Predicting Sufficient Annotation Strength for Interactive Foreground Segmentation

Suyog Dutt Jain Kristen Grauman
University of Texas at Austin
suyog@cs.utexas.edu, grauman@cs.utexas.edu

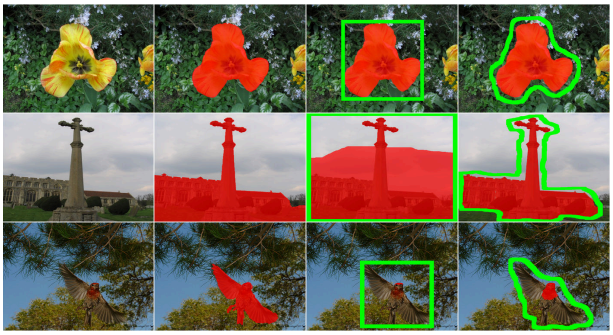
Abstract

The mode of manual annotation used in an interactive segmentation algorithm affects both its accuracy and ease-of-use. For example, bounding boxes are fast to supply, yet may be too coarse to get good results on difficult images; freehand outlines are slower to supply and more specific, yet they may be overkill for simple images. Whereas existing methods assume a fixed form of input no matter the image, we propose to predict the tradeoff between accuracy and effort. Our approach learns whether a graph cuts segmentation will succeed if initialized with a given annotation mode, based on the image's visual separability and foreground uncertainty. Using these predictions, we optimize the mode of input requested on new images a user wants segmented. Whether given a single image that should be segmented as quickly as possible, or a batch of images that must be segmented within a specified time budget, we show how to select the easiest modality that will be sufficiently strong to yield high quality segmentations. Extensive results with real users and three datasets demonstrate the impact.

1. Introduction

Foreground segmentation is a fundamental vision problem with an array of applications. Visual search systems need foreground segmentation to properly isolate a user's query. For example, suppose a mobile phone user snaps a photo of an armchair at his friend's home that he wants to purchase online; the search system needs to issue a query based on the chair's visual features, separate from the surrounding living room. Similarly, training an object recognition system often requires segmenting objects, so that they can be learned from natural scenes. Likewise, graphics applications demand rotoscoping to insert a segmented object into different backgrounds, or to reconstruct a 3D model of an object visible in multiple views. In any such scenario, it is natural for humans to help annotate the foreground.

Research on *interactive segmentation* considers how a human can work in concert with a segmentation algorithm to efficiently identify the foreground region [8, 16, 2, 19, 12, 7, 1]. The idea is to leverage the respective strengths of both the human and the algorithm. While the human understands the semantics of the scene and can easily identify the foreground, outlining pixel-level boundaries is painstaking. While the algorithm can easily assign pixels to objects based on their low-level properties, predicting what properties each object has remains elusive. Thus, the human gives high-level guidance—in the form of coarse spatial annotations—and the algorithm propagates that input down to the pixel level. Often this is done by constructing a foreground color model from the user-indicated regions, then optimizing foreground/background labels on each pixel (e.g., using graph cuts [2, 19]).



(a) Image (b) Ground Truth (c) Bounding Box (d) Sloppy Contour

Figure 1: Interactive segmentation results (shown in red) for three images using various annotation strengths (marked in green). Note how the most effective mode of input depends on the image content. Our method predicts the easiest input modality that will be sufficiently strong to successfully segment a given image.

Existing methods assume the user always gives input in a particular form (e.g., a bounding box or a scribble), and so they focus on how to use that input most effectively. However, simply fixing the input modality leads to a suboptimal tradeoff in human and machine effort. The problem is that each mode of input requires a different degree of annotator effort. The more elaborate inputs take more manual effort,

yet they leave less ambiguity to the system about which pixels are foreground. At the same time, depending on its content, an image may be better served by one form or another.

For example, Figure 1 shows (a) three images, (b) their ground truth foreground, and their interactive segmentation results (shown in red) using either (c) a bounding box or (d) a freehand outline as input (marked in green). The flower (top row) is very distinct from its background and has a compact shape; a bounding box on that image would provide a tight foreground prior, and hence a very accurate segmentation with very quick user input. In contrast, the cross image (middle row) has a plain background but a complex shape, making a bounding box insufficient as a prior; the more elaborate freehand "sloppy contour" is necessary to account for its intricate shape. Meanwhile, the bird (bottom row) looks similar to the background, causing both the bounding box and sloppy contour to fail. In that case, a manually drawn tight polygon may be the best solution.

The tradeoffs are clear, but what is a system to do about it? The system needs to determine what tool works best, but before the human uses the tool! A very experienced user—especially a vision researcher well versed in how the underlying algorithms work—might be able to predict which input tool will suffice, but to require such knowledge is to exclude many application areas where non-experts must be able to assist the system. Furthermore, ideally the system should segment the object well in one shot, as opposed to requiring back-and-forth with the user to correct its mistakes (e.g., with scribbles); this is especially true for visual search on a mobile device, where a user has a query image in hand and would like to quickly identify the foreground and ship it to a server.

We propose to learn the image properties that indicate how successful a given form of user input will be, once handed to an interactive segmentation algorithm. Our approach works as follows. First, we develop features capturing the degree of separability between foreground and background regions, as well as the uncertainty of a graph cuts-based optimal label assignment. Then, we use these features on images for which the true foreground is known to train discriminative models that predict whether an image will be "easy" or "difficult" for each input modality. Given a novel image, we apply a saliency detector to coarsely estimate the foreground. Using that estimate, we extract the separability features, and apply the difficulty classifiers.

Having predicted the relative success of each modality, we can explicitly reason about the tradeoff in user effort and segmentation quality. We propose two ways to determine the appropriate annotation choice. In the first, we take a single image as input, and ask the human user to provide *the easiest (fastest) form of input that the system expects to be sufficiently strong* to do the job. In the second, we take a batch of images as input together with a budget of time that the user is willing to spend guiding the system. We show how to optimize the *mix* of input types that will maximize total segmentation accuracy, subject to the budget.

We validate our approach on three datasets. We demonstrate scenarios where the system must segment individual unrelated snapshots (relevant for search applications) as well as co-segment collections of related images (relevant for recognition and 3D reconstruction applications). We show the proposed difficulty predictions outperform color variance metrics as well as a state-of-the-art technique to predict manual effort [22]. In real user studies with 101 users, our method not only meets the budget, but it does so while producing more accurately segmented results. Overall, the results clearly establish the value in reasoning about sufficient annotation strength in interactive segmentation.

2. Related Work

Early interactive segmentation methods include active contours [8] and intelligent scissors [16], where a user draws loose contours that the system snaps to a nearby object. Alternatively, a user can indicate some foreground pixels—often with a bounding box or mouse scribble—and then use graph cuts to optimize pixel label assignments based on a foreground likelihood and local smoothness prior [2, 19]. Building on this idea, recent work develops co-segmentation [1], topological priors [12], shape constraints [7], and simulated human user models [10]. In all prior methods, the user's annotation tool is fixed. We show how to tailor the user's input modality to achieve best graph cut segmentation results with minimal effort.

Active learning helps minimize the amount of labeled examples needed to train a recognition system, and can be used to solicit region labels [22, 20, 24, 21]. In contrast to our work, the goal of active learning is to build a reliable classifier, and examples are sequentially selected for labeling based on how they reduce category uncertainty. Our method is class-independent and addresses interactive segmentation, not recognition.

More relevant to our problem are methods that aim to interactively annotate a given example, and thus try to optimize exactly what should be requested from the user *for that particular example*. For instance, in video segmentation, the most useful frames to annotate are found with tracking uncertainty measures [25, 26, 23]. In object recognition, a human is asked to click on object parts, depending on what seems most informative [27]. In interactive co-segmentation, the system guides a user to scribble on certain areas of certain images to reduce foreground uncertainty [1, 28]. Like us, all these methods try to reduce human effort. However, whereas prior work predicts *which images should be annotated* (and possibly where) to minimize uncertainty, we predict *what strength of annotation will be sufficient* for interactive segmentation to succeed.

Furthermore, whereas existing methods assume a back-and-forth with the annotator, we take a "one-shot" approach that makes all requests simultaneously, a potential advantage for crowdsourcing or mobile interface settings.

Limited prior work considers estimating how difficult an image is to segment. In [22], a classifier is learned to map image features to the expected time it will take for a human to segment all objects with polygons. In [13], global image features are used to predict the segmentation accuracy of an algorithm before it is applied. A related idea is to run a segmentation algorithm, and then predict how good its results are based on "object-like" descriptors of the regions [18, 4, 5, 9]. Unlike any of these methods, we want to predict difficulty for a segmentation algorithm as a function of the strength of a human's partial input.

3. Approach

First we define the annotation modes and interactive segmentation model our method targets (Sec. 3.1). Then, we define features indicative of image difficulty and learn how they relate to segmentation quality for each annotation mode (Sec. 3.2). Given a novel image, we forecast the relative success of each modality (Sec. 3.3). This allows our method to select the modality that is sufficient for an individual image. Finally, we propose a more involved optimization strategy for the case where a batch of images must be segmented in a given time budget (Sec. 3.4).

3.1. Interactive segmentation model

In interactive segmentation, the user indicates the foreground with some mode of input. Our goal is to predict the input modality that will be sufficiently strong to yield an accurate segmentation.

Our approach chooses from three annotation modalities, as depicted in Figure 2: (1) **Bounding box:** The annotator provides a tight bounding box around the foreground objects. This is typically the fastest input modality. (2) **Sloppy contour:** The annotator draws a rough contour surrounding the foreground. This gives a tighter boundary than a box (i.e., encompassing fewer background pixels) and offers cues about the object shape. It typically takes longer. (3) **Tight polygon:** The annotator draws a tight polygon along the foreground boundaries. We equate a tight polygon with perfect segmentation accuracy. This is the slowest modality. All three are intuitive and well-used tools. Our method extends naturally to handle other modalities where a user specifies foreground pixels (e.g., scribbles).

No matter the annotation mode, we use the pixels inside and outside the user-marked boundary to initialize the foreground and background models, respectively. Specifically, we use them to construct two Gaussian mixture models in RGB color space, G_{fg} and G_{bg}. Then we apply standard graph-cut based interactive segmentation [2, 19] with the mixture models as likelihood functions. Each image pixel is a node, and edges connect neighboring pixels. The objective is to assign a binary foreground/background label $y_p \in \{1, 0\}$ to each pixel p so as to minimize the total energy of all labels L:

$$E(L) = \sum_p A_p(y_p) + \sum_{p,q \in \mathcal{N}} S_{p,q}(y_p, y_q), \qquad (1)$$

where $A_p(y_p) = -\log P(F_p | G_{y_p})$ is the unary likelihood term indicating the cost of assigning a pixel as fg/bg, and F_p denotes the RGB color for pixel p. The term $S_{p,q}(y_p, y_q) = \delta(y_p \neq y_q) \exp(-\beta \|F_p - F_q\|)$ is a standard smoothness prior that penalizes assigning different labels to neighboring pixels that are similar in appearance, where β is a scaling parameter and $\mathcal{N}$ denotes a 4-connected neighborhood.

We use the algorithm of [3] to minimize Eqn. 1, and use the GrabCut idea of iteratively refining the likelihood functions and the label estimates [19].

(a) Bounding box (b) Sloppy contour (c) Tight polygon

Figure 2: Possible modes of annotation

3.2. Learning segmentation difficulty per modality

Having defined the annotation choices and the basic engine for segmentations, we can now explain our algorithm's training phase. The main idea is to train a discriminative classifier that takes an image as input, and predicts whether a given annotation modality will be successful once passed to the interactive graph cuts solver above. In other words, one classifier will decide if an image looks "easy" or "difficult" to segment with a bounding box, another classifier will decide if it looks "easy" or "difficult" with a sloppy contour.

To compose the labeled training set, we require images with ground truth foreground masks. For each training example, we want to see how it would behave with each user input mode. For the bounding box case, we simply generate the bounding box that tightly fits the true foreground area. For the sloppy contour case, we dilate the true mask by 20 pixels to simulate a coarse human-drawn boundary.[1] After running graph cuts (optimizing Eqn. 1) for each one in turn, we obtain two *estimated foreground masks* per training image: fg_{box} and fg_{con}.

We use those masks to extract a series of features (defined next), then train the two SVM classifiers. Let O denote the normalized overlap between an estimated mask and the true foreground. Let $\bar{O}_{box}$ and $\bar{O}_{con}$ denote the median

[1] In a user study, we find these masks are a good proxy; on average, they overlap with actual hand-drawn contours by 84%.

overlap among all training images for the two modes. The ground truth label on an image is positive ("easy", "successful") for an annotation modality x if $O > \bar{O}_x$. That is, the image is easy for that particular form of user input if its accuracy is better than at least half of the examples.[2]

Next we define features that reveal image difficulty. Graph cut segmentation performance is directly related to the degree of separation between the foreground and background regions. It tends to fail if the two are similar in appearance, or if the foreground object has a complex composition. Furthermore, the notion of separability is tied to the form of user input. For example, a bounding box input can fail even for an object that is very distinct from its background if it contains many background pixels. Our features take these factors into account.

Let I_{FG} be an estimated foreground (as specified by either mask fg_{box} or fg_{con} in a training image), and let I_{BG} denote its complement. We define the following features:

Color separability: Since the segmentation model depends on fg and bg appearance, we compute dissimilarity measures between them. We record the χ^2 distance between the color histograms computed from I_{FG} and I_{BG} in RGB (16 bins per channel) and Lab (21 bins per channel) color space. We also consider local color dissimilarity by computing the χ^2 distance between the RGB color histogram from I_{FG} and from a small 40-pixel region around I_{FG}. This captures how distinct the region is from its neighboring pixels. Finally, we record the KL-divergence between Gaussian mixture models estimated with I_{FG} and I_{BG}.

Edge complexity: We expect edges to reflect the complexity of a foreground object. We record a 5-bin edge orientation histogram from I_{FG}. We do this only for the foreground, as we do not want the annotation choice to be affected by background complexity. Next, as a measure of image detail, we compute the sum of gradient magnitudes for I_{FG} and I_{BG}, normalized by their areas. We record the ratio between foreground and background image detail.

Label uncertainty: Our next feature directly captures how uncertain the segmentation result is. We use the dynamic graph cuts approach proposed in [11] to compute the min-marginal energies associated with each pixel's graph cut label assignment. We map them to uncertainty by computing the change in min marginal energy when a pixel is constrained to take the non-optimal label, and record a 5-bin histogram of the uncertainty values within I_{FG}. Intuitively, an easy segmentation will have mostly labels with low uncertainty, and vice versa.

Boundary alignment and object coherence: We expect easy segments to align well with strong image boundaries. To estimate the extent of alignment, we first divide the image into superpixels [6]. For every superpixel that lies on the boundary between I_{FG} and I_{BG}, we see what fraction of its area lies inside I_{FG}. We record the average across all superpixels as a feature. We also use number of connected components in the resulting segmentation as a measure of how coherent the object is.

Altogether, we have 17 features: 4 for color separability, 6 for edge complexity, 5 for label uncertainty, and 2 for boundary alignment and coherence. We stress that all features are object- and dataset-independent. This is important so that we can learn the abstract properties that reflect segmentation difficulty, as opposed to the specific appearance of previously seen objects that were difficult to segment.

3.3. Predicting difficulty on novel images

Given a novel image, we predict which of the annotation modes will be successful. To do so, we need a coarse estimate of the foreground in order to compute the features above. We use a four step process. First, we apply a salient object detector that outputs a pixel-wise binary saliency map [14]. Second, we refine it with "superpixel smoothing", assigning the foreground label to each superpixel that overlaps a salient region by more than 50%. This yields a more coherent estimate aligned with strong image boundaries. Third, if we have multiple input images similar in appearance (i.e., the co-segmentation case), we further reclassify each superpixel using an SVM trained with superpixel instances originating in the current fg-bg masks. Finally, we automatically generate a bounding box and sloppy contour (by dilation), and run graph cuts to get the estimated masks for either modality. We use these estimates for I_{FG} (and their complements for I_{BG}) to compute the features defined above. While often an image has a primary foreground object of interest, our method (like any graph cuts formulation) can accommodate foregrounds consisting of multiple disconnected regions.

The foreground estimate in a test image need only give a rough placement of where the user might put the bounding box or sloppy contour. Indeed, the whole purpose of our work is to get the necessary guidance from a user. Nonetheless, the estimates must be better than chance to ensure meaningful features. We find the saliency-based initializations are a reasonable proxy (overlapping 47-71% on average for our datasets), though in no way replace the real human input that we will seek after applying our method.

Now we apply the difficulty classifiers to the test image. Recall that to properly balance effort and quality, our objective is to predict which mode is *sufficiently strong*. Always requesting tight polygons is sure to yield accurate results, but will waste human effort when the image content is "easy". Similarly, always requesting a bounding box is sure to be fast, but will produce lousy results when the image is

[2] While a regression model would also be a reasonable choice here, we found classification more effective in practice, likely because of the large spread in the overlap scores obtained through graph cuts segmentation.

too "hard". Therefore, if given a single image as input, we use a cascade to request the fastest annotation that is likely to succeed. That is, we show the annotator a bounding box tool if the bounding box classifier predicts "easy". If not, we show the sloppy contour tool if its classifier predicts "easy". If not, we show the user the tight polygon tool.

3.4. Annotation choices under budget constraints

In an alternative usage scenario, our system accepts a batch of images and a budget of annotation time as input. Our objective is to select the optimal annotation tool *for each image* that will maximize total predicted accuracy, subject to the constraint that annotation cost must not exceed the budget. This is a very practical scenario. For example, today's data collection efforts often entail posting annotation jobs to a crowdsourcing service like Mechanical Turk; a researcher would like to state how much money (i.e., worker time) they are willing to spend, and get the best possible segmentations in return.

For a high budget, a good choice may be tight polygons on all of the hardest images, and sloppy contours on the rest. For a low budget, it might be bounding boxes on all but the most difficult cases, etc. Rather than hand code heuristics to capture such intuitions, we propose to automatically optimize the selection. Formulating the problem is possible since we explicitly account for the expected success/failure of a particular kind of user input for a given image.

Suppose we have n images to segment, and a budget of B, which could be specified in minutes or dollars. Let p_k^b and p_k^c denote the probability of successful interactive segmentation for image k with a bounding box or sloppy contour, as predicted by our model. We map the easy/difficult classifier outputs to probabilities of success using Platt's method. Let p_k^p denote the probability of success when using a tight polygon; by definition, $p_k^p = 1$. Let $\mathbf{x} = [x_1^b, x_1^c, x_1^p, \ldots, x_n^b, x_n^c, x_n^p]$ be an indicator vector with three entries for each image, reflecting the three possible annotation modalities we could apply to it. That is, $x_k^b = 1$ would signify that image k should be annotated with a bounding box. Let $\mathbf{c} = [c_1^b, c_1^c, c_1^p, \ldots, c_n^b, c_n^c, c_n^p]$ be a cost vector, where c_k^a denotes the cost associated with annotating image k with annotation type a, specified in the same units as B. That is, $c_k^b = 7$ means it will take 7 sec to draw a bounding box on image k.

We formulate the following objective to solve for the best batch of sufficiently strong annotations:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \sum_{k=1}^{n} p_k^b x_k^b + p_k^c x_k^c + p_k^p x_k^p, \quad (2)$$
$$\text{s.t. } \mathbf{c}^T\mathbf{x} \leq B,$$
$$x_k^b + x_k^c + x_k^p = 1, \ \forall k = 1, \ldots, n,$$
$$x_k^b, x_k^c, x_k^p \in \{0, 1\}, \ \forall k = 1, \ldots, n.$$

The objective says we want to choose the modality per image that will maximize the predicted accuracy. The first constraint enforces the budget, the second ensures we choose only one modality per image, and the third restricts the indicator entries to be binary. We maximize the objective using a linear programming (LP) based branch and bound method for solving integer programs, which finds the optimal integer solution by solving a series of successive LP-relaxation problems. It takes less than a minute to solve for about 500 images and 70 budget values.

While our approach supports image-specific annotation costs c_k, we find the biggest factor in cost is which annotation type is used. Therefore, we let c_k^b, c_k^c and c_k^p each be constant for all images k, based on real user time data. One could optionally plug in fine-grained cost predictions per image when available, e.g., to reflect that high curvature contours are more expensive than smooth ones.

4. Results

We evaluate on three public datasets that provide pixel-level labels: **Interactive Image Segmentation (IIS)** [7] consists of 151 unrelated images with complex shapes and appearance; **MSRC** contains 591 images, and we convert the multi-class annotations [15] to fg-bg labels by treating the main object(s) (cow, flowers, etc.) as foreground; **CMU-Cornell iCoseg** [1] contains 643 images divided into 38 groups with similar foreground appearance, allowing us to demonstrate our method in the optional co-segmentation setting. On MSRC, we never allow the same object class to appear in both the training and test sets, to prevent our method from exploiting class-specific information.

We compare to the following methods:

- **Otsu: [17]** finds the optimal grayscale threshold that minimizes the intra-class variance between foreground and background. To use it to estimate fg-bg separability, we compute the *inter*-class variance (at the optimal threshold) and normalize by total variance. Higher values indicate higher separability, and hence "easier" segmentation.

- **Effort Prediction: [22]** predicts whether an image will be easy or hard for a human to segment, using features indicative of image complexity. We use the authors' public code. This is a state-of-the-art method for estimating image difficulty.

- **Global Features:** We train two SVMs (one for bounding box, one for contours) to predict if an image is easy based on a 12-bin color histogram, color variance, and the separability score from [17]. This baseline illustrates the importance of our features capturing the estimated foreground's separation from background.

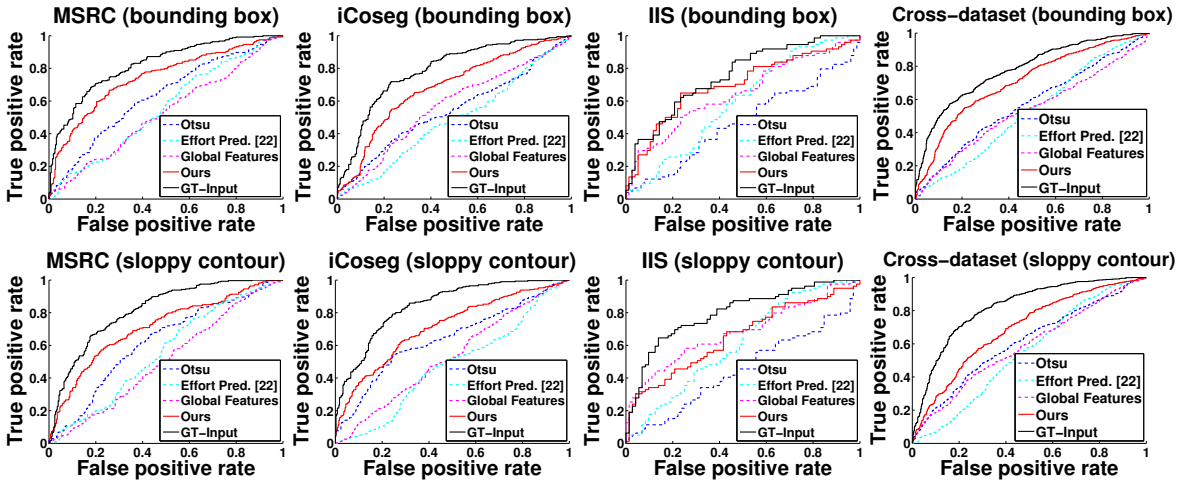


Figure 3: Difficulty prediction accuracy for each dataset (first three columns) and cross-dataset experiments (last column)

- **GT-Input**: uses the ground-truth box/contour masks as input to our method, showing the impact of our features in the absence of errors in the saliency step.

- **Random:** randomly assigns a confidence value to each modality in the budgeted annotation results.

Otsu and Effort Prediction use the same function for both boxes and contours, since they cannot reason about the different modalities. Note that methods for active interactive (co-)segmentation [1, 28] address a different problem, and are not comparable. In particular, they do not predict image difficulty, and they assume a human repeatedly gives feedback on multiple images with the same foreground.

All classifiers are linear SVMs, and the parameters are chosen by cross-validation. We quantify segmentation accuracy with the standard overlap score $\frac{P \cap GT}{P \cup GT}$ between the predicted and ground truth masks P and GT.

Predicting difficulty per modality First we see how well all methods predict the success of each annotation modality. We test both in a *dataset-specific* and *cross-dataset* manner. For the former, we test in a leave-one-out (IIS, MSRC) or leave-one-group-out (iCoseg) fashion. For the latter, we test in a leave-one-dataset-out fashion. We use each method's confidence on the test images to compute ROC curves.

Figure 3 shows the results. Our approach consistently performs well across all datasets, while none of the baselines has uniform performance (e.g., Otsu beats other baselines on MSRC, but fails badly on IIS). On MSRC and iCoseg, our approach significantly outperforms all the baselines, including the state-of-the-art Effort Prediction [22]. On IIS, we are again better for bounding boxes, but Global Features is competitive on sloppy contours. We attribute this to the complex composition of certain images in IIS that makes saliency detection fail.

In the even more challenging cross-dataset setting (Fig. 3, right column), our advantage remains steady. This is a key result. It shows our method is learning which generic cues indicate if a modality will succeed—not some idiosyncrasies of the particular objects or cameras used in the datasets. Whereas the Global Features and Effort Prediction [22] methods learn from the holistic image content, our method specifically learns how fg-bg separability influences graph cuts segmentation. Analyzing the linear SVM weights, we find label uncertainty, boundary alignment, and χ^2 color distance are the most useful features. The GT-Input result underscores the full power of the proposed features.

Figure 4 shows our typical success and failure cases. For the leftmost block of images, our method predicts a bounding box or contour would be sufficient. These images usually have uniform backgrounds, and distinct, compact foreground regions, which are easy to tightly capture with a box (e.g., flower, cows). For the center block, our method predicts a bounding box would fail, but a sloppy contour would be sufficient. These images usually have objects with complex shapes, for which even a tight box can overlap many background pixels (e.g., Christ the Redeemer, Taj Mahal). For the rightmost block, our method predicts neither a box or contour is sufficient. These images contain objects with intricate shape (e.g., bicycle) or high similarity to background (e.g., elephant, bird). Notably, the same object can look easy or difficult. For example, the skaters in the left block are close together and seem easy to annotate with a box, while the skaters in the right block are far apart and tight polygons are needed to extract their limbs. This emphasizes the object-independence of our method; its predictions truly depend on the complexity of the image.

Failures can occur if the salient region detection fails drastically (e.g., in the person image on right, the salient white shirt leads our method to think the image looks easy). We can also fail by overestimating the difficulty of images with low color separability (e.g., shadows in Stonehenge

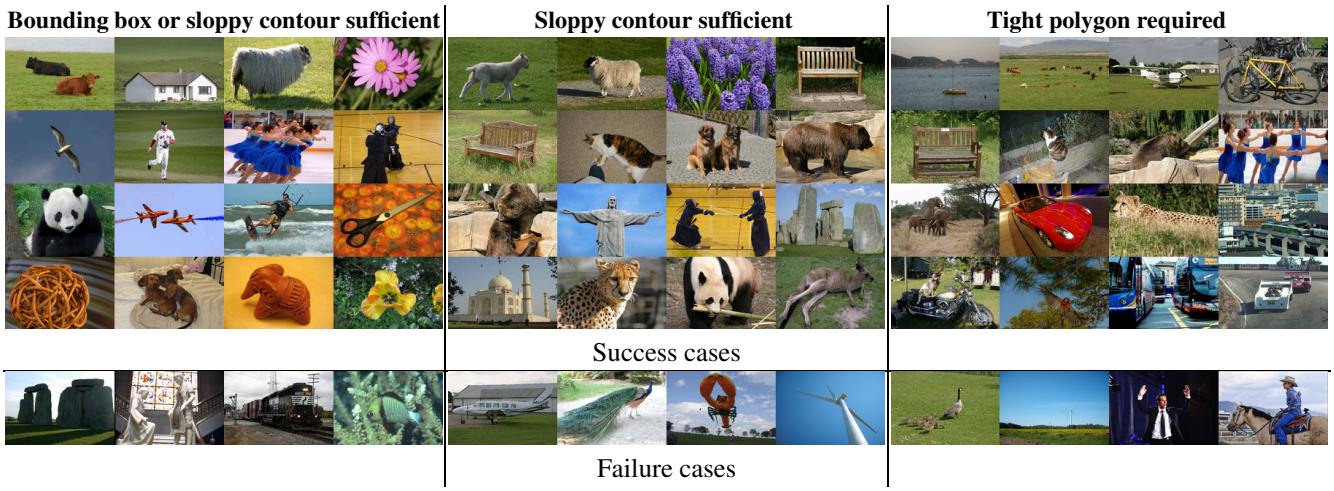


Figure 4: Example success (top) and failure cases (bottom) for our method, per annotation modality. Best viewed on pdf.

and white pixels by statues in left group), suggesting a more refined edge detector could help.

Annotation choices to meet a budget Next we evaluate our idea for optimizing requests to meet a budget. We apply our method and the baselines to estimate the probability that each modality will succeed on each image. Then, for each method, we use our budget solution defined in Sec. 3.4 to decide which image should get which modality, such that total annotation time will not exceed the budget. For the cost of each modality in $\mathbf{c}$, we use the average time required by the 101 users in our user study: 7 sec for bounding box, 20 sec for sloppy contour, 54 sec for tight polygon. If the solution says to get a box or contour on an image, we apply graph cuts with the selected modality (Sec. 3.1). If the solution says to get a tight polygon, we simply use the dataset ground truth, since it was obtained with that tool. The final accuracy is the overlap in the estimated and ground truth foregrounds over all images.

Figure 5 plots the results as a function of budget size. The budget values range from the minimum possible (bounding boxes for all images) to the maximum possible (tight polygons for all images). Our method consistently selects the modalities that best use annotation resources: at almost every budget point, we achieve the highest accuracy.[3] This means our method saves substantial human time. For example, in the cross-dataset result on 1,351 images, the best baseline needs 2.25 hours more annotation effort than we do to obtain 90% average overlap.

What choices does our method typically make? We find as the budget increases, the bounding box requests decrease. The number of sloppy contour requests increases at first, then starts decreasing after a certain budget, making way for more images to be annotated with a tight polygon. For images where either a box or contour is likely to succeed, our method tends to prefer a box so that it can get a tight polygon for more images within the budget.

[3]By definition, all methods yield the same solution for the two extremes, and hence the same accuracy.

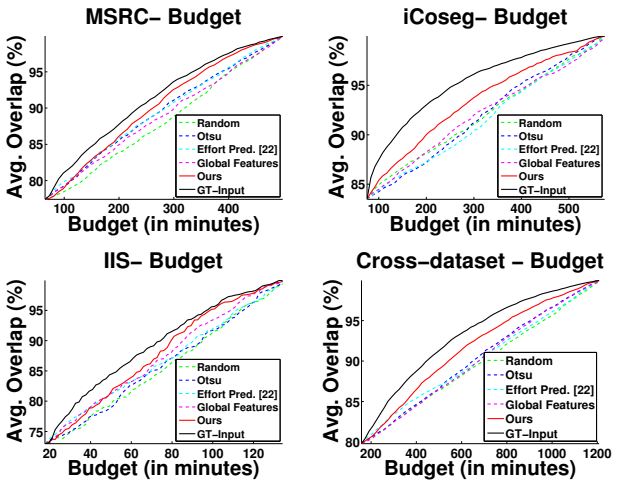


Figure 5: Choosing annotation modalities to meet a budget

Application to recognition To further illustrate the practical impact of our approach, we next apply it to train a recognition system for MSRC. Suppose we are given a set of images known to contain an object category of interest amidst a cluttered background. The goal is to learn a classifier that can differentiate object vs. non-object regions. Rather than ask an annotator to give tight polygons on each training image—the default choice for strongly supervised recognition systems—we apply our cascaded modality selection. Then we train the classifier with the resulting 15-30 interactively segmented images, and apply it to localize the object in novel images. We test in a leave-one-image out setting (more setup details in Supp.).

Table 1 shows the results. Our approach substantially reduces the total annotation time required, yet its accuracy

Object	Avg. overlap (%)		Time saved (%)
	All tight	Ours	
Flower	65.09	65.60	21.2 min (73%)
Car	60.34	60.29	3.9 min (15%)
Cow	72.90	66.53	9.2 min (68%)
Cat	51.79	46.56	13.7 min (23%)
Boat	51.08	50.77	1.4 min (10%)
Sheep	75.90	75.59	17.2 min (64%)

Table 1: We train a recognition system more efficiently by using the modality predicted to be sufficiently strong.

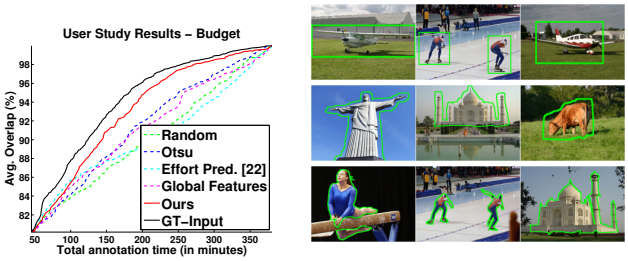


Figure 6: **Left:** Annotation choices under a budget with real user data. **Right:** Example user annotations for bounding box (top), sloppy contour (middle), and tight polygon (bottom).

on novel images is still very competitive with the method that gets perfect tight polygons on all images.

User study Finally, we conduct a user study with Mechanical Turk workers. We randomly select one third of the images from each dataset to make a diverse pool of 420 images. We present users with the necessary tools to do each modality (see Supp. for interface details), and time them as they work on each image. If an object has multiple foreground objects, they must annotate each one. We collect responses from 5 users for each annotation mode per image, then record the median time spent. In total, we obtain 2,100 responses per modality, from 101 unique users.

Figure 6 (right) shows example user annotations. We see the most variance among the sloppy contour inputs, since some users are more "sloppy" than others. Still, as expected, sloppy contours typically only improve interactive segmentation results (85.5% average overlap accuracy) compared to the faster bounding boxes (82.1% average overlap accuracy).

Figure 6 (left) shows the budgeted annotation results with real user data. The plot is like Figure 5, only here 1) we feed the real users' boxes/contours to the graph cuts engine, rather than simulate it from ground truth masks, and 2) we incur the users' per-image annotation times at test time (on x-axis). Across all budgets, our method allocates effort more wisely, and it even narrows the gap with the GT-Input. This result confirms that even though the ultimate annotation time may vary not only per modality, but also per image, using a fixed cost per modality during *prediction* is sufficient to get good savings. Overall, this large-scale user study is promising evidence that by reasoning about the expected success of different annotation modalities, we can use valuable annotator effort much more efficiently.

Acknowledgements: This research is supported in part by ONR YIP N00014-12-1-0754.

References

[1] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen. iCoseg: Interactive co-segmentation with intelligent scribble guidance. In *CVPR*, 2010. 1, 2, 5, 6

[2] Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *CVPR*, 2001. 1, 2, 3

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11):1222–1239, Nov. 2001. 3

[4] J. Carreira and C. Sminchisescu. CPMC: Automatic object segmentation using constrained parametric min-cuts. *PAMI*, 34(7):1312–1328, 2012. 3

[5] I. Endres and D. Hoiem. Category independent object proposals. In *ECCV*, 2010. 3

[6] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, Sept. 2004. 4

[7] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman. Geodesic star convexity for interactive image segmentation. In *CVPR*, 2010. 1, 2, 5

[8] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *IJCV*, pages 321–331, 1988. 1, 2

[9] T. Kohlberger, V. Singh, C. Alvino, C. Bahlmann, and L. Grady. Evaluating segmentation error without ground truth. In *MICCAI*, 2012. 3

[10] P. Kohli, H. Nickisch, C. Rother, and C. Rhemann. User-centric learning and evaluation of interactive segmentation systems. *IJCV*, 100(3):261–274, Dec. 2012. 2

[11] P. Kohli and P. H. S. Torr. Measuring uncertainty in graph cut solutions. *CVIU*, 112(1):30–38, 2008. 4

[12] V. S. Lempitsky, P. Kohli, C. Rother, and T. Sharp. Image segmentation with a bounding box prior. In *ICCV*, 2009. 1, 2

[13] D. Liu, Y. Xiong, K. Pulli, and L. Shapiro. Estimating image segmentation difficulty. In *Machine learning and data mining in pattern recognition*, 2011. 3

[14] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum. Learning to detect a salient object. *PAMI*, 33(2):353–367, Feb. 2011. 4

[15] T. Malisiewicz and A. A. Efros. Improving spatial support for objects via multiple segmentations. In *BMVC*, 2007. 5

[16] E. Mortensen and W. Barrett. Intelligent scissors for image composition. In *SIGGRAPH*, 1995. 1, 2

[17] N. Otsu. A Threshold Selection Method from Gray-level Histograms. *IEEE Trans on Sys, Man and Cybernetics*, 9(1):62–66, Jan. 1979. 5

[18] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*, 2003. 3

[19] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": interactive foreground extraction using iterated graph cuts. 2004. 1, 2, 3

[20] B. Siddiquie and A. Gupta. Beyond Active Noun Tagging: Modeling Contextual Interactions for Multi-Class Active Learning. In *CVPR*, 2010. 2

[21] A. Vezhnevets, J. Buhmann, and V. Ferrari. Active learning for semantic segmentation with expected change. In *CVPR*, 2012. 2

[22] S. Vijayanarasimhan and K. Grauman. What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. In *CVPR*, 2009. 2, 3, 5, 6

[23] S. Vijayanarasimhan and K. Grauman. Active frame selection for label propagation in videos. In *ECCV*, 2012. 2

[24] S. Vijayanarasimhan, P. Jain, and K. Grauman. Far-sighted active learning on a budget for image and video recognition. In *CVPR*, 2010. 2

[25] C. Vondrick and D. Ramanan. Video annotation and tracking with active learning. In *NIPS*, 2011. 2

[26] C. Vondrick, D. Ramanan, and D. Patterson. Efficiently scaling up video annotation with crowdsourced marketplaces. In *ECCV*, 2010. 2

[27] C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *ICCV*, 2011. 2

[28] D. Wang, C. Yan, S. Shan, and X. Chen. Active learning for interactive segmentation with expected confidence change. In *ACCV*, 2012. 2, 6