Improving Few-Shot Part Segmentation using Coarse Supervision

Oindrila Saha Zezhou Cheng Subhransu Maji
{osaha, zezhoucheng, smaji}@cs.umass.edu

University of Massachusetts, Amherst

Abstract. A significant bottleneck in training deep networks for part segmentation is the cost of obtaining detailed annotations. We propose a framework to exploit coarse labels such as figure-ground masks and keypoint locations that are readily available for some categories to improve part segmentation models. A key challenge is that these annotations were collected for different tasks and with different labeling styles and cannot be readily mapped to the part labels. To this end, we propose to jointly learn the dependencies between labeling styles and the part segmentation model, allowing us to utilize supervision from diverse labels. To evaluate our approach we develop a benchmark on the Caltech-UCSD birds and OID Aircraft dataset. Our approach outperforms baselines based on multi-task learning, semi-supervised learning, and competitive methods relying on loss functions manually designed to exploit coarse supervision.

Keywords: Part segmentation, few-shot learning, semi-supervised learning.

1 Introduction

Accurate models for labeling parts of an object can aid fine-grained recognition tasks such as estimating the shape and size of animals, and support applications in graphics such as image editing and animation. But a significant bottleneck is the cost of collecting annotations for supervising part labeling models. In many situations however, one can find datasets with alternate labels such as object bounding boxes, figure-ground masks, or keypoints, which may serve as a source of supervision. However the variations in their level of detail and structure, e.g., bounding boxes and masks are coarser than part labels while keypoints are sparse, implies that they cannot be readily "translated" to part labels to directly supervise learning.

We propose a framework to learn part segmentation models using existing datasets with coarse labels such as figure-ground masks and keypoints. The approach illustrated in Fig. 1 treats part labels as latent variables and jointly learns the part segmentation model and the *unknown* dependencies between the labeling styles in a Bayesian setting (§ 3). The dependencies are represented using deep networks to model complex relationships between labeling styles, allowing supervision from a variety of coarse labels. One technical challenge is that inference requires sampling over high-dimensional latent distributions

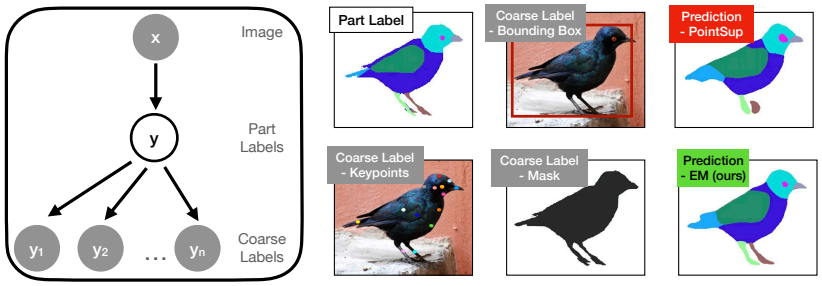

Fig. 1: Overview of our approach. The graphical model over an image x, parts labels y and coarse labels $y_1, \ldots, y_n$, is shown on the left. Coarse labels such as bounding boxes, figure-ground masks, or keypoint locations are easier to annotate than per-pixel part labels, and our learning framework can utilize datasets with coarse labels to train part segmentation models, outperforming previous work - PointSup (Fig. 7).

which is typically intractable. We address this by making certain conditional independence assumptions and develop an amortized inference procedure for learning. Our method allows the use off-the-shelf image segmentation networks and standard back-propagation machinery for training.

To evaluate our approach we design a benchmark for labeling parts on the Caltech-UCSD birds (CUB) [32] and OID Aircraft [31] dataset (§ 4). We utilize the keypoint and masks from the CUB dataset and the part-segmentation labels of the birds of PASCAL VOC to segment birds into 10 parts. Our approach achieves a performance of 49.25% mIoU compared to baseline of fine-tuning an ImageNet pre-trained network on all the available part labels (45.37% mIoU), as well as multi-tasking (41.27% mIoU) and semi-supervised learning baseline (46.01% mIoU). It also outperforms PointSup [4] (46.76% mIoU), an approach for training using point-supervision – for this approach we manually assign keypoints to parts and combine it with the figure-ground mask to provide a set of part segmentation labels. On the OID Aircraft dataset we observed a similar trend, where our approach (58.68% mIoU) outperforms the fine-tuning (55.3% mIoU) and multi-tasking (55.61% mIoU) baselines. These experiments are consistent across different initializations of the network (e.g., ImageNet pre-trained vs. random), as well as the types and combinations of coarse labels (§ 5 & § 6).

Our approach is also relatively efficient — fine-tuning on all the labeled parts of the CUB dataset requires 1 hour, PointSup [2] requires 2.5 hours, while our approach requires 7.5 hours on a single NVIDIA RTX-8000 GPU. Importantly, our approach requires little additional labeling ($\approx$300 or $<$3% instances for CUB dataset), and benefits from *existing* part labels on PASCAL and coarse labels on CUB. These experiments suggest that diverse coarse labels across datasets can be used to effectively guide part labeling tasks within our framework.

To summarize our contributions include: 1) a framework to learning part segmentation models using diverse coarse supervision from existing datasets; 2) an amortized inference procedure that is efficient, roughly 3$\times$ slower than

the leading alternate methods for coarse supervision (e.g., PointSup [4]), and is more accurate; 3) two benchmarks for evaluating part-segmentation from a few-labeled examples on the CUB and OID Aircraft dataset; and 4) a systematic evaluation of various design choices including the role of initialization for transfer learning and the relative benefits of various forms of coarse labels. The source code and data associated with the paper are publicly available at https://people.cs.umass.edu/∼osaha/coarsesup

2 Related work

Weakly supervised image segmentation Previous work use supervision from classification labels, bounding boxes or at sparse locations in the image such as points or lines. Zhou *et al.* [37] use class response at every image location for a given class and train by mapping the response peaks to more informative parts of an object instance. Other approaches generate pseudo ground truth labels using previous image classification models [1,38]. Khoreva *et al.* [15] use bounding boxes as weak supervision. They generate pseudo ground truth using classical approaches such as GrabCut [23] inside given bounding box and use that to train the segmentation model. Hsu *et al.* [13] use a bounding box tightness prior and train a Mask-RCNN [10] using horizontal and vertical patches from the tight bounding box as positive signals and those outside as negative signals. BoxInst [27] uses a projection loss that forces horizontal and vertical lines inside bounding boxes to predict at least one foreground pixel and an affinity loss that forces pixels with similar colors to have the same label. Laradji *et al.* [17] introduce a proposal-based instance segmentation method that uses a single point per instance as supervision. Cheng *et al.* [4] uses multiple points randomly sampled per instance as well as bounding boxes as supervision to train a Mask-RCNN model. ScribbleSup [18] uses a graphical model that jointly propagates information from scribbles to unmarked pixels to learn network parameters. Another stream of work [3, 39] train two models simultaneously with cross supervision from one model to train the other. Naha *et al.* [20] use keypoint guidance to predict part segmentation labels for unseen classes but require keypoint inputs during evaluation time. All these methods design algorithms specific to one kind of supervision and the annotation style has a clear mapping to the desired part labels. *In contrast, our method handles a variety of label styles allowing opportunistic use of existing datasets to learn part segmentation labels.*

Unsupervised learning A number of previous work use self-supervision for learning segmentations. SCOPS [14] uses geometric concentration (areas of the same object part are spatially concentrated), equivariance (enforcing part segmentation to be aligned with geometric transformations) and semantic consistency (over different instances). Wang *et al.* [33] also use equivariance constraints to refine class activation maps which in turn form the final segmentation maps. Another method [21] uses pixel-level contrastive learning to learn feature representations for downstream tasks such as segmentation. Yang *et al.* [35] use a

layered GAN to produce background and foreground layers for an image where the discriminator predicts on the overlayed image. PiCIE [5] enforces invariance to photometric transformations and equivariance to geometric transformations for different views of the same image. A number of recent techniques based on generative [29, 36] and contrastive learning [19] approaches have also been proposed (See [25] for a systematic evaluation). These methods can be used to initialize networks to boost performance in few-shot learning and are complementary to our approach. *For example, we compare the benefits of self-supervised learning over randomly initialized networks and ImageNet pre-trained networks.*

Multi-task learning benefits from diverse source of supervision by sharing parts of the model across tasks. For image segmentation, a prior work [6] proposes multi-task cascaded networks where three networks predict instances, masks and categorize objects respectively. Heuer *et al.* [12] combine the tasks of object detection, semantic segmentation and human pose estimation but fails to perform better than the single task network for segmentation — a trend we also observe in our experiments. Standley *et al.* [26] show that combining some tasks in multi-task setting can degrade performance while for other cases performance can get boosted. To design a multi-task network able to handle different tasks, some methods [8, 9] group tasks that would perform well together. Other works such as [16, 28] use keypoints and bounding box information to predict instance segmentation but use a multistage framework. Mask-RCNN [10] adds a mask segmentation head to Faster-RCNN [22] to predict bounding box and instance segmentation. *Unlike generic multi-tasking approaches, our approach exploits the hierarchical label structure to guide learning and consistently outperforms them.*

3 A Joint Model of Labeling Styles

For an image x denote $y \in S$ the part segmentation label, i.e., pixel-wise label for each part, and $y_1 \in S_1$, $y_2 \in S_2$, $\ldots$, $y_n \in S_n$ denote coarse labels corresponding to various labeling styles. For example, y_1 might denote the coordinates of a set of keypoints and y_2 might denote the figure-ground mask. We call a labeling S_a coarser than S_b if S_a can be derived from S_b *independent* of the image x. For example, the figure-ground mask can be derived from the part label of an object, or the bounding-box can be derived from the figure-ground mask. *Our goal is to learn a part segmentation model $p(y|x)$ given a small set of images with part labels $y \in S$, and a large set of images with coarse labels $y_k \in S_k$.*

This assumption that the coarse labels can be derived from the part labels leads to the following joint probability distribution over the image and the labels:

$$p(y, y_1, \ldots, y_n|x) = p(y|x) \prod_{i=1}^{n} p(y_1|y),$$

and is illustrated by the graphical model in Fig. 1. The assumption might appear to be strong, but we find that it holds for the styles of labels we consider.

For example, a convolutional network can accurately predict the location of keypoints given the part segmentation labels with > 92 PCK which is as good as the accuracy of keypoints given image. However, the form of $p(y_k|y)$ is complex in this case as it involves reasoning about the extent and location of various parts. The distribution might also be unknown, especially when combining existing datasets which may have been collected with a different set of labels and annotation guidelines. For example, there might not be a direct correspondence between the names of parts used for keypoint annotations and those for segmentation task. In contrast, the form is simple and deterministic for figure-ground masks or bounding boxes given part labels. We incorporate this factorization in a Bayesian setup to learn both the part segmentation model and the dependencies between the labeling styles described next.

3.1 Variational EM for Learning

Assume that an image x contains coarse labels $y_1, y_2, \ldots, y_n$. We will estimate parameters θ to maximize the log-likelihood of the data:

$$\max_{\theta} \mathcal{L}(\theta) = \log p(y_1, y_2, \ldots, y_n|x, \theta). \tag{1}$$

Given a distribution $q(y)$ over the latent variables[1] the $\mathcal{L}(\theta)$ can be bounded as:

$$\begin{aligned}
\mathcal{L}(\theta) &= \log \sum_y p(y, y_1, y_2, \ldots, y_n|x, \theta) \\
&= \log \sum_y q(y) \frac{p(y, y_1, y_2, \ldots, y_n|x, \theta)}{q(y)} \\
&\geq \sum_y q(y) \log \frac{p(y, y_1, y_2, \ldots, y_n|x, \theta)}{q(y)} \\
&= \sum_y q(y) \log p(y, y_1, y_2, \ldots, y_n|x, \theta) + H(q) \\
&= \sum_y q(y) \left[\log p(y|x) \prod_{i=1}^{n} p(y_i|y, \theta) \right] + H(q) := \mathcal{F}(q, \theta).
\end{aligned} \tag{2}$$

Where $H(q) = -\sum_y q(y) \log q(y)$ is the entropy of the distribution q. The EM algorithm alternates between:

- **E step:** maximize $\mathcal{F}(q, \theta)$ wrt distribution over y given the parameters:

$$q^{(k)}(y) = \arg\max_{q(y)} \mathcal{F}(q(y), \theta^{(k-1)}).$$

- **M step:** maximize $\mathcal{F}(q, \theta)$ wrt parameters given the distribution $q(y)$:

$$\theta^{(k)} = \arg\max_{\theta} \mathcal{F}(q^{(k)}(y), \theta) = \arg\max_{\theta} \sum_y q^{(k)}(y) \log p(y, y_1, y_2, \ldots y_n|x, \theta)$$

[1] $q(y) \geq 0$ and $p(y, y_1, y_2, \ldots, y_n) > 0 \Rightarrow q(y) > 0$

Note that in the above we have derived the EM algorithm for a single example x, but the overall approach requires estimating the distribution over latent variables $q(y)$ for each training example and parameters across all the training examples. However, optimizing $q(y)$ for each training sample x is typically intractable for high-dimensional distributions like ours. In "Hard EM" the distribution $q(y)$ is replaced by the mode of the posterior distribution but estimating this can also be challenging when the probabilities are expressed using deep networks. In the next section we present an amortized inference procedure where we estimate $q(y)$ using a separate network conditioned on all the observed variables.

3.2 Coarse Supervision from Keypoints and Figure-Ground Mask

As a concrete example consider that two types of coarse labeling styles are available – $y_{\text{mask}} \in S_{\text{mask}}$ denoting the figure-ground mask of the same size as the image, and $y_{\text{kp}} \in S_{\text{kp}}$ denoting the locations of a set of keypoints in an image. To make inference tractable we adopt a Laplace approximation and model the conditional distributions as a random variable centered around a mean as follows:

- $p(y|x) \propto \exp(-\alpha|y - \mu(x)|)$ where $\mu(x)$ is the mean distribution of the part labels for the image estimated using a deep network with parameters θ.
- $p(y_{\text{kp}}|y) \propto \exp(-\lambda|y_{kp} - \mu_{\text{kp}}(y)|)$ where $\mu_{\text{kp}}(y)$ is the mean location of the keypoints estimated using a deep network with parameters θ_{kp} that takes part labels as input and predicts the locations of keypoints.
- $p(y_{\text{mask}}|y) = B(y_{\text{mask}}, \mu_{\text{mask}}(y))$ a Binomial distribution where $\mu_{\text{mask}}(y)$ is obtained by summing over the parts probabilities at each pixel. This function has no learnable parameters.

In the E Step we optimize $q(y)$ for each training example x as:

$$\arg\max_{q(y)} \sum_{y} q(y) \left[\log p(y|x) \prod_{i=1}^{n} p(y_i|y) \right] + H(q). \tag{3}$$

Given the form of the probability distributions this corresponds to maximizing $q(y)$ given $\mu(x), y_{kp}$ and y_{mask} (ignoring the entropy term):

$$\sum_{y} q(y) \exp\big(-\alpha\big|y - \mu(x)\big|\big) \exp\big(-\big|y_{kp} - \mu_{\text{kp}}(y)\big|\big) B\big(y_{\text{mask}}, \mu_{\text{mask}}(y)\big). \tag{4}$$

For hard EM, it is possible to solve for the optimal y using gradient ascent as each of these functions $\mu, \mu_{\text{kp}}(y)$ and $\mu_{\text{mask}}(y)$ are differentiable wrt y. Similarly, one can construct a sample estimate for $q(y)$ using gradient-based techniques such as SGLD [34]. However, both these choices require many gradient iterations and can get stuck in local minima as y is very high-dimensional. Thus, instead of optimizing for each example x individually we approximate the mode with another distribution $q_x(y) \approx q(y|x, y, y_{kp}, y_{mask}, \theta_q)$ parameterized using a deep network with parameters θ_q shared across all training examples. The network takes as input the image and coarse labels and predicts the part labels. In the

E step we optimize θ_q using gradient descent over *all* examples allowing us to amortize the inference cost across examples.

In the M step, for each unlabeled image x we sample labels y using the variational distribution $q(y|x, y, y_{kp}, y_{mask}, \theta_q)$ and update the parameters θ and θ_{kp} of the model for predicting $p(y|x)$ and $p(y_{kp}|y)$ respectively. In practice we sample the mode of each input x predicted by the feed-forward network. This is simple and has worked well for our experiments, though techniques for sampling from deep networks might lead to better estimates. The entire algorithm is outlined in Alg. 1. Here ℓ, ℓ_{kp} and ℓ_{mask} correspond to the loss functions for the part labels, keypoints, mask obtained as the negative log-likelihood of the corresponding probability functions in Eqn. 2 and ℓ_q is the negative entropy.

Remarks. (1) In the above derivation we assumed all the images have the same set of coarse labels. But the method can be generalized to handle images with different number of coarse labels as the log-likelihood (Eqn. 2) decomposes over the labels. However, the model for estimating the variational distribution $q(y)$ should be adapted to condition on the provided labels for the image. One possibility is to train separate models, e.g., $q(y|x, y_{mask})$ and $q(y|x, y_{kp}, y_{mask})$, or treat the missing labels as latent variables and infer them during training. (2) The method can handle different styles of coarse supervision by simply adding $p(y_k|y)$ for the corresponding label style. For example, supervision from object bounding-boxes can be incorporated by treating the box as two keypoints corresponding to the top-left and bottom-right corners or as a mask. Similarly, box-level annotations for the parts can also be used as coarse supervision.

4 Benchmarks for Evaluation

In this section we describe the datasets used for our experiments. Fig. 2 shows the PASCUB dataset for bird part segmentation. The top row is examples we annotated from the CUB dataset and bottom row are examples from the PASCAL parts dataset for the birds category after removing low-resolution and truncated instances (Appendix A). Fig. 3 shows examples from the OID Aircraft dataset. Below we describe the details and evaluation metrics of both.

4.1 Bird part segmentation benchmark

Our goal is to segment each bird into 10 parts: 'beak', 'head', 'left eye', 'left leg', 'left wing', 'right eye', 'right leg', 'right wing', 'tail' and 'torso'. The bird category in the PASCAL parts dataset contains several labeled examples, but most instances are small and truncated as the dataset is primarily designed for object detection. On the other hand, the CUB dataset has higher resolution instances and includes keypoint and figure-ground masks but does not contain part labels. So, we combine the two and provide part labels for a few instances on the CUB dataset to create a benchmark for few-shot part segmentation.

Algorithm 1 Stochastic Variational EM for Part Segmentation

Input: $\mathcal{D}^p := \{(x^p, y^p, y^p_{\text{mask}}, y^p_{\text{kp}})\}$ ▷ Dataset with part labels
Input: $\mathcal{D} := \{(x, y_{\text{mask}}, y_{\text{kp}})\}$ ▷ Dataset with coarse labels
Input: params $= \{\#epochs, b^p, b, \alpha, \lambda_1, \lambda_2, \delta_1, \delta_2\}$
function TrainPartSegSGDVarEM($\mathcal{D}^p, \mathcal{D}$, params)
  Initialize $f(y|x,\theta)$, $f(y_{\text{mask}}|y)$, $f(y_{\text{kp}}|y,\theta_{\text{kp}})$ and $f(y|x, y_{\text{mask}}, y_{\text{kp}}, \theta_{\text{q}})$
  for *epoch* $\leftarrow 1$ to #epochs **do**
    $[x^p, y^p, y^p_{\text{mask}}, y^p_{\text{kp}}] =$ next-batch$(\mathcal{D}^p, b^p)$
    $[x, y_{\text{mask}}, y_{\text{kp}}] =$ next-batch$(\mathcal{D}, b)$
    #E Step
    $\mu_q = f(y|x, y_{\text{mask}}, y_{\text{kp}}, \theta_{\text{q}})$ ▷ Variational distribution
    $\mu = f(y|x,\theta)$ ▷ Part segmentation model
    $\mu_{\text{kp}} = f(y_{\text{kp}}|\mu_q, \theta_{\text{kp}})$. ▷ Keypoint model
    $\mu_{\text{mask}} = f(y_{\text{mask}}|\mu_q)$ ▷ Mask model
    $L = \alpha\ell(\mu_q, \mu) + \lambda_1\ell_{\text{kp}}(y_{\text{kp}}, \mu_{\text{kp}}) + \lambda_2\ell_{\text{mask}}(y_{\text{mask}}, \mu_{\text{mask}}) + \ell_q(\mu_q)$
    gradient-update(L, θ_{q})
    #M Step
    $\mu_q = f(y|x, y_{\text{mask}}, y_{\text{kp}}, \theta_{\text{q}})$ ▷ Sample labels
    $\mu^p = f(y|x^p, \theta)$ ▷ Part segmentation model
    $\mu_{\text{kp}} = f(y_{\text{kp}}|\mu_q, \theta_{\text{kp}})$ ▷ Keypoint model
    gradient-update$(\delta_1\ell(y^p, \mu^p) + \delta_2\ell(\mu_q, \mu), \theta)$
    gradient-update$(\ell_{\text{kp}}(y_{\text{kp}}, \mu_{\text{kp}}), \theta_{\text{kp}})$
  end for
end function

Part Segmentation Data

Split	PASCAL	CUB
#Train	271	150
#Val	132	74
#Test	133	75

Coarsely Labelled Data

#Train	5994
#Val	2897
#Test	2897

Table 1: Data splits for PASCUB.

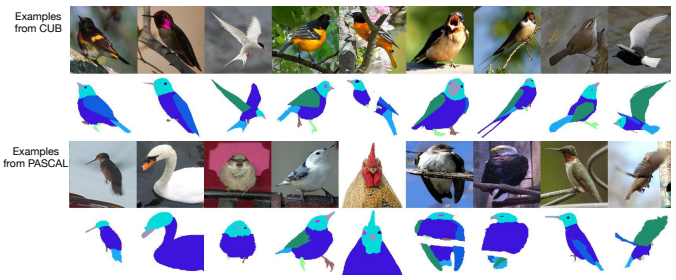

Fig. 2: Example images from the PASCUB dataset.

CUB The Caltech-UCSD birds dataset [32] has 11,788 images centered on individual birds across 200 species. We annotate 299 randomly chosen images with pixel-wise part labels (referred to as CUB Part) for the 10 classes mentioned above. We divide the 299 images we annotated into train, val and test in a 2:1:1 split (Tab. 1). The CUB dataset also includes keypoints and figure-ground masks for all images. We use the full data divided into the official splits as our coarsely labelled data for PASCUB experiments (Tab. 1).

PASCAL The PASCAL VOC [7] dataset has 625 images that contain at least one bird. Chen *et al.* [2] provide part segmentations where each bird has pixel-

Fig. 3: Part labels on the OID Aircraft dataset.

wise part labels for 13 classes – we group classes such as 'neck' and 'head' to a single category 'head' resulting in the 11 classes listed above. We also removed instances that are truncated and are of low resolution to make for a cleaner training and evaluation set — the pre-processing is detailed in Appendix A. Now we are left with 536 centered bird images. Using the official split of PASCAL VOC results in a training set of 271 images. One image can contain more than one bird in PASCAL. Crops originating from an image from train split go in the train split of our dataset. Since the official split does not have val/test demarcation we randomly divide the rest of the images into validation and test sets equally (Tab. 1). Fig. 2 shows the data after pre-processing.

The overall dataset contains 570 instances with part labels, and roughly 12k instances with keypoint and mask labels, divided into training, validation, and test sets. Annotating part segmentations requires roughly 5-10× more effort than masks based on our own experience, and this benchmark contains such labels for less than 5% of the objects.

4.2 Aircraft part segmentation benchmark

The OID Aircraft dataset [31] has 7543 images. Each image has an associated figure ground mask and part labels for four parts — 'nose', 'wings', 'wheels' and 'vertical stabilizer'. The figure ground masks provided are quite accurate, but the part labels are noisy. Thus, we manually select 300 images for which the part labels are visually correct. In keeping with the splits of the datasets described above, we divide these 300 images into 150 for training, 75 each for validation and testing. We refer to this subset as OID Part. For the rest of the dataset, we use the official train/val/test split. Note that the part labels in this case do not collectively form the figure ground masks. Each pixel of the image also can have more than one part label marked. Thus, we handle the segmentation training in a different way for Aircrafts as described in the section below.

5 Part Segmentation Algorithms

For all the baselines and for our approach, we use an encoder-decoder based fully convolutional network. We present details of all architectures in Appendix B. We use colour jittering and flipping augmentations for training all models. We resize images and corresponding labels to 256 × 256. The hyperparameters for each approach were chosen on the validation set of each benchmark.

5.1 Baselines

In this section, we describe training and design details for all baselines that we compare our method with.

Fine-tuning We start with a network pre-trained on another task and fine-tune it for part segmentation. We replace the final fully connected layer to predict part labels and train the network using a cross entropy loss for PASCUB experiments. For Aircrafts, we treat segmentation as a pixel-wise multi-label classification task and use binary cross entropy (BCE) on each channel. We train this using Adam optimizer with a learning rate of 0.0001 for 200 epochs.

Semi-supervised learning We use the method described in PseudoSup [3] as a semi-supervised learning baseline. The method uses an ensemble of two networks obtained by fine-tuning starting from two different initializations. Note that for the 'Random' case (see Tab. 2), both networks start with different random init before fine-tuning, while for 'Keypoint'/'ImageNet' cases only the last layer/decoder has different random inits. After obtaining the two different fine-tuning checkpoints, PseudoSup training uses one ensemble to train the other and vice versa using pseudo-labels on all coarsely labelled images. Pseudo-labels refers to converting the predictions to one-hot labels by computing the argmax over all channels. We also add the fully-supervised loss from images with part labels. We use SGD optimizer with learning rate of 1E-4, momentum of 0.9 and weight decay of 1E-4 for both networks. We train for 90 epochs with cosine learning rate scheduling.

Multi-task learning Here we train a single model to accomplish both the tasks of keypoint prediction and part segmentation. We use a common encoder based on a ResNet-34 and attach decoders for each task described below.

- For PASCUB, the first decoder is for part segmentation labels where we use cross entropy loss over the prediction and ground truth labels. The second decoder predicts keypoints where we use pixel-wise ℓ_1-loss over the predicted and ground-truth keypoints which are represented as Gaussians around each keypoint. The output of the first decoder also receives supervision from the figure-ground masks by summing over channel dimension for the foreground classes of the prediction and taking a cross-entropy loss. The sum of all these losses is backpropagated through the encoder during training. The weights for the figure-ground loss and part segmentation loss are set to be 1 and that of the keypoint loss is set to be 10. We use SGD optimizer with learning rate of 0.1, momentum of 0.9 and weight decay of 0.0001 for both networks. We train for 90 epochs with cosine learning rate scheduling.
- For Aircrafts, the first decoder performs part segmentation and is trained with binary cross-entropy loss for on each channel as parts are not mutually exclusive. The second decoder predicts the figure ground mask and is trained with cross-entropy loss. The weightages for losses from both decoders are set to be 1. Using an initial learning rate of 0.2, the rest of the training procedure remains the same as above.

Handcrafted loss functions We base this method on PointSup [4] — a method to train segmentation models using point supervision. We evaluate this on PASCUB since it has keypoint annotations. The procedure is illustrated in Fig. 7 in the Appendix. We first assign keypoints to each part manually based on their co-occurrence, e.g. the 'head', 'crown' and 'throat' keypoints are assigned the 'head' part. We then dilate these locations using a 5×5 pixel window — we choose 5×5 so as to not exceed the area of the smallest part, the eyes. We then train the network with a pixel-wise cross entropy loss computed on all these annotated points and the corresponding figure-ground mask. This is mask-loss is computed across all pixels by summing over the foreground channels and using a cross entropy loss. For the loss over part labels points we set the weightage as 0.5, for loss from figure-ground mask as 1 and for that from part label we set weightage to 2. We use SGD optimizer with learning rate of 0.001, momentum of 0.9 and weight decay of 0.0001. We train for 90 epochs with cosine lr scheduling.

5.2 Details for our approach

In this section we specify how we initialize each model of the EM algorithm before training and describe the training details of the EM method.

Part segmentation model: $f(y|x)$. We initialize this using a checkpoint obtained by fine-tuning, i.e., a model trained using the provided part labels.

Posterior inference model: $f(y|x, y_{\text{mask}}, y_{\text{kp}})$. We use a split encoders for this model (Fig. 5 in supp.). The first is a ResNet34 pretrained on ImageNet [24] to extract features from the image and second a shallow ResNet-based encoder to process the masks and keypoint heatmaps concatenated in channel dimension. We concatenate the features of the encoders and use a common decoder to create y. For PASCUB, we train using images from CUB for which we have both labelled part segmentations and keypoint annotations. For Aircrafts, similarly we use those images which have both figure ground masks and clean part labels. We provide details on architecture in Appendix B. We use flipping and color jitter augmentations while training. We use a learning rate of 0.1 for the whole network except the image encoder branch for which we set learning rate to 0.01. We use cosine learning rate scheduling and train for 90 epochs. We use SGD optimizer with momentum of 0.9 and weight decay of 1e-4.

Keypoint model: $f(y_{\text{kp}}|y)$. This refers to the model for predicting keypoints given part labels. On PASCUB using the checkpoint from the finetuned $p(y|x)$ model we generate part segmentations for all CUB images. We use y_{kp} from ground truth and generated y from $f(y|x)$ for an initial training. We then fine-tune the model on the images that have both ground truth y and y_{kp}. For this stage we use color jitter and flipping augmentations. For the initial training we use a learning rate of 0.1 with cosine lr scheduling and train for 90 epochs. For fine-tuning, we use a learning rate of 0.001 and train for 10 epochs. We use SGD optimizer with momentum of 0.9 and weight decay of 1e-4 for both. This model

achieves a PCK@10% of **92.85** on the CUB test set which is higher than that obtained by training using image inputs (92.65) for the same architecture.

Mask model: $f(y_{\text{mask}}|y)$. This is the model for predicting figure-ground masks from part labels. For PASCUB, we can predict the mask directly by marginalizing (summing) over the all the part labels. For Aircrafts, we need to use a model to predict $f(y_{\text{mask}}|y)$ since the part labels do not cover the whole mask and are not mutually exclusive. We use a model similar to the $f(y_{\text{kp}}|y)$ for PASCUB and follow the same initialization procedure.

EM Training As described in Alg. 1, the EM training proceeds by updating the posterior model $p(y|x, y_{\text{kp}}, y_{\text{mask}})$ (E Step), followed by updating the part $p(y|x)$ and keypoint $p(y_{\text{kp}}|y)$ models (M Step) over batches of training data. For PASCUB with keypoint and ImageNet initialization, we use learning rate for part model as 1e-5, that of posterior model as 1e-3 and coarse supervision model $p(y_k|y)$ as 1e-8. For random initialization, we set learning rates as 5e-4, 1e-5 and 1e-8 respectively. We use SGD optimizer with momentum of 0.9 and weight decay of 1e-4. We train the model for 40 epochs and choose the best checkpoint based on lowest cross entropy loss of $p(y|x)$ on validation set of PASCUB dataset. We use batch size of 32 for the coarse labelled dataset and a batch size of 4 for the part labelled dataset. We detail rest of the hyperparameters in Appendix C. We follow a very similar procedure for the Aircrafts. The loss for the E step comes from the predicted labels and the figure-ground mask, while for the M step we train the part model $p(y|x)$ using posterior mode. We use learning rate for part model as 0.005, that of posterior model as 1e-6 and coarse supervision model $p(y_m|y)$ as 1e-8. For Aircrafts we perform experiments for ImageNet init and share details of all hyperparameters in Appendix C. Fig. 4 shows the progress segmentation models using EM over epochs on an image.

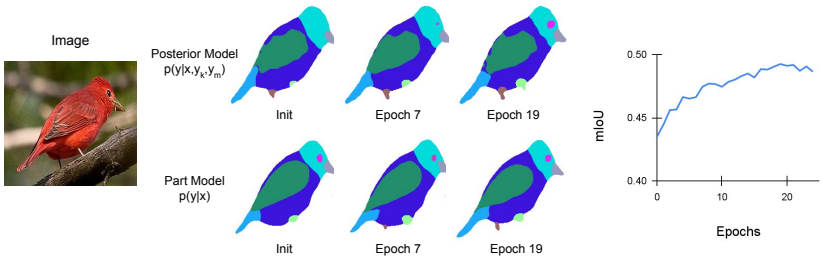

Fig. 4: EM training. The top row shows the output of the posterior model $p(y|x, y_{\text{mask}}, y_{\text{kp}})$ and the bottom shows the output of the part model $p(y|x)$ for the image at various epochs. Both models improve and influence each other – the posterior model learns to recognise the eyes while the part model learns to segment the feet. The right shows that the validation mIoU increases over epochs.

6 Results

Below we summarize the key conclusions of our experiments.

Our approach outperforms alternatives. Tab. 2 compares our approach to baselines on the PASCUB dataset for various network initializations. Performance is reported as the mean intersection-over-union (mIoU) across parts. Note that the benchmark has train/val sets for CUB and PASCAL. Tab. 2 shows the results on CUB, while those for PASCAL are included in the supp. Our approach handily outperforms the fine-tuning baseline — with the largest gains when the network is randomly initialized. We also outerform PointSup, a strong baseline based on handcrafted labels obtained from keypoints. Designing handcrafted labels might be challenging if keypoints are densely labeled, or if the annotation style varies. In comparison, our approach does not assume prior knowledge on the style of labels and learns them as part of training. Tab. 4 shows the same results for the OID Part dataset, where our approach outperforms fine-tuning and multi-tasking baseline. For this dataset we use ImageNet initialization. PointSup is not applicable as keypoints are not annotated on this dataset.

Multi-tasking is rarely effective. The simple strategy of multi-tasking was effective only when the models are trained from scratch. Despite careful hyperparameter search, we found that the overall performance degrades when better initializations are used. A staged strategy, where the network is trained to predict keypoints on the whole CUB dataset and then fine-tuned to predict part labels was more effective (Tab. 2 Keypoint init. + Fine-tuning outperforms Multi-task).

Semi-supervised learning provides minor benefits. The semi-supervised learning approach based on PseusoSup provides relatively small (0.5-1% MIoU) improvement over the fine-tuning baseline.

Our approach benefits from various coarse labels. Table 3 shows the results on the CUB test set using various forms of coarse supervision. A model trained using mask supervision only obtain 46.30% mIoU, one with Keypoint only obtains 47.96% mIoU, while using both Keypoints and masks obtains 49.25% mIoU. All these models are better than the fine-tuning baseline (45.37% mIoU) and the semi-supervised learning baseline (46.01% mIoU).

Our approach is relatively efficient. The key benefit of our approach is that it is relatively efficient. First, we were able to utilize existing labels on PASCAL and CUB dataset to train the part segmentation model. Our model required labeling 300 part labels on CUB, half of which were used for evaluation. Considering that it takes on the order of a minute or two to label parts for each instance, the ability to train part-segmentation models using existing coarse labels is a compelling alternative to labeling large datasets of parts. Second, the overall training for our approach (7.5 hr) is also a small factor increase over fine-tuning (1 hr), semi-supervised learning (6.2 hr), multitasking (4 hr) and PseudoSup (2.5 hr) on a single NVIDIA RTX8000 GPU.

Table 2: Performance on Birds. Comparison of the EM method with baselines described in § 5.1 on the testing and validation set of CUB parts. Our method (in green) outperforms baselines for all initializations. We present results on PASCAL val/test splits in Appendix D. The std-deviation over runs for Fine-tuning, MultiTask, PseudoSup and EM is $< \pm 1$ mIoU. For PointSup the std-deviation is $\sim \pm 2$ mIoU.

Method	CUB Part Test			CUB Part Val		
	Random	Keypoint	ImageNet	Random	Keypoint	ImageNet
Fine-tuning	29.88	41.12	45.37	35.28	44.64	48.62
MultiTask	36.96	38.00	41.27	40.24	41.74	43.93
PseudoSup [3]	30.77	41.62	46.01	36.32	45.03	48.67
PointSup [4]	35.18	46.45	46.76	38.05	48.01	48.84
Ours	37.98	49.25	48.05	40.85	52.19	51.11

Table 3: Effect of coarse supervision. The mIoU on the CUB test using various coarse labels.

EM Supervision	mIOU
Keypoint + Mask	49.25
Mask only	46.30
Keypoint only	47.96

Table 4: Performance on OID. Our method (in green) outperforms baselines based on multi-tasking and fine-tuning.

Method	OID val	OID test
Fine-tune	54.17	55.30
MultiTask	55.94	55.61
Ours	57.46	58.68

7 Conclusions

We present a framework for learning part segmentation models using a few part labels by exploiting existing coarsely labelled datasets. Our approach jointly learns the dependencies between labeling styles allowing supervision from diverse labels. This allowed us to train a bird part segmentation model by combining the part labels on PASCAL VOC with figure-ground mask and keypoint labels on CUB dataset. The model outperforms baselines based on fine-tuning, semi-supervised learning, multi-tasking, as well as learning with handcrafted labels and loss functions. We also presented results on the Aircraft dataset where we improve over the baselines. Our framework can handle multiple types of annotations (e.g., boxes, keypoints, masks, etc.) providing a way to combine existing labels across datasets without requiring manual translation across styles. For example, we could combine annotations from the NABirds dataset [30] which contains keypoints and object bounding-box to improve results.

Acknowledgements. The research is supported in part by NSF grants # 1749833 and #1908669. Our experiments were performed on the University of Massachusetts GPU cluster funded by the Mass. Technology Collaborative.

References

1. Ahn, J., Cho, S., Kwak, S.: Weakly supervised learning of instance segmentation with inter-pixel relations. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2209–2218 (2019) 3
2. Chen, X., Mottaghi, R., Liu, X., Fidler, S., Urtasun, R., Yuille, A.: Detect what you can: Detecting and representing objects using holistic models and body parts. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1971–1978 (2014) 2, 8
3. Chen, X., Yuan, Y., Zeng, G., Wang, J.: Semi-supervised semantic segmentation with cross pseudo supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2613–2622 (2021) 3, 10, 14
4. Cheng, B., Parkhi, O., Kirillov, A.: Pointly-supervised instance segmentation. arXiv preprint arXiv:2104.06404 (2021) 2, 3, 11, 14
5. Cho, J.H., Mall, U., Bala, K., Hariharan, B.: Picie: Unsupervised semantic segmentation using invariance and equivariance in clustering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16794–16804 (2021) 4
6. Dai, J., He, K., Sun, J.: Instance-aware semantic segmentation via multi-task network cascades. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016) 4
7. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2), 303–338 (Jun 2010) 8
8. Fifty, C., Amid, E., Zhao, Z., Yu, T., Anil, R., Finn, C.: Efficiently identifying task groupings for multi-task learning. Advances in Neural Information Processing Systems **34** (2021) 4
9. Guo, P., Lee, C.Y., Ulbricht, D.: Learning to branch for multi-task learning. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 3854–3863. PMLR (13–18 Jul 2020), https://proceedings.mlr.press/v119/guo20e.html 4
10. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017) 3, 4
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 19
12. Heuer, F., Mantowsky, S., Bukhari, S., Schneider, G.: Multitask-centernet (mcn): Efficient and diverse multitask learning using an anchor free approach. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops. pp. 997–1005 (October 2021) 4
13. Hsu, C.C., Hsu, K.J., Tsai, C.C., Lin, Y.Y., Chuang, Y.Y.: Weakly supervised instance segmentation using the bounding box tightness prior. Advances in Neural Information Processing Systems **32** (2019) 3
14. Hung, W.C., Jampani, V., Liu, S., Molchanov, P., Yang, M.H., Kautz, J.: Scops: Self-supervised co-part segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 869–878 (2019) 3
15. Khoreva, A., Benenson, R., Hosang, J., Hein, M., Schiele, B.: Simple does it: Weakly supervised instance and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 876–885 (2017) 3

16. Kocabas, M., Karagoz, S., Akbas, E.: Multiposenet: Fast multi-person pose estimation using pose residual network. In: Proceedings of the European Conference on Computer Vision (ECCV) (September 2018) 4
17. Laradji, I.H., Rostamzadeh, N., Pinheiro, P.O., Vazquez, D., Schmidt, M.: Proposal-based instance segmentation with point supervision. In: 2020 IEEE International Conference on Image Processing (ICIP). pp. 2126–2130. IEEE (2020) 3
18. Lin, D., Dai, J., Jia, J., He, K., Sun, J.: Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3159–3167 (2016) 3
19. Liu, W., Wu, Z., Ding, H., Liu, F., Lin, J., Lin, G.: Few-shot segmentation with global and local contrastive learning. arXiv preprint arXiv:2108.05293 (2021) 4
20. Naha, S., Xiao, Q., Banik, P., Reza, M., Crandall, D.J., et al.: Part segmentation of unseen objects using keypoint guidance. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1742–1750 (2021) 3
21. O Pinheiro, P.O., Almahairi, A., Benmalek, R., Golemo, F., Courville, A.C.: Unsupervised learning of dense visual representations. Advances in Neural Information Processing Systems **33**, 4489–4500 (2020) 3
22. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems **28** (2015) 4
23. Rother, C., Kolmogorov, V., Blake, A.: " grabcut" interactive foreground extraction using iterated graph cuts. ACM transactions on graphics (TOG) **23**(3), 309–314 (2004) 3
24. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y 11
25. Saha, O., Cheng, Z., Maji, S.: GANORCON: Are Generative Models Useful for Few-shot Segmentation? In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9991–10000 (2022) 4
26. Standley, T., Zamir, A., Chen, D., Guibas, L., Malik, J., Savarese, S.: Which tasks should be learned together in multi-task learning? In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 9120–9132. PMLR (13–18 Jul 2020), https://proceedings.mlr.press/v119/standley20a.html 4
27. Tian, Z., Shen, C., Wang, X., Chen, H.: Boxinst: High-performance instance segmentation with box annotations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5443–5452 (2021) 3
28. Tripathi, S., Collins, M., Brown, M., Belongie, S.: Pose2instance: Harnessing keypoints for person instance segmentation. arXiv preprint arXiv:1704.01152 (2017) 4
29. Tritrong, N., Rewatbowornwong, P., Suwajanakorn, S.: Repurposing gans for one-shot semantic part segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4475–4485 (2021) 4
30. Van Horn, G., Branson, S., Farrell, R., Haber, S., Barry, J., Ipeirotis, P., Perona, P., Belongie, S.: Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 595–604 (2015). https://doi.org/10.1109/CVPR.2015.7298658 14

31. Vedaldi, A., Mahendran, S., Tsogkas, S., Maji, S., Girshick, B., Kannala, J., Rahtu, E., Kokkinos, I., Blaschko, M.B., Weiss, D., Taskar, B., Simonyan, K., Saphra, N., Mohamed, S.: Understanding objects in detail with fine-grained attributes. In: Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2014) 2, 9
32. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. Tech. rep. (2011) 2, 8
33. Wang, Y., Zhang, J., Kan, M., Shan, S., Chen, X.: Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12275–12284 (2020) 3
34. Welling, M., Teh, Y.W.: Bayesian learning via stochastic gradient langevin dynamics. In: Proceedings of the 28th international conference on machine learning (ICML-11). pp. 681–688. Citeseer (2011) 6
35. Yang, Y., Bilen, H., Zou, Q., Cheung, W.Y., Ji, X.: Unsupervised foreground-background segmentation with equivariant layered gans. arXiv preprint arXiv:2104.00483 (2021) 3
36. Zhang, Y., Ling, H., Gao, J., Yin, K., Lafleche, J.F., Barriuso, A., Torralba, A., Fidler, S.: Datasetgan: Efficient labeled data factory with minimal human effort. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10145–10155 (2021) 4
37. Zhou, Y., Zhu, Y., Ye, Q., Qiu, Q., Jiao, J.: Weakly supervised instance segmentation using class peak response. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3791–3800 (2018) 3
38. Zhu, Y., Zhou, Y., Xu, H., Ye, Q., Doermann, D., Jiao, J.: Learning instance activation maps for weakly supervised instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3116–3125 (2019) 3
39. Zou, Y., Zhang, Z., Zhang, H., Li, C.L., Bian, X., Huang, J.B., Pfister, T.: Pseudoseg: Designing pseudo labels for semantic segmentation. arXiv preprint arXiv:2010.09713 (2020) 3