Draw&Tell: Efficient Annotation for Semantic Image Segmentation by Drawing and Speaking

Anonymous ECCV submission

Paper ID 621

Abstract. The performance of object recognition systems is largely dependent on the quantity and quality of training annotations. Most current methods focus on recognition accuracy and neglect the cost of obtaining training samples. In this paper, we propose a natural, cheap annotation method by letting annotators speak, leading to an approach which draws a good trade-off between recognition accuracy and annotation cost. In particular, we let annotators draw scribbles on objects of interest while they say their class names. The speech is recognized by a speech recognition engine that has been specifically trained for the purpose, and the coarse annotation is explored by an extension of standard CNN models. Since drawing scribbles and speaking are very natural to human, our method unleashes the expressive ability of annotators and solves the *what* and *where* problems of object annotation both at the same time. We show in the context of semantic image segmentation that (i) our annotation requires 11 times less effort than conventional full-mask annotation while obtaining comparable recognition accuracy; (ii) our annotation is also faster than image-level annotation while yielding significantly better results; and (iii) under the same annotation budget, our annotation, coupled by our learning method, yields better results than the conventional, full-mask annotations. We believe that letting annotators speak is a natural and effective way for many other annotation tasks in vision, especially those relevant to semantics.

Keywords: Speech-based Annotation, Semantic Image Segmentation, Weak Supervision, Integration of Vision and Speech

1 Introduction

Tremendous progress has been made during the last few years in object detection and semantic image segmentation due to (1) the success of training deep convolutional neural networks [**?**, **?**, **?**] on large classification datasets and (2) the flexibility of transferring CNN models pre-trained for image classification to the task of object detection and semantic segmentation where not as much training data is available [**?**, **?**, **?**]. Model transfer, often called fine-tuning, partially lifts the strong need for large training sets of CNNs. Yet, the fact remains that CNNs *intrinsically* call for large-scale training data to unleash their learning capacity. As such, we can expect that the limit of CNNs for the tasks is far from reached and that one of the obstacles is the cost of obtaining training samples.

Obtaining training data for object detection or semantic image segmentation consists of two main jobs: answering the *what* and *where* questions. *What* is to annotate

(a) masks [🖱] (b) b-boxes [🖱] (c) points [🖱] (d) keywords [🖱] **(e) scribbles** [🖱+🎙]

Fig. 1. A comparison of our annotation (e) to other forms of annotations for semantic segmentation, ranging from (a) full segmentation masks, to (b) bounding boxes, to (c) single points, and to (d) image-level keywords. The devices to obtain these annotations are shown as well: others only use the mouse (+keyboard), while ours also uses voice input through a microphone.

what objects are present and *where* is to indicate their locations in the image. As to *what*, annotators usually need to choose class names from some form of menus with the mouse and/or keyboard [**?**]. The procedure can be costly especially when the number of classes considered is large. Imagine the amount of work for annotators to find the class name out of a list of hundreds of names for every object they annotate, not even to mention attributes, views, etc. In other systems [**?**, **?**], binary questions ask whether object classes are present, which is also very expensive when the number of classes is large, though exploiting the hierarchy of classes may reduce the cost [**?**, **?**]. As to *where*, various forms of annotations have been used for training: from the full segmentation masks [**?**, **?**, **?**], over bounding boxes [**?**, **?**, **?**] or just single points [**?**], to nothing spatial at all (image-level annotation) [**?**, **?**]. As can be seen in Fig.**??**, the more specific the annotation, the more informative it is, but also the more expensive to obtain.

In this paper, we present a novel annotation method, which strikes a balance between annotation cost and informativeness provided. In particular, annotators are asked to draw scribbles (strokes) on objects of interest, while saying aloud their class names (attribute if necessary). The scribbles are recorded to solve the *where* problem; the speech is recognized by a specially trained recognition engine to solve the *what* problem. In order to further increase the recognition accuracy of the object classes, we integrate the speech recognition with a webly-supervised object recognition. The object recognition system is trained with images retrieved from image searching engine directly, and is applied to the image region where the scribble is drawn. An example of the annotation by our method is shown in Fig. **??**, along with that of other methods.

The method is inspired by two observations: (i) *what* and *where* are handled separately in previous methods. For instance, in [**?**] annotators first mask out a region and then assign a label to it; and in [**?**] annotators label the presence of objects in the first round, followed by positioning them in the second round. This separation is unnecessary and introduces extra cost, because the two problems are interdependent and solved together by the annotators vision. It seems the separation is due to the fact that only a single mode of input devices was used in the previous methods, which is the mouse (+keyboard). We lift this restriction by including the speech channel, which allows annotators to communicate with the computer more naturally and efficiently. (ii) Drawing scribbles is another natural and efficient ability, and has been widely used for interactive image segmentation. But to the best of our knowledge, it has not been applied to create training data for semantic image segmentation. We demonstrate that combining drawing and speaking can tackle both the *what* and *where* problems, leading to an efficient annotation method for semantic image segmentation. We call the method **Draw&Tell**.

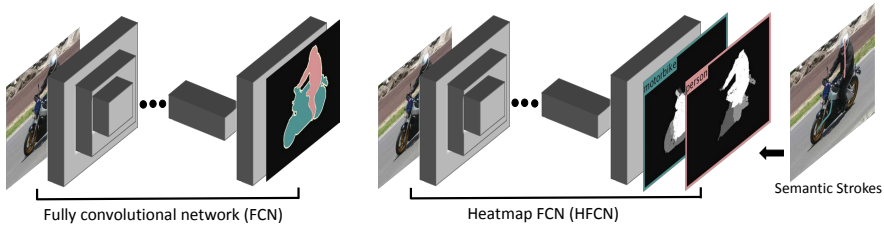


Fig. 2. The framework of the standard fully convolutional network (FCN) [**?**] and our heatmap-based FCN.

Having obtained the annotations, we convert them to soft confidence maps of the corresponding classes by combining interactive image segmentation and ensemble learning. We call these soft confidence maps semantic heatmaps. Finally, we extend the standard CNN model [**?**] to accommodate the soft semantic heatmaps as training data rather than the standard crisp labels. The pipeline of the method is shown in Fig.**??** (right hand side), which also shows the standard fully convolutional network [**?**] for ease of comparison. We show in experiments (i) that our annotation method is 11 times faster than pixel-wise annotation, and also faster than conventional image-level annotation; (ii) that our adapted CNN trained with the annotations yields significantly better results than the CNN trained with image-level training data, and yields results comparable with those of the CNN trained with pixel-wise annotations; and (iii) that under the same annotation budget, our annotations, combined with our learning method, yield better results than the conventional precise annotations.

Our contributions are mainly: (i) a new method which combines webly-supervised vision and automatic speech recognition to efficiently create training data for semantic image segmentation, and (ii) a method to train CNNs with scribble-based training data, by converting scribbles to soft heatmaps and extending the standard neural networks. Introducing speech recognition into visual annotation is novel and our method can be used for other vision tasks as well, such as annotations for object detection, part detection, attribute detection and view estimation. The integration of webly-supervised object recognition and speech recognition can also serve as an example of combing vision and speech.

The paper is organized as follows. Sec. **??** presents related work. Sec. **??** details the annotation method, followed by Sec. **??** which is devoted to the segmentation method. Sec. **??** then reports on the experiments and Sec. **??** concludes the paper.

2 Related Work

Previous work related to ours mainly falls into two groups: semantic image segmentation and integration of vision and language (speech).

2.1 Semantic Image Segmentation

Methods: There is a rich literature of semantic image segmentation (SIS) [**?**, **?**], and the field has made tremendous progress since CNNs were applied. The seminal R-CNN [**?**]

and the follow-up systems [**?**, **?**, **?**] yield significantly better performance than previous methods. As noted above, SIS through CNNs has probably not come to full fruition yet due to the small size of the existing training sets. Several successful attempts have been made to use weaker supervision in order to reduce the annotation cost. For instance, [**?**] exploits multiple instance learning to train FCN [**?**] with image-level annotations. [**?**, **?**, **?**] leverage the power of object proposals to train FCN with object bounding-boxes. [**?**] presents a system to train FCN with point annotation – objects are indicated by single points. In the same vein our work tries to train FCN with annotations that are efficient to obtain.

Supervision: Datasets play a critical role in computer vision. They qualitatively 'define' the learning tasks and guide research directions. Training annotations for SIS often come as full segmentation masks (c.f. Fig.**??**). The most popular ones fall into this category: CamVid [**?**] for outdoor scenes, NYU [**?**] for indoor scenes, PASCAL [**?**] and COCO [**?**] for general objects, and PASCAL-Context [**?**] for objects in context. Creating datasets for SIS is very expensive even with excellent annotation tools [**?**, **?**]. As a result, methods were proposed to reduce the cost. For instance, [**?**, **?**] exploit the hierarchical structures of object classes to reduce annotation space. [**?**, **?**] exploit active learning techniques to suggest the most informative samples to annotate under a budget. [**?**] employs eye tracking systems to help create training data for object detection. Our proposal is to exploit speech recognition to help dataset creation for SIS.

2.2 Integration of Vision and Language (Speech)

Research interest in integrating vision and language has increased recently, e.g. for image/video caption generation [**?**, **?**, **?**, **?**, **?**]. The objective is to learn from a corpus of sentences and images / video snippets to generate meaningful descriptions for new images. One of the main research topics is the alignment of representations from the two different domains: vision and language. In terms of language&vision understanding, our goal is more conservative, because the words we handle are restrictive and vision and language are aligned with human help. However, our purpose of the integration is different.

Speech-driven interfaces are going through a renaissance, with popular commercial products like Apple's Siri. The most relevant to our work are those integrating speech and vision. Pixeltone [**?**] and Image spirit [**?**] are examples that use voice commands to guide image processing and semantic segmentation. The difference between image spirit and our work is that they use voice commands to refine labeling results and we use them to collect training data. Also, image spirit uses voice commands alone, while our method combines speech input with mouse interactions. There also is academic work [**?**, **?**, **?**, **?**] and an app [**?**] that use speech to provide image descriptions. Again, our purpose is very different.

3 Speech-based Annotation

3.1 The Draw & Tell Annotation Tool

This section describes our annotation method, which consists of two parts: drawing scribbles on the objects and telling the system their names. Drawing scribbles is straight-

forward and we follow popular interactive image segmentation (IIS) methods [**?**] to record the mouse tracks. As to the speech recognition, we draw on the state-of-the-art, end-to-end speech recognition systems [**?**], trained with bidirectional recurrent neural networks under the connectionist temporal classification loss function. The systems successfully avoid the significant human effort in creating the pronunciation dictionary. The characters are then decoded to words with a dictionary of desirable class names by the CTC Beam Search algorithm in [**?**]. Improving such a system itself is challenging and beyond the scope of this paper, thus we choose to treat the system as a black box and operate on an n-best list of possible object names for each utterance.

Speech recognition is a complex research area in its own right, and there are still problems with recognizing natural speech with high accuracy [**?**]. We, however, want to use it to simplify the task of annotation, for which high accuracy is required. Fortunately, some constraints can be imposed in our case, which our annotation task naturally fulfills. The constraints and solutions are:

- Constraining the vocabulary and syntax of the utterances to ensure robust speech recognition. For our annotation, the vocabulary is restricted to the names of the objects of interest, and we can also instruct annotators to follow a specific syntax.
- Synchronizing the speech input with mouse input. We instruct the annotator to only say the object names while they draw the scribble on the corresponding object, not at any other time. The speech recognition engine uses this synchronization to better identify speeches relevant to the object being annotated.
- Re-ranking the n-best words by integrating information from visual object recognition, as analyzing the visual content in the drawing area provides complementary information. We choose to learn the recognizer in a webly-supervised manner, where images retrieved from image search engines such as Google and Bing are used directly for the training. This integration is also interesting in the wider context of combining vision and speech. Sec. **??** elaborates the method.

As a fall-back solution, Draw&Tell provides ways to permit correction of the recognition errors, by either repeating the operation (drawing + speaking) for the same object; or typing the object name directly. The annotator will be asked to review the name list if the system fails multiple times to recognize the speech, mainly because the spoken words are out of the dictionary. Our implementation is built on top of the open source code of [**?**].

3.2 Integration with Webly Supervised Object Recognition

As discussed in the previous section, speech recognition still has problem for high-precision applications like ours. Thus, we propose to re-rank the n-best words (object names) from the speech recognition engine by using visual information. The idea is straightforward as follows: once a scribble is drawn, a corresponding image region is cropped and fed into an object recognition system for visual recognition. The region is generated by choosing an object proposal out of 500 candidates yielded by edge-box [**?**]: (i) obtaining the enclosing bounding-box of the scribble, and (ii) choosing an object proposal with which the bounding-box has the highest intersection-over-union

score. The recognition scores are then fused with the scores from speech recognition to obtain the name of the object.

The object recognition system can be trained with samples collected by conventional annotation methods, if available. We here explore the potential of training it with web images, due to (i) an enormous amount of visual data is online to use for free; and (ii) noisy recognition (up to some level) is acceptable here as it is only used to complement speech recognition, which itself already performs well for a small dictionary of pre-defined words, though not perfect.

We follow the footprint of [**?**] to train a webly-supervised RCNN [**?**] model for object recognition. Our RCNN is trained with a classification loss on images returned by Google directly – images returned by the same keyword are taken as of the same class. The rationale is that the top images returned by Google mostly come with a clean background and a single centered object of the query class. If needed, more sophisticated techniques can be used to prepare the training data, such as seeds-based co-segmentation [**?**]. In this work, we downloaded 2500 images for each class, and fine-tuned the VGG Net [**?**] pre-trained on ImageNet [**?**]. The integration of speech recognition and webly-supervised object recognition is done by simply fusing their posterior probabilities:

$$P(y|\mathbf{v},\mathbf{x}) = \sigma P(y|\mathbf{v}) + (1-\sigma)P(y|\mathbf{x}) \quad (1)$$

where $y \in \{1,...,K\}$ denotes the object names with K the number of classes, $\mathbf{v}$ the voice represented as spectrograms, $\mathbf{x}$ the cropped image region, and σ a scalar value to balance the two terms. σ is set to 0.7 empirically in this work to put more weight on speech recognition as the object recognizer is trained with noisy data.

3.3 Annotation Results

Three results are reported: (i) recognition accuracy of object names, (ii) annotation accuracy of the scribbles, and (iii) annotation speed of our method.

Recognition accuracy for object names: Our method is evaluated with a comparison to PocketSphinx [**?**], a standard speech recognition engine. For fair comparison, we re-trained a new dictionary and language model [1] for PocketSphinx as well so that it focuses on the object names of interest. The evaluation is conducted on the box annotation of PASCAL VOC 2012 [**?**] and that of MSCOCO dataset [**?**]. An annotated object in the image is highlighted and its name is displayed. Annotators are asked to draw scribbles on the object while speak aloud the name displayed. Examples are shown in the supplementary material. The recognition result will be compared to the ground truth and the average accuracy is reported. 20 object instances are sampled for each object classes, leading to $400 = 20 \times 20$ objects for PASCAL VOC and $1760 = 20 \times 88$ objects for MSCOCO. Three annotators (graduate students) are asked to perform the annotation, and their results are averaged. Table **??** shows the results.

The results shows that our method performs significantly better than the pure speech recognition methods, namely the HMM-GMM based method PocketSphnix, and the

[1] They are trained by simply uploading the text corpus to CMUSphinx's web service [**?**].

PASCAL VOC				MSCOCO			
PocketSphnix	Ours			PocketSphnix	Ours		
	Speech	Web	Combined		Speech	Web	Combined
71.2	77.4	51.0	92.1	57.3	60.0	39.6	75.8

Table 1. Recognition accuracy (%) of object names, where speech means our method without the help of the object recognition, web denotes only using the object recognition, and combined stands for our final method.

bidirectional RNN based method [**?**, **?**]. Webly-supervised object recognition provides reasonably good results, but itself alone is still not enough to be used for the annotation task. By combining the strength of vision and speech, our method yields excellent results for the 20 classes of PASCAL VOC, and quite good results for the 88 classes of MSCOCO. The results suggest that our method is accurate enough to be used directly for annotation tasks of around 20 classes. For more classes like that in MSCOCO, people often annotate them in a hierarchical manner [**?**], leading to annotation tasks with smaller number of classes, which makes our method applicable as well. In the rest of this work, we will mainly evaluate our annotation method on the 20 classes of PASCAL VOC.

Annotation Accuracy: We report and analyze our annotation results on PASCAL VOC 2012 [**?**] here. We annotated the 20 object classes of PASCAL VOC for $12,031$ images, including all images in the training and validation set of the PASCAL VOC 2012 segmentation benchmark and all images in the augmented dataset from [**?**]. The scribbles for the background will be automatically sampled in order to reduce the annotation cost. Two annotators annotated the data, independently, with the same goal of drawing scribbles as much as possible on the objects, but on different budgets: the first one has 1.5 seconds per object, while the second one has 3 seconds. They will be referred as Anno-1.5 and Anno-3. Exemplar annotations are shown in the supplementary material.

The annotation results are listed in Table **??**. Four types of errors are reported: two at object-level and two at pixel-level. At object-level, we assign the scribbles to their closest object masks (by the number of pixels shared) and check the consistency of their labels. False positives (FP) and false negatives (FN) are reported, which are fairly few. Also, we must be aware that objects of very small size may be confusing to annotators in terms of whether they need to be annotated. These cases contributed the most of these FP and FN cases. Scribbles that are 'correct' at object-level were evaluated for their accuracy at pixel-level. We specify two errors: the percentage of pixels drawn onto background as well as onto other objects. They are actually very small, which means for the correctly detected objects human annotators can spot their position and outline very accurately. Overall, the annotations are precise, because (i) scribbles are flexible, easily adaptable to different object layouts and (ii) drawing scribbles is very natural to human.

Annotation Speed: We compare the speed of our annotation method to four other popular annotation methods (c.f. Fig.**??**): full segmentation masks, bounding boxes, singular points on objects, and image-level keywords. Because all these methods need to loop over all classes to solve the *what* problem, the minimum time - browsing time - is constant for all. According to [**?**], the time for browsing one image for one class is

Anno-1.5				Anno-3			
object		pixel		object		pixel	
false neg.	false pos.	background	other obj.	false neg.	false pos.	background	other obj.
4.7	3.8	1.7	5.1	4.5	4.0	1.0	3.8

Table 2. Errors (%) of the annotation: false positives and false negatives at object-level, and the percentages of pixels drawing to the background and objects of other classes for the scribbles which are 'correct' at object-level.

Mask	Bounding-Box	Point	Keyword	Ours (Speech-based Scribble)	
				Anno-1.5	Anno-3
104.8	39.9	23.2	20.0	6.6	11.1

Table 3. The annotation speed (seconds per image) of all methods, measured on PASCAL VOC.

1 second, which is consistent with what we found in experiments. The total browsing time per image is 20.0 seconds, which is also the time for image-level annotation. The time for other forms of annotations is the sum of this 20.0 seconds and the time for annotating 2.8 (on average) objects in each image. Point-based annotation [**?**] costs 3.2 seconds for clicking points on the objects, resulting in 23.2 seconds in total.

For the time of drawing one bounding box, different numbers are reported, varying from 7 to 25.5 seconds [**?**, **?**], probably because the types of objects being handled and the quality of the annotation are different. We experimented with images from PASCAL VOC 2012 and drew bounding boxes for 200 randomly chosen objects. The annotation time per bounding box is 7.1 seconds on average, which is quite efficient compared to the numbers reported in the literature. Thus the total time for bounding-box based annotation per image is 39.9 $(20.0 + 2.8 \cdot 7.1)$ seconds. Similarly, we annotated the masks for 200 randomly sampled objects using the annotation tool developed in [**?**]. Annotating each mask takes 30.3 seconds on average. Thus, full-mask based annotation time is 104.8 $(20.0 + 2.8 \cdot 30.3)$ seconds.

For our annotations, we allocate 2.0 seconds for browsing the image, which was found sufficient in the annotation. Drawing one scribble takes 1.5 and 3.0 seconds for Anno-1.5 and Anno-3. The recognition error rate of speech recognition is about 8% for our task, which contributes to the correction time. Putting all together, the annotation time is 6.6 $(2 + 2.8 \cdot 1.5 \cdot (1 + 0.08 + 0.08 \cdot 0.08 + ...))$ seconds per image for Anno-1.5, and 11.1 seconds for Anno-3. All the numbers are listed in Table **??**. According to the numbers, our annotation is the fastest one (faster than the image-level annotation as well), due to the help by the integration of speech recognition and webly-supervised object recognition, which lets annotators solve the *what* and *where* tasks of object annotation both at the same time. Some of these numbers are quite *rough* estimations, but they reflect the cost to a reasonable level of accuracy.

4 Semantic Image Segmentation

We follow recent methods [**?**, **?**, **?**] to fine-tune CNNs for semantic image segmentation. To this end, we convert the annotated semantic scribbles to semantic heatmaps – confidence maps of corresponding classes – and then adapt the fully connected CNNs (FCN) [**?**] to accommodate the heatmap-based training data.

(a) original scribbles (b) augmented scribbles (c) generated semantic heatmaps

Fig. 3. An example of scribble augmentation and heatmap generation: (a) annotated scribbles; (b) augmented scribbles for the background; (c) generated semantic heatmaps for the two objects.

4.1 Scribbles to Heatmaps

We convert the annotated scribbles to semantic heatmaps of all classes considered. Let $I \in \mathbb{R}^{W\times H\times Z}$ be the input image, with size $[W, H]$ and depth Z (3 for RGB images), its semantic heatmaps are denoted by $P^k \in [0,1]^{W\times H}$, where $k \in \{1, 2, ..., K\}$, and K is the number of classes considered. For classes not present in the images, their heatmaps are simply set to zero. For classes which are present, we generate the heatmap for each class individually. The heatmap is generated by a combination of interactive image segmentation (IIS) and ensemble learning. In particular, we take the scribbles of the class considered as the scribbles for the foreground object in the context of IIS, and the scribbles on other objects as that for the background. For instance if the dog class in Fig. **??**(a) is considered, the scribble on the person is taken as the scribble for background. However, as the example shows, only having the one scribble for the entire background is wanting. We want to add scribbles sampled from the rest of the background as well, thus forming an augmented scribble set. See Fig. **??**(b) for the three additional scribbles. With all scribbles in the augmented set, we run the IIS method proposed in [**?**] to get one solution to the heapmap P^k (Fig.**??**(d)), with 1 indicating foreground and 0 background. To further increase the robustness, we run the method T times with different augmented scribbles to obtain T such solutions. The final heatmap (shown in Fig. **??**(c)) for class k is computed as the average of all the individual solutions:

$$P^k = 1/T \sum_{t=1}^{T} P_t^k. \tag{2}$$

This is inspired by ensemble learning. Below we present the scribble augmentation.

Scribble Augmentation For each class, three (sufficient in practice) more scribbles are added. For simplicity, they are added sequentially. New scribbles should be far from the foreground for good separation, and far from existing background scribbles to be complementary. Below, we define the distance between the scribbles.

Given an image I, all paths that connect pixel a and pixel b are denoted by $\mathcal{Q}_{ab}$. Suppose we have a path Γ described by the pixels it passes through $\{\Gamma^1, \Gamma^2, ..., \Gamma^r\}$. The distance between the pixels Γ^1 and Γ^r along path Γ is defined as (following [**?**]):

$$D(\Gamma) = \sum_{j=1}^{r} \sqrt{d_{eu}(\Gamma^j, \Gamma^{j+1}) + \lambda \|\nabla I(\Gamma^j)\|^2}\ , \tag{3}$$

where $d_{eu}(\Gamma^j, \Gamma^{j+1})$ is the Euclidean distance between two consecutive pixels, and $\|\nabla I(\Gamma^j)\|^2$ is the gradient magnitude between (Γ^j, Γ^{j+1}), which is computed by the edge detector of the structured random forest [**?**], to avoid texture edges. λ is used to balance the two terms.

Let's denote the existing scribbles as a set of pixels $\mathcal{E}$. Then, the geodesic distance of pixel a to $\mathcal{E}$ is defined as:

$$G(a) = \min_{b \in \mathcal{E}} \min_{\Gamma \in \mathcal{Q}_{ab}} D(\Gamma) \tag{4}$$

Without any specific preference, we fix the shape of the new scribble $\bar{\mathcal{E}}$ to a disk of radius $r = 20$ (a balance between localization and informativeness). Then, the geodesic distance from the scribble centered at position a to existing scribbles $\mathcal{E}$ is:

$$G(\bar{\mathcal{E}}_a) = \min_{c \in \bar{\mathcal{E}}_a} G(c). \tag{5}$$

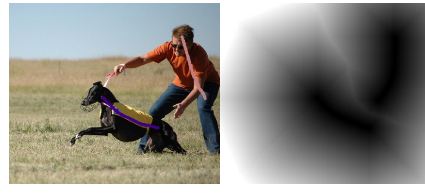
The sampling probability for the next scribble is then defined as:

$$Pr(a) = \frac{\exp(G(\bar{\mathcal{E}}_a)^2/\sigma^2)}{\sum_{b=1}^{WH} \exp(G(\bar{\mathcal{E}}_b)^2/\sigma^2)} \tag{6}$$

where σ is set adaptively to the average of all $G(\bar{\mathcal{E}}_a)$. $Pr(a)$ needs to be updated each time a new scribble is added. One example of $Pr(a)$ for all values of a is shown in Fig.**??**. The randomness introduced by this sampling allows for the ensemble approach. Note that other information such as objectness [**?**] can be exploited for sampling the background scribbles. We leave this as future work.

4.2 Heatmap-based FCN

FCN [**?**] is designed to regress the 2D label map $O \in \{1, 2, ..., K\}^{W \times H}$, directly from the input image I. The output of FCN is K score maps S, one for one class, i.e. $S \in \mathbb{R}^{W \times H \times K}$. The class labels are obtained either by simply taking the best-scoring classes at each pixel [**?**] or by using conditional random fields for further refinement [**?**]. FCN is trained to minimize the prediction error of cross-entropy over all pixels, and its optimization function is:



(a) existing scribbles (b) probability map

Fig. 4. Probability map for sampling the next new scribble (c.f. Eq.**??**).

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{a=1}^{WH} \mathcal{L}_{na}(S_{na}, O_{na}), \tag{7}$$

where N is the number of training images and $\mathcal{L}_{na}$ is the per-pixel loss function, which is commonly defined as:

$$\mathcal{L}_{na} = -\log\left(\frac{\exp(S_{na}^{O_{na}})}{\sum_{k=1}^{K} \exp(S_{na}^{k})}\right). \tag{8}$$

Supervision	Image-level	Point	Bounding-box	Full-mask	Scribbles (ours)		
Method	ConsCNN [**?**]	WhatPoint [**?**]	CNN-EM [**?**]	FCN-8s [**?**]	HFCN-1.5	HFCN-3	HFCN-3+CRF
mIoU	35.3	42.7	60.6	62.7	56.2	61.9	64.1

Table 4. The results of different methods with varying levels of supervision on the validation set of PASCAL VOC 2012.

The loss function is only computed for pixels having ground truth labels. Otherwise, the loss function is set to zero.

Directly applying FCN to our training annotations is problematic, as our annotations are soft semantic heatmaps rather than crisp segmentation masks. To solve this, we extend the loss function in Eq.**??** to the following:

$$\mathcal{L}'_{na} = \sum_{k=1}^{K} P_{na}^{k} \left[-\log\left(\frac{\exp(S_{na}^{k})}{\sum_{k'=1}^{K} \exp(S_{na}^{k'})}\right) \right]. \tag{9}$$

By the adaptation, the loss function is modulated by the heatmaps of relevant classes – for each pixel, the most confident classes affect the loss function the most. It can be seen that the loss function on crisp segmentation masks in Eq.**??** is a special case of our new loss function. The new loss function can be optimized the same way as used in standard FCN, with a modification to the loss layer. We call the model heatmap-based FCN (HFCN), and its pipeline is shown in Fig. **??**.

5 Experiments

We evaluate HFCN with different settings, and compare it to other competing methods. The goal is to show that training with scribbles-based annotations is a good trade-off between annotation cost and prediction accuracy.

5.1 Experimental Settings

Dataset: We evaluate HFCN on PASCAL VOC 2012 [**?**], which comes with three subsets: training (1464 images), validation (1449 images) and test (1456 images), having 20 object classes annotated in full segmentation masks. In order to keep the same settings with previous methods [**?**, **?**, **?**, **?**], we also extend the training set by adding the extra annotations created by Hariharan et. al. [**?**] (excluding images from the original validation set), ending up with $10,582$ training images. The method is evaluated on the validation set and the test set, under the metric intersection-over-union (IoU) for all the 20 classes.

CNN: We adopt the FCN-8s [**?**] model, as it has shown excellent performance and there is code available. The FCN model is adapted from the VGG network [**?**] pre-trained on the ILSVRC dataset [**?**]. For the optimization, we employ a procedure similar to FCN: the SGD solver is used, with an initial learning rate of 10^{-6}; the network is trained with $80,000$ iterations. HFCN is implemented with the Caffe framework [**?**].

Method	aerop	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mIoU
Image-level:																					
MIL-FCN [?]	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	24.9
Img2Pix-CNN [?]	25.4	18.2	22.7	21.5	28.6	39.5	44.7	46.6	11.9	40.4	11.8	45.6	40.1	35.5	35.2	20.8	41.7	17.0	34.7	30.4	32.6
Single-Point:																					
What's-point [?]	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	43.6
Bounding-Box:																					
CNN-EM [?]	64.4	27.3	67.6	55.1	64.0	81.6	70.5	76.0	24.1	63.8	58.2	72.1	59.8	73.5	71.4	47.4	76.0	44.2	68.9	50.9	60.8
BoxSup [?]	80.3	31.3	82.1	47.4	62.6	75.4	75.0	74.5	24.5	68.3	56.4	73.7	69.4	72.5	75.1	47.4	70.8	45.7	71.1	58.8	64.6
Full-mask:																					
SDS [?]	63.3	25.7	63.0	39.8	59.2	70.9	61.4	54.9	16.8	45.0	48.2	50.5	51.0	57.7	63.3	31.8	58.7	31.2	55.7	48.5	51.6
FCN-8s [?]	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1	62.2
Zoomout [?]	81.9	35.1	78.2	57.4	56.5	80.5	74.0	79.8	22.4	69.6	53.7	74.0	76.0	76.6	68.8	44.3	70.2	40.2	68.9	55.3	64.4
DeepLab-CRF [?]	83.5	36.6	82.5	62.3	66.5	85.4	78.5	83.7	30.4	72.9	60.4	78.5	75.5	82.1	79.7	58.2	82.0	48.8	73.7	63.3	**70.7**
Speech-Scribbles:																					
HFCN-1.5	72.1	32.1	63.2	47.0	59.7	72.1	70.4	72.8	21.6	58.2	41.6	68.2	58.2	71.3	71.5	41.4	56.8	32.1	64.5	52.1	56.4
HFCN-3	76.1	37.4	69.3	53.5	64.9	79.6	74.4	76.4	25.9	62.5	45.8	72.3	62.4	76.8	74.6	45.7	72.1	38.3	68.9	56.2	61.7
HFCN-3+CRF	78.7	37.5	71.7	52.8	64.5	78.6	77.8	79.7	25.9	65.6	49.9	75.1	65.9	79.4	76.6	48.5	74.9	39.9	73.9	59.5	63.9

Table 5. Comparison to other methods with different levels of supervision on PASCAL VOC 2012 test.

5.2 Results

Scribble Augmentation. The scribble for the background is augmented in an ensemble sampling manner, which saves the annotation cost for the background; background is often large and scattered, so annotating it can be costly. The parameter T for the ensemble sampling is evaluated over 10 values: $[1, 2, ..., 10]$ for HFCN-3. The results is shown in Fig **??**. As the figure shows, the performance increases with T from the beginning and then starts stabilizing. The scribble augmentation from each single round has its own weakness and the combination of multiple rounds 'cancels out' the weaknesses because the weaknesses from different rounds are different due to the randomness in the sampling. This property has been widely explored in the filed of ensemble learning. We use $T = 10$ for all the following experiments.

Validation Result: We first evaluate the method on the validation set. Table **??** show the results, where the results of several other methods are also reported for comparison. For HFCN, we trained it with the two versions of annotations: scribbles from Anno-1.5 and Anno-3. They are referred hereafter by HFCN-1.5 and HFCN-3. Following the literature [**?**, **?**], we also added the refinement by CRF [**?**] to Anno-3, which is referred by HFCN-3+CRF. It generally true from the table that stronger (more expensive) supervision leads to better performance. However, it also shows that with the scribbles by Anno-1.5, HFCN is already able to yield quite decent results. If the training data is upgraded to the scribbles of Anno-3, the results are comparable to that of FCN-8s [**?**] and better than that of CNN-EM [**?**], though their training annotations are much more expensive to obtain (c.f. Table **??**). Our method also shows significantly better results than the methods [**?**, **?**] trained with comparable annotation cost. HFCN-3+CRF further improves the

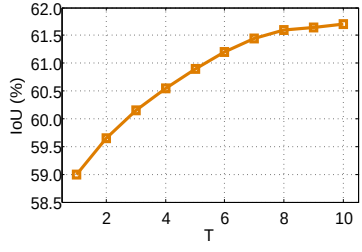


Fig. 5. The performance of HFCN-3 as a function of T.

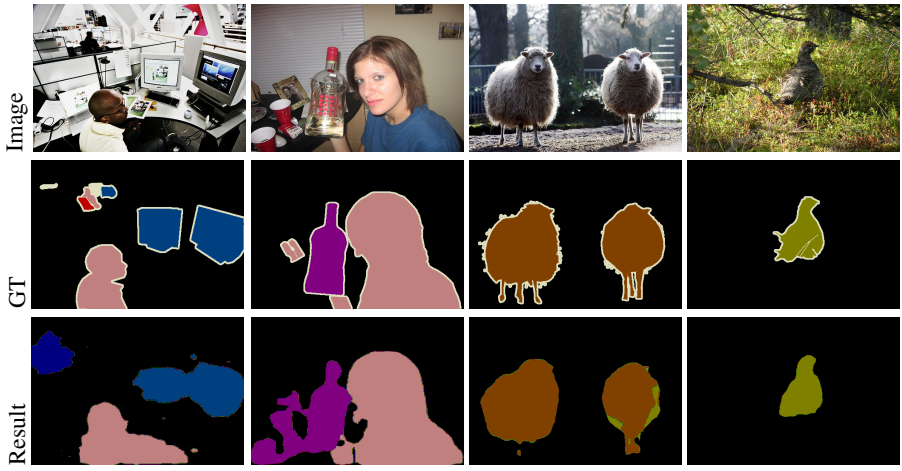


Fig. 6. Examples of the segmentation results on the validation set of PASCAL VOC 2012.

performance on top of HFCN-3, showing the benefits of combining graphical model and deep neural networks. Fig. **??** shows several segmentation examples.

Test Results: We also evaluate HFCN on the test set. Table **??** shows all the results. The conclusions drawn on the validation set hold on the test set as well. Our method obtains results which are comparable to other methods trained with more expensive supervision, i.e. full masks and bounding boxes, and are better than methods trained with supervision of comparable annotation cost. These annotation costs are only computed on the PASCAL VOC training images, however, some competing methods also use 'extra' supervision. For instance, Img2Pix-CNN [**?**] learns image prior from another larger set of images; BoxSup [**?**] uses the MCG object proposal [**?**] method, which is trained with full-mask supervision from the PASCAL VOC dataset. Also, the bounding boxes used by the two methods [**?**, **?**] are generated from the labeled full segmentation masks, which may transfer some supervision from there because the generated bounding boxes are often tighter than those annotated directly by annotators. The performance of HFCN is improved the same way as other methods [**?**, **?**, **?**] by the CRF [**?**] for further refinement.

Annotation Cost vs. Accuracy: We tabulate the annotation cost and the mIoU of all the methods considered. See Fig. **??** for the results. From the plot, it is evident that HFCN strikes a good balance between annotation speed and prediction accuracy. The merit makes our method adaptable and applicable to new, customized tasks, which is beneficial to many real vision applications. As more and more forms of supervision are explored, we believe that an evaluation of annotation cost vs. accuracy is very necessary in order to clearly show the trade-offs made by different alternative approaches.

Weak Annotation vs. Strong Annotation. The development of methods with different forms of annotations naturally raises a question: under a fixed annotation budget, should one work for more weak annotations or fewer precise annotations? We answer this question by comparing the performance of HFCN-3 trained on 9900 images annotated by Anno-3 to that of FCN-8 trained on 900 images with full-mask annotations.

The training data for FCN-8 is randomly sampled from all the training data. 5 FCN-8 models are trained, each with its own training set, and their results are averaged. The final results (mIoU) are: 56.3% for FCN-8 and 60.9% for HFCN-3. The results are exciting, and they suggest that with the same amount of annotation effort, our method gathers more semantic information than the conventional more expensive annotation methods. In a wider context, the results suggest that gathering weak training data can be more helpful than gathering strong training data, if the same amount of annotation effort is given. Learning with a mixture of strong and weak annotations can be interesting as well, as shown in [**?**]. We leave this as our future work.

Soft Heatmap vs. Hard Segmentation. We investigated whether the soft heatmaps are helpful. To this aim, a baseline is designed for comparison by simply taking the max object probability from the soft maps, and then passing the resulting object masks to FCN-8. We found that by doing so, the performance drops from 61.7 to 59.2. The advantage of soft heatmaps is that objects (parts) with uncertainties are left to be decided and explored by the FCN model that has been trained with objects of higher certainty. The trained FCN model is more knowledgeable than simple thresholding in distinguishing uncertain objects.

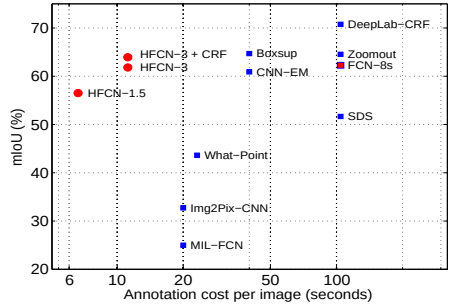


Fig. 7. Annotation vs. segmentation performance (mIoU) of all the methods considered. Evaluated on PASCAL VOC 2012 test.

6 Conclusion

In this paper, we developed an annotation method Draw&Tell to create training data for semantic image segmentation. Draw&Tell allows annotators to simply draw scribbles (strokes) on objects and speak their names in the meanwhile, solving the *what* and *where* problems once at the same time. We have proven experimentally that Draw&Tell is faster than other annotation methods, e.g. 11 times faster than full-mask annotation, 4 times faster than bounding-box annotations, and 2 times than the image-level annotation. A method of integrating visual information and acoustic information is also proposed for robust object name recognition. This combination can serve as an example of integrating vision and speech, and inspires research in this direction. Furthermore, we proposed a method that allows CNNs models to learn from scribble-based training data, by converting scribbles to semantic confidence maps and extending standard CNNs to accommodate soft confidence maps. We showed in experiments that our annotation method, coupled with the learning method, yields significantly better results than competing methods trained with annotations of comparable cost, and yields comparable results to the methods trained with significantly more expensive annotations. Introducing speech recognition to visual annotation is helpful especially for tasks on mobile devices. This work is just a very first step in this direction.