller in regions of high contrast (image edges). It also depends on the distance $d(p,q)$ between the pixel positions in the image.

The first unary potential $E^A_{ip}(x_{ip}; c_{ip}, A_i)$ evaluates how likely a pixel of color c_{ip} is to take the label x_{ip} according to the image-specific color appearance model A_i. The model accounts for visual characteristics unique to an image. As in [52], the appearance model A_i 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 and each component is a full-covariance Gaussian over the RGB color space. We take the negative log-likelihood of the GMM as the potential

$$E^A_{ip}(x_{ip}; c_{ip}, A_i) = -\log p(x_{ip}; c_{ip}, A_i). \quad (6)$$

Many works using analog energy functions [53,7,10,72] require user interaction to estimate the appearance model, typically a manually drawn bounding-box or scribbles. In our work instead, the appearance models are *automatically* estimated from the transferred mask M_i. This is done by thresholding M_i to obtain an initial binary segmentation, from which foreground and background models are estimated.

Our energy function (4) also contains an additional unary term, which plays the role of a location prior preferring segmentations close to M_i. Because of the probabilistic nature of M_{ip}, we can directly use the negative log-likelihood of the corresponding Bernoulli distribution

$$E^L_{ip}(x_{ip}; M_{ip}) = -x_{ip} \log M_{ip} - (1 - x_{ip}) \log(1 - M_{ip}) \quad (7)$$

as a unary potential (where M_{ip} is the value of M_i at pixel p). This second term encourages a foreground segmentation at regions where M_i has high probability mass, which are quite reliably on the object of interest (fig. 2). This has a

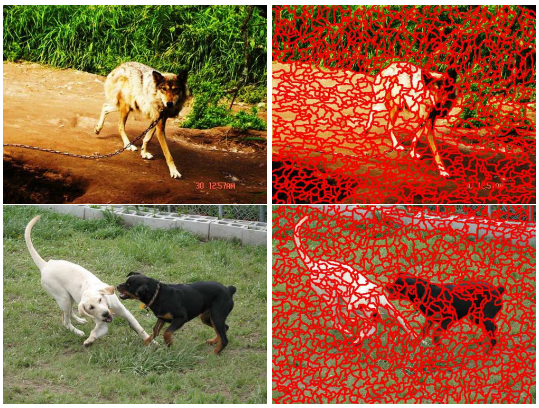

Fig. 8 Example images from ImageNet with their superpixel segmentations

complementary effect to using M_i to estimate the appearance models. Even with good appearance models, the segmentation could be attracted to similarly colored regions elsewhere in the image (fig 6). As a combined effect of using M in these two ways, the energy minimization becomes a controlled refinement operation, where the appearance models are used to outline the contours of the object in detail, but at the same time the segmentation is anchored approximately at the position indicated by M. Hence, our model fully exploits the information derived from segmentation transfer (sec. 3).

Now that the model is fully defined, we obtain a binary segmentation by minimizing (4) over all possible $\mathbf{x}_i$. Following [53], we now use this segmentation to update the appearance models, and then iteratively alternate between these two steps: finding the optimal segmentation $\mathbf{x}$ given the appearance models, and updating the appearance models given the segmentation. The first step is solved globally optimally using graph-cuts as our pairwise potentials are submodular. The second step fits GMMs to labeled pixels using the EM algorithm. Note that the segmentation transfer soft-mask M_i remains fixed during the entire procedure. Figure 7 illustrates the potential benefit of using our segmentation transfer scheme to guide GrabCut, compared to a baseline which initializes its appearance models from a rectangle in the center of the image.

4.2 Label sharing with superpixels

The method described in the previous section has the shortcomings of requiring us to store the full RGB image in memory and to construct a large graph-cut problem where every pixel is a variable. Reducing the size of the problem becomes interesting in our large-scale setting, as we consider the co-segmentation of thousands of images at the same time (section 5). A simple and widely used technique is to group pixels into *superpixels* [67,39], and assume that all the pixels inside a superpixel share the same label. This results in a simplified energy function with only one unary term per superpixel and with pairwise terms only between neighbouring superpixels. We use the superpixel method of Felzenszwalb and Huttenlocher [26], which is readily available online, with parameters $k = 10, \sigma = 0.5$ and a minimum of 50 pixels in each superpixel.

Let x_{is} be the label of superpixel s in image i, and $\mathbf{c}_{is}$ be the vector of pixel colors in s. We denote with M_{is} the transferred soft-mask for the image region covered by s. For simplicity, we overload the notation $\mathbf{x}_i$ to denote the vector of all superpixel labels in image i. With these definitions, we can now rewrite the energy (4) as

$$E(\mathbf{x}_i; \mathbf{c}_i, M_i, A_i) = \sum_s E^A_{is}(x_{is}; \mathbf{c}_{is}, A_i) + \sum_s E^L_{is}(x_{is}; M_{is}) + \sum_{s,t} E_{ist}(x_{is}, x_{it}) \quad (8)$$

The potentials are simply the sum of their counterparts in (4) over the pixels p inside a superpixel

$$E^{\cdot}_{is}(x_{is}; \cdot) = \sum_{p \in s} E^{\cdot}_{ip}(x_{is}; \cdot) \quad (9)$$

$$E_{ist}(x_{is}, x_{it}) = \sum_{p \in s,\ q \in t,\ (p,q) \in G} E_{ipq}(x_{is}, x_{it}) \quad (10)$$

This new energy indeed has a reduced set of variables, substantially speeding-up its minimization. Moreover, as we assume that all pixels in a superpixel share the same label, the corresponding pairwise terms vanish ($E_{ipq}(l, l)$ is 0 for any label l, see eq. (5)). This greatly reduces the number of pixel comparisons required to evaluate the pairwise terms. As a matter of fact, eq. (10) only sums over neighbouring pixels *along the boundary between neighbouring superpixels*.

However, there are no real memory benefits so far as we still need to evaluate the appearance likelihoods at each pixel, and the GMM apparance models themselves are still estimated using pixel values. In order to greatly reduce memory consumption and also speed-up the estimation of the appearance GMMs, we derive an accelerated EM algorithm below. This technique assumes that all the pixels inside a superpixel s not only share the same label but also the same responsibility z_{sk} towards the components k of the GMMs. This assumption is reasonable here, as a superpixel contains pixels of similar color, by construction [26] (fig. 8). This makes it likely for those pixels to have similar responsibilities.

The key idea is to retain only the sufficient statistics of the color distribution within each superpixel s, i.e. the number of pixels n_s, the color mean μ_s and covariance Σ_s. With this information, and similar to [68] for accelerated

EM clustering, we can derive an accelerated EM algorithm to estimate the parameters (n_k, μ_k, Σ_k) and mixture weight π_k of the GMMs. Below, we use $\mathcal{N}(x|\mu, \Sigma)$ to denote the probability of x under the Gaussian distribution centered at μ and with covariance Σ.

E-step: Update the responsibilities z_{sk} using the current parameters of the GMM.

$$\Sigma_{sk}^{-1} = \Sigma_k^{-1} + \Sigma_s^{-1} \tag{11}$$

$$\rho_{sk} = \mathcal{N}(\mu_s|\mu_k, \Sigma_{sk}) \tag{12}$$

$$z_{sk} = \frac{\pi_k \rho_{sk}}{\sum_l \pi_l \rho_{sl}}. \tag{13}$$

M-step: Re-estimate the parameters and mixture weight of each component under fixed responsibilities z_{sk}.

$$n_k = \sum_s n_s z_{sk} \tag{14}$$

$$\mu_k = \frac{1}{n_k} \sum_s n_s z_{sk} \mu_s \tag{15}$$

$$\Sigma_k = \frac{1}{n} \sum_s n_s z_{sk} \left(\Sigma_s + (\mu_s - \mu_k)(\mu_s - \mu_k)^\top\right) \tag{16}$$

$$\pi_k = \frac{n_k}{n}. \tag{17}$$

After estimating the appearance models, we can use an analog trick to also accelerate the computation of the appearance likelihood for all pixels in a superpixel

$$\begin{aligned} E_{is}(x_{is}; \mathbf{c}_i, A_i) &\approx E_{is}(x_{is}; n_s, \mu_s, \Sigma_s, A_i) \\ &\approx -n_s \log \left(\sum_k \pi_k \rho_{sk} \right). \end{aligned} \tag{18}$$

Hence, in order to apply GrabCut on our superpixel model, we only need to store the second-order statistics of each superpixel. This amounts to 13 values per superpixel (one for the number of pixels n_s, 3 for the color mean μ_s, and 9 for the 3×3 color covariance matrix Σ_s), compared to 3 per pixel in a standard model. In a typical 500×300 image, the algorithm [26] produces between 100 and 1000 superpixels, in about 0.1 seconds. This leads to memory savings in the order of $30\times$ to $300\times$, at negligible computational overhead. Moreover, our experiments (sec. 6.1) show that the accuracy of this approximate model is very close to the original one described in sec. 4.1.

5 Joint segmentation of a set of images

This section describes how to jointly segment all the images in a class C. Section 5.1 explains the general joint segmentation scheme, which extends the single-image model (8) with an additional unary potential carrying a class-wide appearance model. This scheme is adapted to each stage of the segmentation propagation to fit the situation (sec. 1.1). Stages 1 and 2 operate on classes for which some images with bounding-box annotations are available, so they can help constraining the segmentation (sec. 5.2). Later stages can import appearance models from semantically related classes that have been segmented in previous stages (sec. 5.3). This gives rise to further additional unary potentials. In sec. 5.4, we explain how to learn the optimal weights of all potentials so as to maximize segmentation accuracy on a validation set using structured-output SVMs [65].

5.1 Sharing appearance within a class

Given the set $\mathcal{I}$ of all images in a class C of ImageNet, let $\mathbf{x}$ be the vector of all pixel labels x_{ip} in all images. The energy function for jointly segmenting all images in $\mathcal{I}$ using the current source pool $\mathcal{S}$ is

$$\begin{aligned} E(\mathbf{x}; \mathcal{A}, \mathcal{S}) = \sum_i \Bigg(&\sum_p E_{ip}(x_{ip}; \mathcal{A}, \mathcal{S}) \\ &+ \sum_{(p,q) \in G_i} E_{ipq}(x_{ip}, x_{iq}) \Bigg) \end{aligned} \tag{19}$$

The pairwise potential remains unchanged from eq. (5), but the unary potential is now a linear combination of several terms

$$\begin{aligned} E_{ip}(x_{ip}; \mathcal{A}, \mathcal{S}) = &- \alpha_I \log p(x_{ip}; c_{ip}, A_i) \\ &- \alpha_C \log p(x_{ip}; c_{ip}, A_C) \\ &- \alpha_M \log M_{ip}(x_{ip}; \mathcal{S}) \end{aligned} \tag{20}$$

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 (as in sec. 4.1) and one class model A_C common to all images in $\mathcal{I}$. This class model enables us 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 color of a particular cow), while the class model accounts for classwide characteristics (e.g. the color of common cow backgrounds, such as grass and sky). All appearance models, *i.e.* $\{A_i\}_i$ and A_C, are GMMs with 5 full-covariance components for foreground and background. A_i are learnt separately on their respective images, whereas A_C is learnt on the union of all images. Finally, the last unary term is the image-specific location prior formed by the transferred soft-mask M_i (as in sec. 4.1). Figure 9 illustrates the various unary potentials.

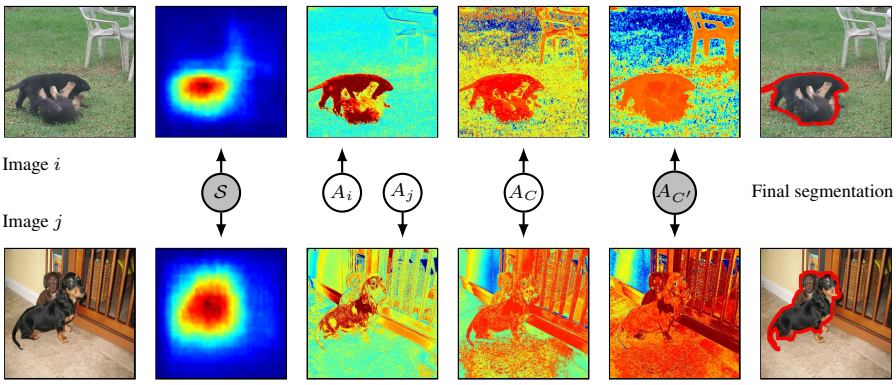

Fig. 9 *Our joint segmentation model. Left: two images i and j of a class to segment. The location priors M_i and M_j are obtained by segmentation transfer from $\mathcal{S}$ (second column). Image models A_i and A_j contribute to an image-specific unary potential (third column). The fourth column shows 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 potentials 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.*

This joint segmentation model can be seen as a generalization of both GrabCut [53] and Batra et al. [7]. 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. [7] uses only a single model shared among all images: $\mathcal{A} = \{A_C\}$.

The model (19) is used to segment the images $\mathcal{I}$ with the usual iterative optimization scheme which alternates between finding the optimal segmentation given the appearance models, and updating the appearance models given the segmentation. Each image model A_i is fitted to the current segmentation of its respective image i, 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. Image models complement the class model with extra GMM components that finely adapt to the specificities of each image.

As for the single-image model (sec. 4.2), we can also derive an accelerated joint class-level model (19) using superpixels. Like the image models, the class appearance model A_C is also learnt from the sufficient statistics of the color distributions in the superpixels, using the same accelerated EM algorithm, except we use the union of all foreground (resp. background) superpixels over all images.

5.2 Stages 1 and 2: Exploiting bounding-box annotations

As mentioned in sec. 1.1, stage $t = 1$ consists of segmenting images annotated with ground-truth bounding-boxes, as they are easier to segment. Those images are jointly segmented as presented in sec. 5.1 while constraining the minimization of (19) to the available ground-truth bounding-boxes (some images have multiple bounding-box annotations). This is done by imposing an infinite unary cost for foreground for all pixels outside any bounding-box.

At stage 2, when segmenting unannotated images in the same classes as stage 1, we include the images of stage 1 in (19) 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.

5.3 Later Stages: 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. 1.1). As these related classes have already been 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 [63], localization [29] and detection [55], but we believe it is unexplored for segmentation.

More precisely, when segmenting a new class C, we add to (19) a unary potential for each of its related classes $C' \in \mathcal{R}(C)$, which carries its appearance model $A_{C'}$. Since the related classes C' are already segmented from stage $t-1$, their appearance models can be stored and used at stage t without any extra computational cost. We therefore extend the unary potentials in eq. (20) to

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

(21)

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. 5.4).

Note how in eq. (21) we restrict the source pool used for segmentation transfer to $\mathcal{R}(C)$, to make it maximally related to C (as discussed in sec. 1.1).

5.4 Learning the weights α

Many of the models described above combine multiple unary potentials in a weighted sum. We learn the weights $\boldsymbol{\alpha}$ of the unary potentials on a small subset of 90 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 weight vector $\boldsymbol{\alpha} = \{\alpha_I, \alpha_C, \alpha_M, \alpha_R\}$ common to all later stages.

Let $\mathbf{x}_i$ be the labeling of all pixels in image i. Given n training images $\mathcal{I}$ with associated ground-truth labelings $\mathbf{x}^* = (\mathbf{x}_1^*, \ldots, \mathbf{x}_n^*)$, we seek the weights $\boldsymbol{\alpha}$ such 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

$$E(\mathbf{x}_i^*|i,\boldsymbol{\alpha}) \leq E(\mathbf{x}_i|i,\boldsymbol{\alpha}), \quad \forall \mathbf{x}_i \neq \mathbf{x}_i^*, \quad \forall i \in \mathcal{I}. \tag{22}$$

where $E(\mathbf{x}|i,\boldsymbol{\alpha})$ is one term in the outermost summation of eq. (19), corresponding to only one image. For simplicity, we omit $\mathcal{A}$ and $\mathcal{S}$ as they are predetermined by the segmentation transfer process and cannot change during the minimization of (19). To learn the parameters $\boldsymbol{\alpha}$ we solve a structured-output SVM training problem, following [65]

$$\begin{aligned} &\min_{\boldsymbol{\alpha},\xi} \frac{1}{2}\|\boldsymbol{\alpha}\|^2 + C\sum_{j=1}^{n}\xi_i \\ \text{s.t.} \quad &\forall \mathbf{x}_i \neq \mathbf{x}_i^*, \\ &\quad E(\mathbf{x}_i|i,\boldsymbol{\alpha}) - E(\mathbf{x}_i^*|i,\boldsymbol{\alpha}) \geq \Delta_i(\mathbf{x}_i^*,\mathbf{x}_i) - \xi_i, \\ &\forall i \in \mathcal{I}, \quad \xi_i \geq 0. \end{aligned} \tag{23}$$

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 and background pixels in the image

$$\Delta_i(\mathbf{x}_i^*, \mathbf{x}_i) = \sum_{p \in i} w_{ip}|x_{ip} - x_{ip}^*|, \tag{24}$$

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 and background pixels in i, respectively. 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. Note how this is a good proxy to the intersection-over-union (IoU) performance measure, on which we base much of our experiments (see sec. 6 for a discussion). However, IoU cannot be expressed exactly as a sum over unary potentials.

As each labeling $\mathbf{x}_i$ corresponds to a constraint, the number of constraints is exponential in the number of pixels. Constraint generation [65] circumvents this issue by iteratively solving (23) 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 [61].

In the case of models based on superpixels (sec. 4.2), we only need to modify Δ_i to reflect the misclassification of pixels using the shared label x_{is}:

$$\Delta_i(\mathbf{x}_i^*, \mathbf{x}_i) = \sum_{s \in i}\sum_{p \in s} w_{ip}|x_{is} - x_{ip}^*|. \tag{25}$$

To maximize performance, we learn separate sets of weights for early and later stages of the segmentation propagation, as the characteristics of the source pool and the role of the terms might change over stages.

6 Experiments

We validate the components of our approach on the recent iCoseg dataset [7] in sec. 6.1, and then present results on ImageNet in sec. 6.2. We conclude in sec. 7.

6.1 Cosegmentation on iCoseg

The iCoseg dataset [7] contains 643 images grouped into 38 classes (e.g. stonehenge, brown bear, gymnasts, airplanes). The task, as set out by previous works [7,34,71,45] is 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*). Additionally, we also report performance as the area of intersection between the foreground in the output segmentation and the foreground in the ground-truth segmentation, divided by the area of their union(*IoU* [1]).

In tab. 2 we compare several stripped down versions of our model (sec. 5.1). The first three use no segmentation transfer (sec. 3) and initialize their appearance models

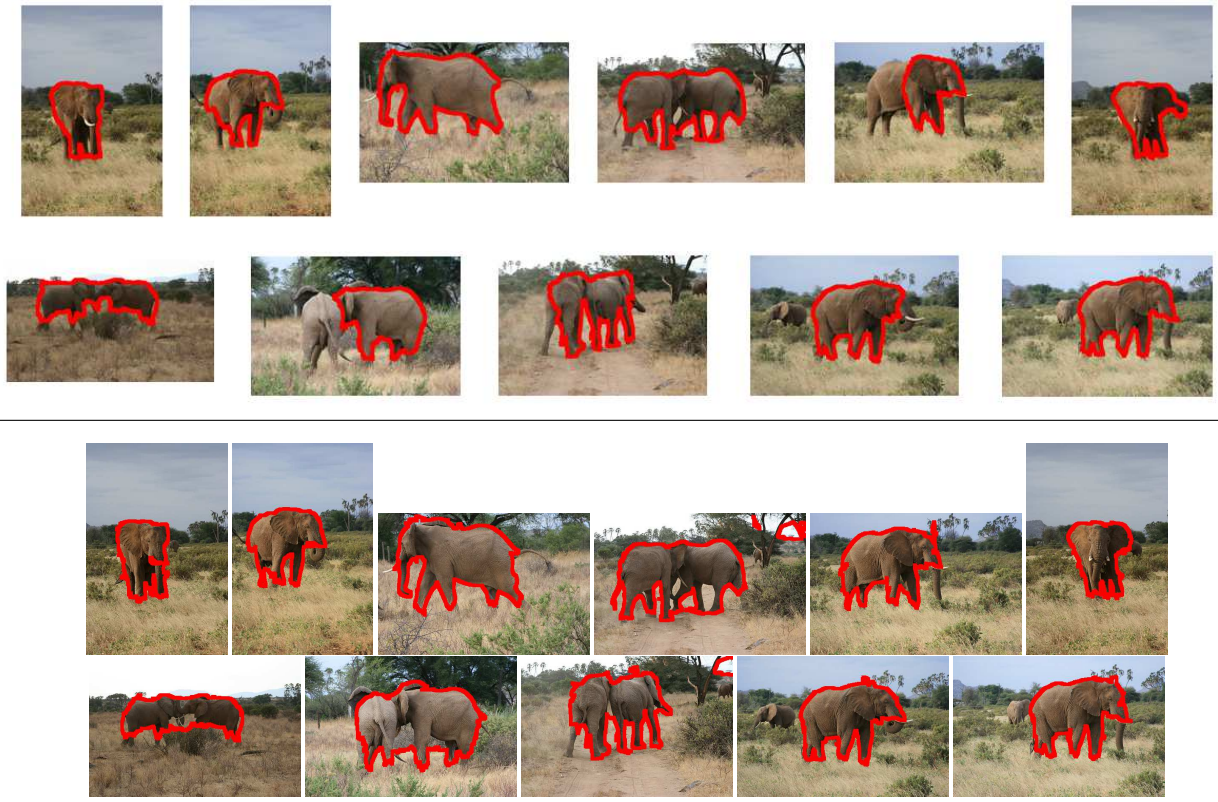

Fig. 10 *Top: Segmentations produced by our image+transfer+class method at the pixel-level on the* Elephants *class of the iCoseg dataset. Bottom: the same images segmented using complete superpixels models.*

from a window centered on the image. **(1) image only**: using only the image-specific unary potential A_i. This is essentially GrabCut [52], but with the user initialization replaced by a window in the image center; **(2) class only**: using only the class-wide unary potential A_C. This is very similar to [7], but again without user initialization; **(3) image+class**: using both types of unaries; **(4) image+transfer**: using the image-specific unary A_i and segmentation transfer (sec. 3) to initialize the appearance models and to add a location prior unary potential M_i (sec. 4). The source pool is fixed to the PASCAL VOC10 training set. This is a computationally efficient version of [37] using the speedups we proposed in sec. 3.2. As reported in [37], 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. Note that here we cannot evaluate the idea of recursively updating the source pool (sec. 1.1) nor of importing appearance models from related classes (sec. 5.3), as classes in iCoseg are not organized in a hierarchy.

The size of the initialization window for models (1-3) is set to 25% of the image area, which worked best on this dataset. For the models using multiple unary potentials (3-5), we use the technique in sec. 5.4 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.

For each method, in addition to the pixel-level models (**pixel model**), we also report the accuracy obtained when sharing the labels of pixels in each superpixel as described in sec. 4.2. The results are obtained without (**label sharing**) or with the accelerated EM algorithm (**complete superpixel model**).

As the first row of table 2 shows, the baseline pixel-level GrabCut model already shows a good performance (82.4% accuracy). Using class-wide appearance 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. 10). Segmentation transfer [37] 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 we observed in [37] on other datasets (PASCAL VOC10, Graz-02, Weizmann horses).

The second row of table 2 shows that sharing labels within superpixels has an impact on performance around a few tenth of percent, and is largest for the full model (+0.8% on image+transfer+class). The same conclusions as for pixel-level models remain valid: adding class-wide appearance models improves over image-specific models (+5.5% alone, +4.2% combined with transfer), and segmentation transfer provides substantial benefits (+5.5% on image-only, +4.2% on image+class). Interestingly, the approximate speeded-up EM algorithm to estimate the GMM for superpixel-level models obtains very similar performance as well (third row of table 2). This implies the approximation described in sec. 4.2 is reasonable and that the underlying assumptions (notably, that the pixels inside a superpixel share the same responsibility) are valid. The approximation is tighter when using more powerful models ($+0.4\%$ on the full model, -0.1% on image+class vs. -0.8% on image-only). This is expected in particular for class models, as their GMMs are estimated from many more superpixels, and therefore are less likely to overfit to the statistics of a few superpixels. Importantly, these models defined completely on superpixels are much faster to run ($\approx 15\times$ faster) and use orders of magnitude less memory ($\approx 100\times$) than the pixel-level ones. Therefore, they are a good choice for large-scale image co-segmentation (sec. 6.2). Interestingly, these computational savings come at no loss of performance for the full model, which in fact improves by a small amount (+1.2%).

Table 2 also reports the average accuracy of two recent state-of-the-art works [71,34] as reported in [71]. In a comparable setting using only iCoseg images, our image+class method outperforms them both (image+class). Our image+transfer+class method performs best by a considerable margin. While it uses manually segmented PASCAL VOC10 images as training data, we stress that these contain different classes than iCoSeg (e.g. there are no elephants in PASCAL VOC10). Importantly, our method is also computationally much more efficient than [71,34]. It takes only about 60 seconds to segment a typical iCoSeg class containing 20 images, including all processing stages. This is in contrast to several hours per class reported by [71,34]. Hence, we can apply our technique to the much larger ImageNet dataset. As our method is roughly linear in the number of images in a class, we report here a breakdown of the computational cost of each stage *per image*: 2s for extracting objectness windows, 0.1s for the HOG features, 0.2s for segmentation transfer (sec. 3.2), 0.1s for extracting superpixels [26], 0.5s to setup the segmentation model (i.e. computing the color models, unary and pairwise potentials) 0.1s for energy minimization (see table 1). As an additional remark, our best performance of 92.6% accuracy is also similar to the one reported in the recent work of [45] (92.5%). However, their average is computed over only 14 of the 38 classes, which makes this comparison only indicative.

extract objectness windows	2.0s
HOG features	0.1s
segmentation transfer	0.2s
superpixels	0.1s
segmentation model setup	0.5s
energy minimization	0.1s
total per image	3.0s

Table 1 Breakdown of the computational costs per image.

We also computed the performance of the different components of our method using the intersection-over-union measure [1]. This is a much more challenging and realistic measure of performance. It is considered superior to the simpler percentage of correctly labeled pixels [1], as it is automatically normalized to the scale of the foreground object and properly penalizes segmentations which miss the object. An empty segmentation scores 0 on IoU, but it might still score high in per-pixel accuracy (especially for small objects, fig. 11). Therefore, we expect that what were small differences in accuracy in 2 can correspond to larger differences in IoU. This is particularly true beyond 85% accuracy. Equivalently, this corresponds to the idea that IoU decreases much more rapidly than accuracy as the number of incorrect foreground pixels increases. Table 3 reports the results. The conclusions are similar to what observed under the accuracy measure. Class models now perform considerably better than image models. Adding either class models or segmentation transfer is always beneficial (e.g. $+7.2\%$ by adding segmentation transfer to image models). Combining image model, class models and segmentation transfer leads to the best results, which are substantially better than the basic image only GrabCut ($+15\%$). Analog to what observed for the accuracy measure, using superpixel-level models has only a minor impact on segmentation performance. The performance of our full method (image+transfer+class) increases by 1.2% compared to the pixel-level models, yielding a final IoU of 73.2%.

6.2 Segmentation propagation on ImageNet

We have run our full segmentation propagation method on two subtrees (*animal* and *instruments*) 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. For those classes with bounding-boxes, only a fraction (typically about 25%) of the images indeed have a bounding-box annotation. 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.

Accuracy	[34]	[71]	image only ≈GrabCut [52]	class only ≈Batra [7]	image+class	image+transfer ≈Kuettel [37]	image+transfer +class
Pixel model	78.9	85.4	82.4	83.6	88.2	87.6	**91.4**
Label sharing	-	-	82.5	83.4	88.0	88.0	**92.2**
Complete superpixel model	-	-	81.3	82.3	87.9	87.8	**92.6**

Table 2 *Segmentation accuracy on iCoseg. The results for [71, 34] are taken from table 1 in [71]. Columns 3-6 are stripped down versions of our model. The last column is our complete model (see main text).*

IoU	image only ≈GrabCut [52]	class only ≈Batra [7]	image+class	image+transfer ≈Kuettel [37]	image+transfer +class
Pixel model	57.3	61.7	62.6	64.5	**72.0**
Label sharing	57.8	61.4	62.6	64.8	**72.7**
Complete superpixel model	55.7	60.2	62.5	64.4	**73.2**

Table 3 *Performance of segmentation models on iCoseg as measured by* IoU. *Columns 1-4 are stripped down versions of our model. The last column is our complete model (see main text).*

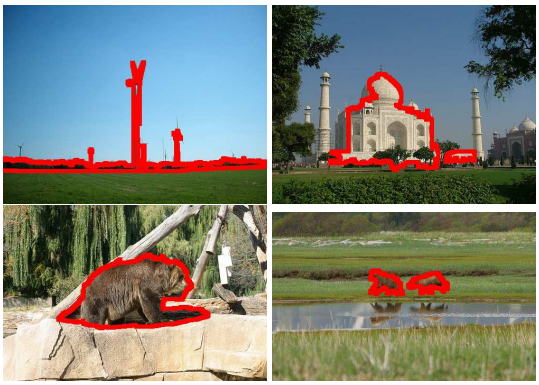

Fig. 11 From the top left to the bottom right image, the intersection over union performance (21%, 60%, 82%, 91%, resp.) better represents the quality of the segmentation than accuracy (95%, 94%, 97%, 100%, resp.). This is particularly true for small objects.

To quantitatively evaluate our approach, we obtained segmentation annotations via Amazon Mechanical Turk for 10 random images from 446 classes, for a total of 4460 images. We requested multiple human annotators per image. Our preliminary results in [38] were based directly on these segmentations, which included a small amount of noise. After a manual clean up, 184 images that had no good quality annotations were discarded.[3] We released the remaining 6225 segmentations for 4276 images of 445 classes online at `http://www.vision.ee.ethz.ch/~mguillau/imagenet.html?calvin`. These annotations enable us to reliably estimate the segmentation performance of our method on a wide range of classes. Additionally, we have held out a small set of 90 images to estimate α, as discussed in sec. 5.4.

In the remainder of this section, we evaluate the different components of our model, and provide in-depth analysis of the resulting segmentations.

[3] Therefore, the numbers reported in [38] are not directly comparable with the ones in this article.

Spatial support for transfer. An important element of our approach is on which spatial support to perform segmentation transfer (sec. 3). Hence, we evaluate the quality of the transferred masks while varying the kind of spatial support: (1) **Full image**: transfer masks based on global similarity at the image level. The mask of the target image is the weighted sum of the masks of its 10 nearest neighbours (sec. 3.3). (2) **Random windows**: use 100 uniformly sampled windows in the source and target images to perform window-level mask transfer. (3) **Objectness windows**: use 100 objectness windows, sampled following [3]. This corresponds to our method in sec. 3.

We present our quantitative evaluation of the transfer masks in table 4 for accuracy and in table 5 for IoU. We assess the quality of the transfer masks in two ways: (1) directly, by thresholding them at 0.5 (i.e. keep as foreground all pixels with probability > 0.5, column 'Thresholded mask'); (2) refine the segmentation by using the transfer mask to guide our GrabCut-like segmentation model described in sec. 4.1 (column 'Mask + GrabCut'). We observe that using local support, either random windows or objectness windows, is always beneficial over the use of global support (full image). Indeed, using random windows outperforms using the full image both when thresholding the mask and with GrabCut. The difference is as big as +11% (with GrabCut and under IoU). Interestingly, objectness windows bring a further improvement over random windows in all settings, with as much as +13% IoU with GrabCut. This confirms the observations in [37] that transferring masks benefits from object-centered spatial support. GrabCut improves segmentation over thresholding for all spatial supports and performance measures (+10% IoU for our method using objectness windows).

Segmentation models. We compare here the segmentation performance of various baselines and variants of our system on ImageNet: (a) **GrabCut [52] image center**: run in-

	Thresholded mask	Mask + GrabCut
Full image	72.7	75.8
Random windows	72.1	78.5
Objectness windows	79.2	82.5

Table 4 Mean accuracy for different spatial supports for segmentation transfer on ImageNet.

	Thresholded mask	Mask + GrabCut
Full image	21.4	29.0
Random windows	23.9	40.0
Objectness windows	42.1	52.0

Table 5 Mean IoU for different spatial supports for segmentation transfer on ImageNet..

	Accuracy	IoU
GrabCut image center	73.4	24.0
Pixel image+transfer [37]	82.5	52.0
Superpixel image+transfer	82.2	52.7
Superpixel image+propagation	84.1	57.0
Superpixel image+propagation+class	84.4	57.3

Table 6 Mean accuracy and mean IoU on ImageNet.

dividually on every image, using a centered window for initialization (as in "image only" in iCoseg experiments); (b) **Pixel image+transfer**: use the objectness transfer mask to guide GrabCut. This is the best performing method from the previous paragraph, and corresponds to our segmentation transfer scheme [37] (with the modifications detailed in sec. 3), using VOC10 as a fixed source pool; (c) **Superpixel image+transfer**: the superpixel version of (b), using the accelerated models of sec. 4.2; (d) **Superpixel image+propagation**: now including our propagation scheme, where the segmentations output by a stage are added to the source pool of the next (sec. 1.1); this also uses ground-truth bounding-boxes at stage 1; (e) **Superpixel image+propagation+class**: now including also class appearance models (sec. 5.1) and importing appearance models from related classes segmented in previous stages (sec. 5.3)). This is our full pipeline.

As reported in table 6, the performance of GrabCut initialized from the image center is 73.4% accuracy and 24.0% IoU. When using VOC10 as a fixed source pool for segmentation transfer to guide GrabCut, the performance greatly increases to 82.5% accuracy / 52.0% IoU. Accelerated superpixel models obtain very similar performance: -0.3% accuracy, +0.7% IoU. This confirms that these faster models come at no significant loss in performance. Propagating the segmentation masks between stages further increases performance to 84.1% accuracy and 57.0% IoU. It is interesting to analyse the effects of propagation on each stage individually (table 7). In stage 1, the large performance improvement between the first two rows is due to using the segmentation process to guide the ground-truth bounding-box. 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 segmentation propagation idea, compared to segmentation transfer from the fixed $\mathcal{S}_0$ pool. As the table shows, propagation improves IoU for *all levels* by about 1.8%.[4] This demonstrates the value of recursively employing images segmented before to help segmenting new images. Finally, we note how sharing appearance models between images of a class and importing models from related classes brings only a minor additional benefit of +0.3% IoU on average (third row of table 7).

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. Comparing the performance of our full method on iCoseg (92.6% accuracy and 73.2% IoU) and ImageNet 84.4% / 57.3%, we see that iCoseg is an easier dataset, and ImageNet provides a much more challenging setting.

Propagation statistics. A central element of our propagation scheme is that the source set for mask transfer continuously grows over stages (sec. 1.1). This is in contrast to using only PASCAL VOC10 as a fixed source set throughout. If the propagation idea works, then the fraction of retrieved window neighbours that come from ImageNet itself should gradually increase as the stages progress (sec. 3.1). The fraction of VOC10 neighbours should instead decrease. We investigate this phenomenon here.

Naturally, the first level uses 100% VOC10 neighbours, as there no ImageNet images have been segmented yet. At stage 2, 58% of the neighbours are from VOC10, the others propagate from level 1. Interestingly, at stage 2, the VOC10 data still makes up for 81% of the source set. The fraction of neighbours that are actually from VOC10 is below this expected value, which demonstrates that the propagation is already happening at this stage. Moreover, as propagation unfolds over levels 3 to 5, the fraction of neighbours that come from VOC10 shrinks further. At level 3, 26% VOC10 neighbours are used, whereas the source set is composed

[4] This differs from the conclusion we reached in our earlier paper [38]. The output segmentations were affected by a bug in our GrabCut implementation, resulting in many erroneous segmentations. These errors were amplified through propagation, leading to the observation that performance decreased with stages. On average over all images, in [38], we reported 77.1% accuracy. When evaluated using the refined ground-truth, those segmentations yield 80.0% accuracy and 37.3% IoU, clearly below the correct result we report in this paper (84.4% accuracy and 57.3% IoU).

IoU	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Overall
Superpixel image+transfer	50.6	50.3	54.4	54.6	60.9	52.7
Superpixel image+propagation	63.3	52.1	56.3	56.0	62.6	57.0
Superpixel image+propagation+class	63.6	52.7	56.6	56.2	62.4	57.3

Table 7 Breakdown of performance per stage, measured in IoU, for "Superpixel image+transfer", "Superpixel image+propagation", "Superpixel image+propagation+class".

at 39% by VOC10 data. At level 4, it is 14% versus 22%, and at level 5, 10% versus 17%. As the VOC10 neighbours are always below the proportion observed in the source set, we can conclude that our scheme to choose related classes for transfer in the propagation scheme is appropriate, as the source set of a stage truly contains windows that are more visually similar to the target images in that stage than the default source VOC10.

7 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 [7] for co-segmentation, where it delivers state-of-the-art results.

In future work, we plan plan to exploit 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 that we adapt the segmentation technique to each target class, and 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 propagating 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. 2d).

References

1. The pascal visual object classes challenge. http://pascallin.ecs.soton.ac.uk/challenges/VOC/
2. Alexe, B., Deselaers, T., Ferrari, V.: ClassCut for unsupervised class segmentation. In: Proceedings of the European Conference on Computer Vision (2010)
3. Alexe, B., Deselaers, T., Ferrari, V.: What is an object? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2010)
4. Arora, H., Loeff, N., Forsyth, D., Ahuja, N.: Unsupervised segmentation of objects using efficient learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis (2007)
5. Aytar, Y., Zisserman, A.: Tabula rasa: Model transfer for object category detection. In: Proceedings of the International Conference on Computer Vision (2011)
6. Aytar, Y., Zisserman, A.: Enhancing exemplar svms using part level transfer regularization. In: Proceedings of the British Machine Vision Conference (2012)
7. Batra, D., Kowdle, A., Parikh, D., Luo, J., Chen, T.: iCoseg: Interactive co-segmentation with intelligent scribble guidance. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3169–3176 (2010)
8. Batra, D., Kowdle, A., Parikh, D., Luo, J., Chen, T.: Interactively co-segmentating topically related images with intelligent scribble guidance. International Journal of Computer Vision (2011)
9. Bertelli, L., Yu, T., Vu, D., Gokturk, S.: Kernelized structural SVM learning for supervised object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2011)
10. Blake, A., Rother, C., Brown, M., Perez, P., Torr, P.: Interactive image segmentation using an adaptive GMMRF model. In: Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic (2004)
11. Borenstein, E., Sharon, E., Ullman, S.: Combining top-down and bottom-up segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC (2004)
12. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada (2001)
13. Carreira, J., Sminchisescu, C.: Constrained parametric min cuts for automatic object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2010)
14. Chai, Y., Lempitsky, V., Zisserman, A.: Bicos: A bi-level co-segmentation method for image classification. In: Proceedings of the International Conference on Computer Vision, pp. 2579–2586 (2011)
15. Chai, Y., Rahtu, E., Lempitsky, V., Gool, L.V., Zisserman, A.: Tricos: A tri-level class-discriminative co-segmentation method for image classification (2012)
16. Dalal, N., Triggs, B.: Histogram of Oriented Gradients for Human Detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 886–893 (2005)
17. Deng, J., Berg, A., Satheesh, S., Su, H., Khosla, A., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012). http://www.image-net.org/challenges/LSVRC/2012/
18. Deng, J., Berg, A.C., Li, K., Fei-Fei, L.: What does classifying more than 10,000 image categories tell us? In: Proceedings of the European Conference on Computer Vision (2010)
19. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-fei, L.: ImageNet: A large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2009)

20. Deng, J., Krause, J., Berg, A., Fei-Fei, L.: Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2012)
21. Deng, J., Satheesh, S., Berg, A., Fei-Fei, L.: Fast and balanced: Efficient label tree learning for large scale object recognition. In: Advances in Neural Information Processing Systems (2011)
22. Deselaers, T., Alexe, B., Ferrari, V.: Localizing objects while learning their appearance. In: Proceedings of the European Conference on Computer Vision (2010)
23. Deselaers, T., Ferrari, V.: Visual and semantic similarity in imagenet. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2011)
24. Endres, I., Hoiem, D.: Category independent object proposals. In: Proceedings of the European Conference on Computer Vision (2010)
25. 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)
26. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. International Journal of Computer Vision **59**(2), 167–181 (2004)
27. Gong, Y., Lazebnik, S.: Iterative quantization: A procrustean approach to learning binary codes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2011)
28. Guillaumin, M., Ferrari, V.: Large-scale knowledge transfer for object localization in imagenet. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2012)
29. Guillaumin, M., Ferrari, V.: Large-scale knowledge transfer for object localization in ImageNet. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2012)
30. Guillaumin, M., Mensink, T., Verbeek, J., Schmid, C.: TagProp: discriminative metric learning in nearest neighbor models for image auto-annotation. In: Proceedings of the International Conference on Computer Vision (2009)
31. Hays, J., Efros, A.: Scene completion using millions of photographs. In: Proceedings of the ACM SIGGRAPH Conference on Computer Graphics (2007)
32. Jiang, H.: Human pose estimation using consistent max-covering. In: Proceedings of the International Conference on Computer Vision (2009)
33. Jojic, N., Perina, A., Cristani, M., Murino, V., Frey, B.: Stel component analysis: Modeling spatial correlations in image class structure. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2009)
34. Joulin, A., Bach, F., Ponce, J.: Discriminative clustering for image co-segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1943–1950 (2010)
35. Kim, G., Xing, E., Fei-Fei, L., Kanade, T.: Distributed cosegmentation via submodular optimization on anisotropic diffusion. In: Proceedings of the International Conference on Computer Vision, pp. 169–176 (2011)
36. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (2012)
37. Kuettel, D., Ferrari, V.: Figure-ground segmentation by transferring window masks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2012)
38. Kuettel, D., Guillaumin, M., Ferrari, V.: Segmentation Propagation in ImageNet. In: Proceedings of the European Conference on Computer Vision (2012). URL `http://groups.inf.ed.ac.uk/calvin/Publications/kuettelECCV12.pdf`
39. Ladicky, L., Russell, C., Kohli, P.: Associative hierarchical crfs for object class image segmentation. In: Proceedings of the International Conference on Computer Vision (2009)
40. Lampert, C., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2009)
41. Li, f., Carreira, J., Sminchisescu, C.: Object recogntion as ranking holistic figure-ground hypotheses. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2010)
42. Lin, Y., Lv, F., Zhu, S., Yang, M., Cour, T., Yu, K., Cao L. Huang, C.: Large-scale image classification: fast feature extraction and SVM training. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2011)
43. Liu, C., Yuen, J., Torralba, A.: Nonparametric scene parsing: Label transfer via dense scene alignment. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2009)
44. Malisiewicz, T., Gupta, A., Efros, A.A.: Ensemble of exemplar-svms for object detection and beyond. In: Proceedings of the International Conference on Computer Vision (2011)
45. Mukherjee, L., Singh, V., Xu, J., Collins, M.D.: Analyzing the subspace structure of related images: Concurrent segmentation of image sets. In: Proceedings of the European Conference on Computer Vision (2012)
46. Norouzi, M., Punjani, A., Fleet, D.J.: Fast search in hamming space with multi-index hashing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2012)
47. Oliva, A., Torralba, A.: Modeling the shape of the scene: a holistic representation of the spatial envelope. International Journal of Computer Vision **42**(3), 145–175 (2001)
48. Ott, P., Everingham, M.: Shared parts for deformable part-based models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2011)
49. Quattoni, A., Collins, M., Darrell, T.: Transfer learning for image classification with sparse prototype representations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2008)
50. Rohrbach, M., Stark, M., Szarvas, G., Gurevych, I., Schiele, B.: What helps where and why? semantic relatedness for knowledge transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2010)
51. Rosenfeld, A., Weinshall, D.: Extracting foreground masks towards object recognition. In: Proceedings of the International Conference on Computer Vision (2011)
52. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. In: Proceedings of the ACM SIGGRAPH Conference on Computer Graphics (2004)
53. Rother, C., Kolmogorov, V., Minka, T., Blake, A.: Cosegmentation of image pairs by histogram matching - incorporating a global constraint into MRFs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2006)
54. Russel, B., Torralba, A., Liu, C., Fergus, R.: Object recognition by scene alignment. In: Advances in Neural Information Processing Systems (2007)
55. Salakhutdinov, R., Torralba, A., Tenenbaum, J.: Learning to share visual appearance for multiclass object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2011)
56. Van de Sande, K., J.R.R., U., Gevers, T., Smeulders, A.: Segmentation as selective search for object recognition. In: Proceedings of the International Conference on Computer Vision (2011)
57. Schoenemann, T., Cremers, D.: Introducing curvature into globally optimal image segmentation: Minimum ratio cycles on product graphs. In: Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, Brazil (2007)
58. Shotton, J., Blake, A., Cipolla, R.: Contour-Based Learning for Object Detection. In: Proceedings of the International Conference on Computer Vision (2005)

59. Shotton, J., Winn, J., Rother, C., Criminisi, A.: TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: Proceedings of the European Conference on Computer Vision (2006)
60. Stark, M., Goesele, M., Schiele, B.: A shape-based object class model for knowledge transfer. In: Proceedings of the International Conference on Computer Vision (2009)
61. Szummer, M., Kohli, P., Hoiem, D.: Learning CRFs using graph cuts. In: Proceedings of the European Conference on Computer Vision (2008)
62. Tighe, J., Lazebnik, S.: Superparsing: Scalable nonparametric image parsing with superpixels. In: Proceedings of the European Conference on Computer Vision (2010)
63. Tommasi, T., Orabona, F., Caputo, B.: Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2010)
64. Torralba, A., Fergus, R., Weiss, Y.: Small codes and large image databases for recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2008)
65. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research **6**, 1453–1484 (2005)
66. Tu, Z., Chen, X., Yuille, A., Zhu, S.: Image parsing: Unifying segmentation, detection, and recognition. International Journal of Computer Vision **63**(2), 113–140 (2005)
67. Veksler, O., Boykov, Y., Mehrani, P.: Superpixels and supervoxels in an energy optimization framework. In: Proceedings of the European Conference on Computer Vision, pp. 211–224 (2010)
68. Verbeek, J., Nunnink, J., Vlassis, N.: Accelerated EM-based clustering of large data sets. Data Mining and Knowledge Discovery **13**(3), 291–307 (2006)
69. Verbeek, J., Triggs, B.: Region classification with Markov field aspect models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2007)
70. Vicente, S., Kolmogorov, V., Rother, C.: Graph cut based image segmentation with connectivity priors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, Alaska (2008)
71. Vicente, S., Rother, C., Kolmogorov, V.: Object cosegmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2217–2224 (2011)
72. Wang, J., Cohen, M.: An iterative optimization approach for unified image segmentation and matting. In: Proceedings of the 10th International Conference on Computer Vision, Beijing, China (2005)
73. Winn, J., Jojic, N.: LOCUS: learning object classes with unsupervised segmentation. In: Proceedings of the International Conference on Computer Vision (2005)

Fig. 12 *Example segmentation output by our full segmentation propagation scheme. The rightmost column shows some failure cases.*

Fig. 13 *More example segmentation output by our full segmentation propagation scheme.*