Part Segmentation of Unseen Objects using Keypoint Guidance

Shujon Naha Qingyang Xiao Prianka Banik Md Alimoor Reza David J. Crandall
Luddy School of Informatics, Computing, and Engineering
Indiana University
{snaha,mdreza}@iu.edu, {xiaoq,djcran}@indiana.edu, prianka.banik.buet@gmail.com

Abstract

While object part segmentation is useful for many applications, typical approaches require a large amount of labeled data to train a model for good performance. To reduce the labeling effort, weak supervision cues such as object keypoints have been used to generate pseudo-part annotations which can subsequently be used to train larger models. However, previous weakly-supervised part segmentation methods require the same object classes during both training and testing. We propose a new model to use keypoint guidance for segmenting parts of novel object classes given that they have similar structures as seen objects — different types of four-legged animals, for example. We show that a non-parametric template matching approach is more effective than pixel classification for part segmentation, especially for small or less frequent parts. To evaluate the generalizability of our approach, we introduce two new datasets that contain 200 quadrupeds in total with both keypoint and part segmentation annotations. We show that our approach can outperform existing models by a large margin on the novel object part segmentation task using limited part segmentation labels during training.

1. Introduction

Object part segmentation is the problem of producing pixel-level semantic annotations that indicate fine-grained object parts instead of just object labels. Part segmentation has a wide range of practical applications such as fine-grained object classification [24], pose estimation [18], object re-identification [2], etc. While recent deep learning based methods [5,6,15] give impressive results for part segmentation, most focus on training a segmentation model for a single object class. However, training a part segmentation model for a *new* object in this setting requires annotating a large quantity of training images with fine-grained, pixel-wise part segmentation masks, which can be extremely labor-intensive.

To avoid the need for expensive manual annotation, some recent work has considered weakly-supervised approaches for part annotation. Fang et al. [3] propose transfer learning to generate pixel-level part annotations for an unlabeled target object instance by using keypoints to propagate part segmentation knowledge from a labeled source object instance of the same class. As annotating keypoint locations is significantly less labor-intensive than generating pixel-wise part masks, this approach can greatly reduce the manual annotation cost. While promising, their work required the source and target object instances to come from the same class — i.e., transferring part segmentation annotations from one person to another, or from one animal photo to another instance of the same animal species. Thus their approach still requires annotated data for each separate object class.

Figure 1. Quadrupeds vary widely in both shape and local appearance, but nevertheless share similar body parts. Our goal is to learn a generalized part segmentation model that can take an image and corresponding keypoint annotations (top) of a *previously unseen class* of animal, and produce a part segmentation map (bottom).

In this paper, we propose the novel idea that part segmentation annotations from one object *class* could be used to generate part annotation for other *classes*. Many object classes share similar parts, even if their overall appearances are quite different. We introduce and evaluate an approach to take a small labeled set of object classes and use it to segment the parts of an instance of a new object class, with only minimal human annotation in the form of keypoints. Our approach should apply to a wide range of object classes having similar parts and structure. We evaluate on one specific family of object classes — quadruped (four-legged) an-

imals — which has widely different sizes and appearances but share similar parts and structure (see Figure 1).

In particular, we present a novel technique using a CNN model to combine both appearance and structural information to estimate object parts. This allows our model to transfer part information from a limited number of known object classes to novel object classes with similar structure by considering the keypoint annotations as the transfer medium. Our model can handle diverse and novel object poses, and does not require the source and target objects to have similar poses. We also handle the problem of segmenting small and relatively rare object parts (e.g. tail for quadruped animals) by using a non-parametric prediction approach. We perform extensive experiments to show that our approach can effectively transfer part segmentations from known objects classes to novel objects, even with large pose changes, better than the existing models. Due to the limited number of datasets appropriate for this novel cross object class part segmentation problem, we have created two new datasets [1] with a total 200 quadruped animals with both keypoint and part segmentation annotations.

To summarize, we make the following contributions:

- We develop an end-to-end learning approach to transfer pixel-level object part segmentations from a fully labeled object set to another weakly-labeled object set, using keypoint locations to guide the transfer learning process;
- We show that our model can generate part segmentation labels for unseen object classes with similar semantic parts as the training objects;
- We evaluate our technique against several baselines and on several datasets, including two new datasets with a total 200 animal images with both keypoint and part segmentation annotations.

2. Related Work

Relevant related work for this paper includes pose-guided part segmentation models and weakly-supervised semantic segmentation approaches.

Pose Guided Part Segmentation. Several papers have considered using object keypoint locations to improve object part segmentation accuracy. Xia et al. [21] used a combination of pose estimation and intermediate semantic part score maps for refining part segmentations, and also explored pose-guided segmentation proposal [22]. Mutual feature sharing between pose prediction and part segmentation was proposed by Nie et al. [12] for improving the accuracy of both problems. Zhao et al. [26] learned to accumulate weighted multi-scale features for improved human parsing with the constraint of explicit part-joint consistency. Naha et. al. [11] directly converted pose to pseudo part segmentation and used it to guide the final part segmentation predictions. But all of this work used strong supervision, and none of it explored transferring segmentations across object classes.

Weakly Supervised Semantic Segmentation. Weak supervision for semantic segmentation can come in different forms such as point supervision [1], scribbles [9], bounding boxes [7], etc. An iterative refinement approach for transforming pose-based part priors for human body part segmentation has been recently proposed by Yang et al. [23]. While these approaches can be applied for a specific object class, it is difficult to generalize them to new object classes due to the difference in part shapes and appearances.

3. Our Approach

Our goal is to train a model that can segment the body parts of an instance of a novel (previously unseen) object class, given only the image and keypoints of that instance. In this paper, we specifically consider transferring among different four-legged animals, although the approach could be applied much more generally. We use the animal case here just for ease of discussion. The training dataset contains a very small number C of object classes (which is much smaller than the number of quadruped species in the world). Assume all training and test objects have a maximum of p body parts and k keypoints. Consider a training instance as $s_i^c = \{I_i^c, K_i, P_i\}$, $i = 1...N^c$, where $I_i^c \in R^{h \times w \times 3}$ is an input image of class c, $K_i \in R^{h \times w \times k}$ are the heatmaps generated from the set of k 2D keypoint annotations, and $P_i \in R^{h \times w \times p}$ is the corresponding part segmentation of the object. Let $N = \sum_{c=1}^{C} N^c$ be the total number of training images. Now consider a test instance $x^{c'} = \{I_t^{c'}, K_t\}$ where c' can be a completely different quadruped species than any c in C. Our goal is to use the provided keypoint annotations of $x^{c'}$ to transfer the part segmentation labels from the training animals to generate part annotations for $x^{c'}$.

3.1. Overview

Our model consists of three main parts: (1) the Structural module, (2) the Visual module, and (3) the Transfer block. The Structural module encodes the keypoints of a given instance and provides useful structural information to the Visual module, which then takes an image as input and generates feature representations for estimating the part segmentations of the given object. The Transfer block allows both the visual module and the structural module to communicate with each other for propagating useful structural and appearance information.

To estimate the final part segmentations, we use template images which then produce part basis feature representa-

[1] Datasets are available at http://vision.sice.indiana.edu/animal-pose-part

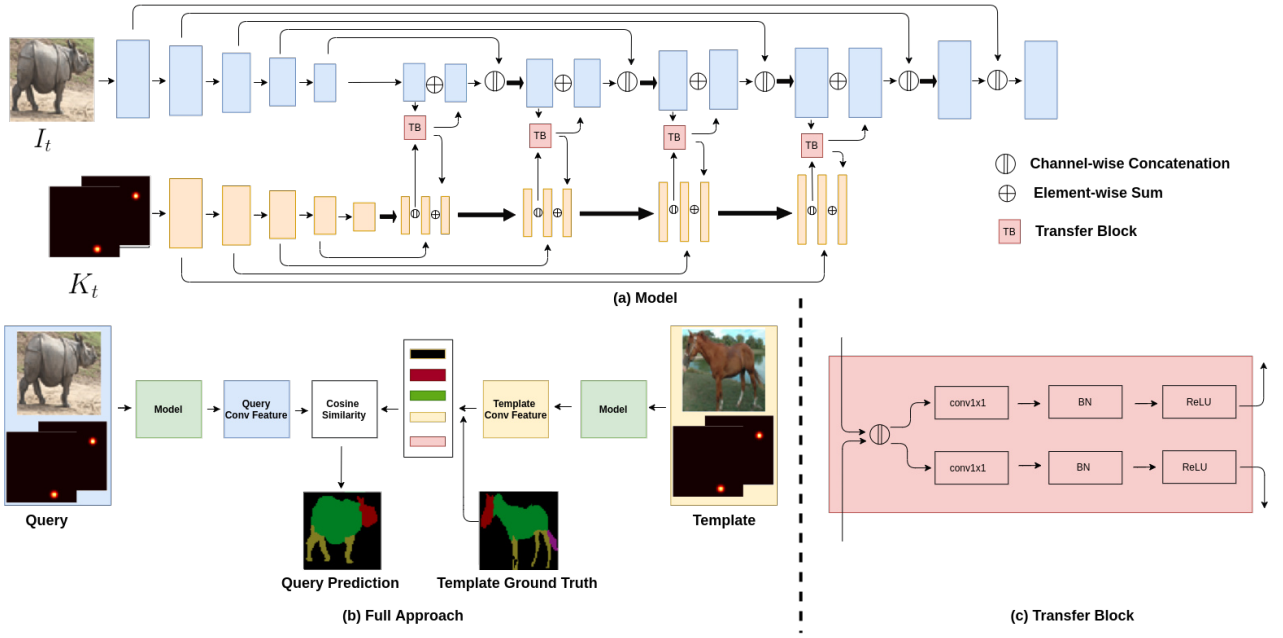

Figure 2. Pipeline of our approach. (a) The Visual module takes the target image I_t and the Structural module takes the keypoint heatmaps K_t as input, and generates a convolutional feature representation. (b) The convolutional feature and the ground truth part segmentation map of a template object is used to generate the final prediction mask of a query object. (c) Detailed architecture of the transfer block used for interactions between the visual and structure streams.

tions. The part features from the template images are used to predict the part class label of each pixel of the given target image. An overview of the complete approach can be seen in Figure 2.

3.2. Structural Module

Our goal is to generate body part segmentation maps of images of unseen quadruped animals using the information from few instances of very few other quadruped classes. A straightforward approach to this problem is to train a fully-convolutional neural network (FCN) using the training set and then simply apply it on the new animal. However, the new animal can have quite different body shape and appearance from the training animals, even though they share the same number of body parts. To transfer the part information from the training classes to any novel animal, we use keypoint locations to guide the part segmentation estimation process. We assume the keypoint locations are already given for each image, either through an accurate automatic algorithm or through manual annotation (which can easily be provided with just a few clicks, unlike a dense segmentation map which is extremely labor intensive). Since keypoints are common between the known and the novel objects, they can be used to propagate part information from the training objects to any novel test object by binding the part predictions with the keypoints.

Our Structural module learns to transform the pose or keypoint annotations of a given object to useful semantic information for predicting part segmentations. This module consists of a U-Net [14] , which is an encoder-decoder CNN with skip connections. We first convert the keypoint locations to 2D heatmaps for each of the instances. Let us consider the keypoint heatmap label of the target object as K_t. The encoder reduces the spatial dimension of the input so that the network can understand the relative locations of the keypoints of the target object, and the decoder then transforms the keypoint locations to useful information for part segmentation. Consider the output of decoder step i of the Structural module as S_t^i.

3.3. Visual Module

While the Structural module provides structural information from keypoints only, we need another module to capture the appearance information of the given object. The Visual module is a fully-convolutional encoder-decoder network with skip connections. The network first encodes the target input image $I_t^{c'}$ from class c' as a convolutional feature map and then passes it through a series of learnable deconvolution layers to predict the final part segmentation output. The structural information of the object from the Structural module is propagated to the Visual module at the decoding stages for generating the final convolution feature representation of the object. This allows both the visual and structural features to complement each other and produce a more refined part segmentation results than the Visual module could do by using appearance information alone. We

denote the output of decoder step i of the Visual module as A_t^i.

3.4. Transfer Block

The Transfer block serves as the communication medium between the Structural and Visual modules. The Transfer block takes A_t^i from the Visual module and S_t^i from the Structural module, which are both convolutional features with the same height and width. The Transfer block then concatenates these feature representations and passes them through two different streams, each consisting of a convolution layer with 1x1 kernel, a batch normalization layer, and a ReLU layer. We call these outputs $\widetilde{A_t^i}$ and $\widetilde{S_t^i}$. The number of output channels of $\widetilde{A_t^i}$ and $\widetilde{S_t^i}$ are the same as those of A_t^i and S_t^i, respectively. We then modify A_t^i and S_t^i by adding $\widetilde{A_t^i}$ and $\widetilde{S_t^i}$ to them, respectively.

Then we pass the modified A_t^i and S_t^i to the next decoding stage, so that both the Structural and Visual streams receive complementary information for better part segmentation. We apply this Transfer block after every decoding stage of the Structural module and after the first four decoding stages of the Visual module. A depiction of a Transfer block is in Fig 2(c).

3.5. Template Generation

Since we assume that we have very few training images annotated with part segmentations, using a typical convolution layer for the final prediction does not perform as well for small, often occluded parts (e.g., tail) as it does for larger and more visible parts (e.g., torso). To alleviate this problem, we use a non-parametric template matching approach for pixel-wise class prediction that was previously explored for few-shot segmentation [19]. We found this technique was particularly helpful for relatively rare parts, especially with few training images.

For generating the template of each part during training, we randomly sample an image I_s^c having all the parts present for each training target instance $I_t^{c'}$. If no such template image is available, we can use multiple images as templates to generate feature representations of all the parts, but for our case we always found enough template images with all parts in the training dataset. We also ensure that the template object is from a different class than the target training instance (i.e., c' is different than c), to encourage generalization between the parts of different object classes despite their appearance differences (e.g., horses and cows have very different tails). After selecting a template, we pass the template image I_s^c and corresponding keypoints K_s through our model to generate the final convolution feature output A_s from the visual module. We then use the ground truth part segmentation mask P_s of the template image I_s^c to generate the template features t_j for each part j (including background) using masked average pooling [25],

$$t_j = \frac{\sum_l A_s^l \mathbb{I}[P_s^l = p]}{\sum_l \mathbb{I}[P_s^l = p]}, \tag{1}$$

where l is a pixel location and $\mathbb{I}[.]$ is an indicator function that produces 1 for true and 0 for a false argument. A cosine similarity map is then generated using each template part feature t_j and the target object feature A_t to generate the part segmentation prediction. The cosine map is also multiplied by a fixed value following [19]. The cosine maps for all the parts are concatenated to generate the prediction map $\widetilde{P_t}$, with number of channels equal to p.

	Pascal Part					COCO Part	AwA Part
	Sheep	Horse	Cow	Dog	Cat		
train	2627	2468	2639	1775	1979	N/A	N/A
test	245	404	233	1097	893	100	100

Table 1. The number of instances in three part segmentation datasets. For Pascal, we train on 4 categories and test on the other one. For COCO and AwA, we train only on all 2872 images from the Pascal Part dataset to guarantee that the testing animals are not seen during training.

3.6. Training

The model is trained end-to-end using a per-pixel cross entropy loss. $\widetilde{P_t}$ is first passed through a softmax function to produce $\widetilde{PS_t}$ and then finally passed to the cross entropy loss function to calculate the segmentation loss, $loss_{seg}$,

$$loss_{seg} = -\frac{1}{L} \sum_l \sum_{j \in p} \mathbb{I}[P_t^l = j] \log \widetilde{PS_{t;j}^l}, \tag{2}$$

where L is the total number of pixel locations.

4. Experiments

We conducted extensive experiments to evaluate the effectiveness of our proposed approach.

4.1. Dataset

There are very few publicly available datasets for quadruped animals that have annotated part segmentation. To address this limitation, we annotated two additional datasets, covering more novel classes.

Pascal Part is a part-segmentation dataset which also contains keypoint locations and bounding box annotations [20]. We use the same setup as [11] and only consider the images containing any of the 5 quadruped animals: *Cat, Cow, Horse, Dog*, and *Sheep*. The ground truth bounding box annotations are used to crop the objects from the images, so each image contains a single quadruped animal. This preprocessing yields a total of 2,872 images from the 5

quadruped classes. Following the previous work [3, 11], we only consider 4 parts for each animal: *head, body, legs,* and *tail*, although the dataset contains part annotations of more, finer-grained parts as well.

AwA Part is a dataset based on the Animals with Attributes dataset. AwA has 50 animal classes and is widely used for zero-shot learning [8]. This dataset contains only the class and attribute labels for each object, without any keypoint or part annotation labels. We selected 10 quadruped animal classes, none of which overlap with any of the 5 quadrupeds in Pascal Part although they have structural similarities. The animals are *Antelope, Bobcat, Buffalo, Fox, Giant Panda, Leopard, Lion, Pig, Rhinoceros,* and *Wolf*. We randomly selected 10 images for each of these 10 classes with various poses, and then manually annotated each of these 100 images with both keypoints and pixel-level part segmentation labels using the publicly available LabelMe annotation tool [16]. We also calculate the top-left and right-bottom locations of each object using the annotated keypoints, add a margin of 50 pixels around them to generate a pseudo bounding box annotation, and crop the object using the bounding box. We use this dataset only for testing and not for training.

COCO Part is based on the COCO dataset [10] and contains 9 quadruped animals, of which 5 overlap with the quadrupeds in the Pascal Part dataset. We selected the other 4 quadrupeds, *Zebra, Giraffe, Elephant* and *Bear*, and sampled 25 images for each of these 4 classes. Like AwA Part, we then created another new test dataset by annotating these 100 images with keypoints and part segmentation labels. It is more challenging than Pascal Part and AwA Part due to animals like *Giraffe* and *Elephant* which have significantly different body structure from the other quadrupeds. We apply the same cropping method as with the AwA Part dataset. Like AwA Part, this dataset is also used only for evaluation, not for training.

4.2. Implementation Details

For the Structural module, we first convert the keypoint 2D location annotations to heatmaps using a Gaussian function with $\sigma = 7$, where each heatmap has a height and width of 128. The encoder of the Structural module has 5 downsampling residual blocks and the decoder has 4 upsampling residual blocks. The upsampling blocks use bilinear upsample layers. For the Visual module, we use the encoder-decoder network with skip-connections from [17]. The encoder part of the network consists of an Imagenet-pretrained VGG-16 network, and the decoder consists of a series of 5 upsampling blocks with learnable deconvolution layers. The Transfer block consists of two separate sets of blocks, each containing a 1x1 convolution layer, a batch normalization layer, and a ReLU activation layer in series. All the layers in the model are learned during training. We train the full network end-to-end using cross-entropy loss.

We use 5 fold cross-validation to consider one animal class at a time as the test class, while the remaining four classes are training classes for the Pascal Part dataset. The numbers of training and test images for each class are given in Table 1. For the other two datasets, we train the network using all the images in the Pascal Part dataset and use the trained network for predicting part segmentations of the objects in AwA Part and COCO Part. We use batch size 24 and resize the input images and the ground truth part segmentations to 256×256 during training. We use the RMSprop optimizer for training with a learning rate of 0.0001 and train the network for 150 epochs. All the experiments were done using a single NVidia Titan X GPU. We use PyTorch [13] to implement our model.

4.3. Baselines

We consider several other models as baselines to compare with our results.

RefineNet proposed by Fang et al. [3], transforms part annotations of similar pose source objects to a target object part annotation using keypoint labels and affine transformations. For this baseline, we use five nearest neighbor labeled source objects for each target object for transferring the part segmentation. We use the source code provided by the authors for the prior generation and refinement networks. In our case, source and target objects can have different numbers of visible keypoints, so we try to approximate the invisible keypoints as much as possible (e.g. use the average of the locations of left and right eyes to estimate the location of the nose in cases when it is invisible) to make sure that the source and target keypoints are matched, and then follow the same pipeline as in [3]. Also, during training, we use the same "separate source and target class" approach used for our model.

Transform is a pose-to-part module proposed by Naha et al. [11]. This network takes keypoint locations as input and directly convert them to part segmentations using a U-Net.

Hourglass network is widely used for keypoint prediction and part segmentation problems. We follow the setting of Nie et al. [12] to train an hourglass network only for part segmentation without using the keypoint prediction module. The hourglass network uses multi-stage loss functions and is trained from scratch.

TernausNet [17] is the encoder-decoder network with skip-connections used as our Visual module. This baseline only takes the target image as input and does not use keypoint annotation for the part segmentation.

TernausNet+ is a modified TernausNet model to take keypoints as input in addition to the image. We produce multi-

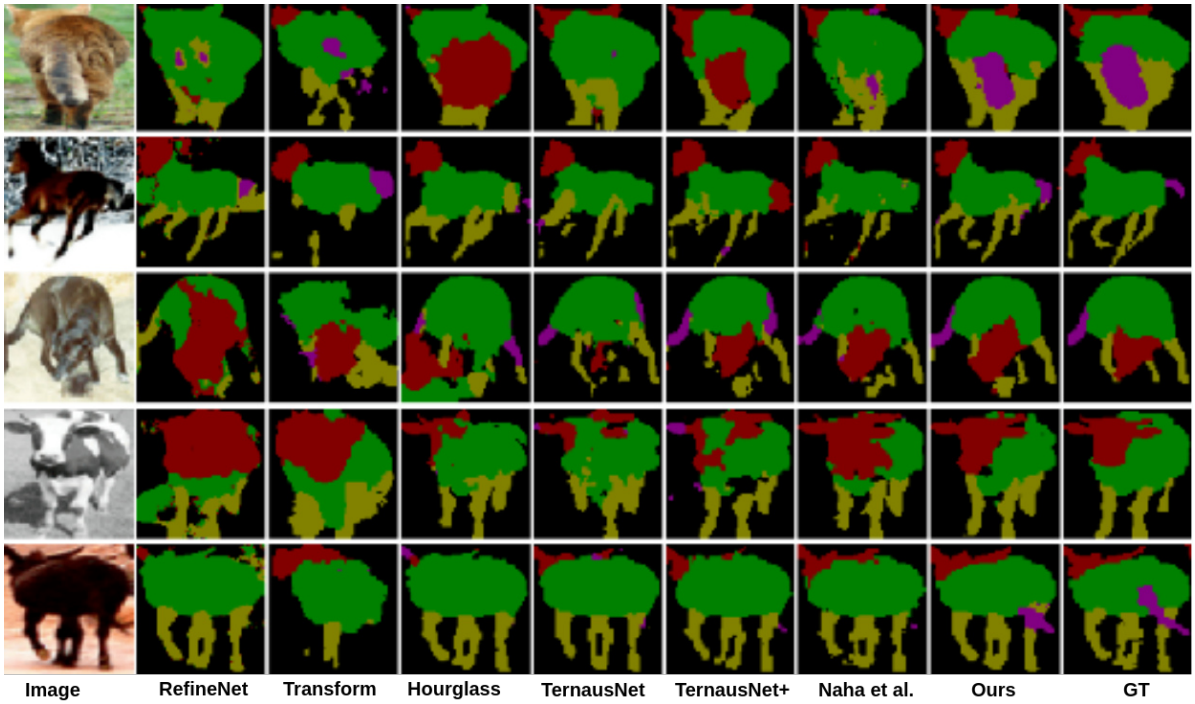

Figure 3. Qualitative comparison on Pascal Part dataset.

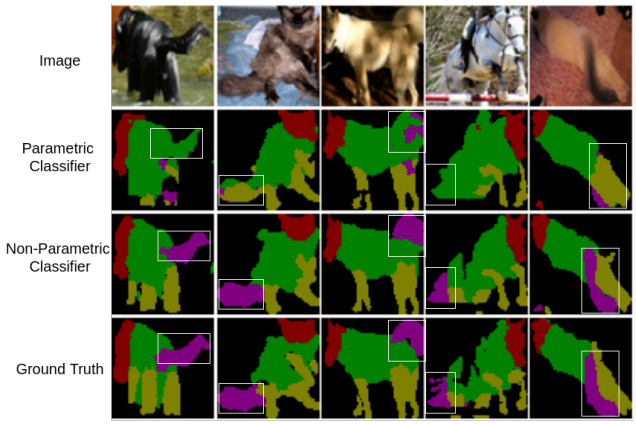

Figure 4. Qualitative comparison on *tail* segmentation using regular parametric pixel classification approach and the non-parametric template matching approach on Pascal Part Dataset. The second row shows the results from parametric classification and the third row shows the results of the non-parametric template matching approach. The last row shows the ground truth.

scale keypoint heatmaps and concatenate them with each decoder input features of TernausNet. This model makes the structural information available to the TernausNet while predicting the part segmentation in a simplistic manner.

Naha et al. [11] combined pseudo part segmentation generated by a pose-to-part module with a visual evidence module. Their approach is conceptually similar to ours but uses a less effective method to propagate knowledge from the keypoint and known object part segmentation labels.

Ours (Classifier) is our network with a convolutional classification layer at the end instead of using template matching for pixel classification. We use this baseline to show the effects of using a regular classification layer and cosine similarity-based non-parametric approach for part segmentation from limited data.

4.4. Evaluation on Pascal Part Dataset

We applied five-fold cross-validation (considering four animals for training and one animal for testing) on the Pascal Part Dataset. Table 2 shows comparisons with the baselines, using intersection-over-union (IOU) as the evaluation metric. The table shows that our model outperforms all baselines for each of the animals. RefineNet has the worst performance among all the baselines which is consistent with the results from [11]. Transform performs better than RefineNet but the results are significantly worse compared to the other models, which suggests that pose information alone is not enough for accurate part segmentation.

Hourglass, our third baseline, performs significantly better than RefineNet and Transform but fails to generalize to novel test classes. As Hourglass is a large network with many parameters, it can easily overfit to the training classes, but performs better when there is more training data (such as for *sheep* and *cow*). In spite of having fewer parameters than Hourglass, TernausNet performs much better in terms of generalization. This shows that the appearance simi-

larities between different quadruped animals can be used to perform generalized part segmentation to some extent. The results of TernausNet+ show that combining the structural information in the form of keypoint heatmaps with visual features helps to produce better part segmentation results. But TernausNet+ still performs significantly worse than our model, which also indicates that simply concatenating structural information with visual features does not yield major improvements. Naha et al. [11] performs much better than these baselines which shows directly generating part labels from the pose and incorporating it in the visual stream can significantly improve part segmentation even for novel classes.

All of these baseline models performs significantly better for *sheep, cow,* and *horse* compared to *cat* and *dog,* presumably because *cat* and *dog* have both significantly less training data and more diverse poses than the other classes. Interestingly, our model performs similarly for all the classes irrespective of their pose difficulties and amount of training data, as can be seen in Table 2. This suggests that our model more efficiently uses the structural information for performing generalized part segmentation of novel objects.

Finally, the parametric classifier approach performs significantly better than Naha et al. [11] approach, which shows that our proposed transfer block is more effective in utilizing pose information for improved part segmentation. But our non-parametric approach performs significantly better overall (in terms of average IOU across parts) and especially for the *tail*, while sometimes performing slightly worse for some other larger parts. We expect this is because the *tail* is often occluded by the larger and more visible body parts such as *torso* and *legs*, and thus the parametric classification model often mistakes *tail* regions for either *legs* or *torso*. On the other hand, the template matching-based approach gives more equal importance to all parts at inference time. This finding is aligned with the results in [4, 19] which showed that cosine similarity-based non-parametric classifiers perform better for classes with less training data. Qualitative comparisons of *tail* segmentation between our template-based model and the convolutional classifier-based model can be seen in Figure 4.

Fine-Grained Part Segmentation. In addition to the above experiments using the same number of parts as previous work, we also perform experiments on fine-grained part segmentation with more part categories. In particular, we consider *left legs* and *rights legs* as two separate parts. As shown in Table 3, our model again outperforms all the baselines except that the regular classifier performs slightly better than the template matching approach for *cow.* Interestingly, the performance gaps increase for the five-part segmentation task between our model and the best performing baseline [11] compared to the four-part segmentation task for some animals. For example, the difference between our

Method	Pose	BG	Head	Torso	Legs	Tail	Avg
Test on Sheep:							
RefineNet [3]	+	43.66	8.86	35.90	7.67	0.21	19.26
Transform [11]	+	63.93	59.94	56.86	28.07	12.14	44.18
Hourglass [12]		81.62	57.30	73.99	47.21	6.97	53.41
TernausNet [17]		83.59	66.60	77.97	51.40	7.62	57.43
TernausNet+	+	83.56	69.73	78.11	51.95	8.38	58.34
Naha et al. [11]	+	83.61	73.95	77.42	52.96	11.28	59.84
Ours (Classifier)	+	84.68	**76.33**	**80.33**	55.31	12.57	61.84
Ours (Template)	+	**85.01**	75.62	80.23	**56.01**	**22.02**	**63.77**
Test on Horse:							
RefineNet [3]	+	45.79	10.29	27.83	8.24	0.98	18.63
Transform [11]	+	66.39	58.84	53.85	28.98	7.58	43.12
Hourglass [12]		80.42	42.73	58.62	48.60	12.42	48.55
TernausNet [17]		83.04	59.04	66.19	52.11	26.32	57.34
TernausNet+	+	83.54	64.15	67.86	55.47	22.70	58.74
Naha et al. [11]	+	83.85	70.29	69.15	58.72	30.90	62.58
Ours (Classifier)	+	85.43	**72.46**	70.96	60.66	29.45	63.79
Ours (Template)	+	**85.76**	72.44	**71.99**	**60.67**	**42.82**	**66.73**
Test on Cow:							
RefineNet [3]	+	44.78	11.43	31.54	8.34	0.51	19.32
Transform [11]	+	64.65	57.84	61.39	30.34	6.03	44.04
Hourglass [12]		79.09	50.72	64.42	46.98	12.81	50.80
TernausNet [17]		81.53	60.28	69.73	50.53	12.94	55.00
TernausNet+	+	81.87	67.81	72.09	52.88	16.35	58.19
Naha et al. [11]	+	82.65	75.65	75.15	55.43	20.82	61.93
Ours (Classifier)	+	**83.42**	76.56	76.24	**56.97**	21.97	63.03
Ours (Template)	+	83.13	**77.62**	**76.99**	55.93	**22.75**	**63.28**
Test on Dog:							
RefineNet [3]	+	42.22	18.58	19.71	8.32	0.28	17.82
Transform [11]	+	62.41	63.60	47.64	29.81	8.62	42.41
Hourglass [12]		78.26	54.18	46.93	33.39	6.85	43.92
TernausNet [17]		82.52	66.82	56.26	42.22	10.80	51.72
TernausNet+	+	83.12	69.12	57.24	44.96	12.12	53.31
Naha et al. [11]	+	83.82	76.10	60.65	48.72	18.36	57.52
Ours (Classifier)	+	85.21	**79.75**	**64.06**	52.45	23.12	60.91
Ours (Template)	+	**85.34**	78.50	63.72	**52.94**	**34.19**	**62.93**
Test on Cat:							
RefineNet [3]	+	37.99	19.10	20.94	7.24	0.21	17.10
Transform [11]	+	58.13	65.46	43.19	18.10	5.50	38.07
Hourglass [12]		74.40	51.92	49.81	26.12	2.40	40.92
TernausNet [17]		81.50	68.46	60.10	32.30	9.74	50.41
TernausNet+	+	81.61	70.82	61.44	33.87	11.55	51.85
Naha et al. [11]	+	81.42	79.52	64.07	39.32	16.17	56.09
Ours (Classifier)	+	84.55	81.88	**69.01**	39.10	19.00	58.70
Ours (Template)	+	**84.58**	**82.56**	68.38	**42.71**	**28.93**	**61.43**

Table 2. Evaluation on Pascal Part dataset in terms of 4-part parsing. We test on one category and train on the other four. BG denotes background and Avg is the average across parts.

Method	Pose	Sheep	Horse	Cow	Dog	Cat
RefineNet [3]	+	16.03	15.51	16.23	14.87	14.20
Transform [11]	+	38.43	39.62	40.31	38.53	35.06
Hourglass [12]		29.86	35.51	44.81	37.9	36.02
TernausNet [17]		48.84	48.1	48.32	45.58	45.58
TernausNet+	+	50.39	49.67	50.92	46.31	46.51
Naha et al. [11]	+	55.21	56.85	57.97	53.45	50.85
Ours (Classifier)	+	56.83	58.98	**59.64**	55.86	54.09
Ours (Template)	+	**59.08**	**59.49**	59.07	**59.99**	**58.29**

Table 3. Study of 5-part parsing including *background, head, torso, left legs, right legs*, and *tail*. Results tested on Pascal Part dataset in terms of the mean IoU.

Method	Pose	AwA						COCO					
		BG	Head	Torso	Legs	Tail	Avg	BG	Head	Torso	Legs	Tail	Avg
RefineNet [3]	+	48.25	11.05	31.20	14.25	0.60	21.07	58.47	8.14	31.95	14.42	0.63	22.72
Transform [11]	+	68.30	62.38	61.51	31.60	5.86	45.93	76.20	51.48	59.32	37.07	5.49	45.91
Hourglass [12]		80.52	58.34	65.50	46.22	11.34	52.39	85.86	36.38	60.23	50.11	9.75	48.47
TernausNet [17]		83.17	70.47	71.28	53.93	23.30	60.43	88.76	41.79	64.30	54.35	10.87	52.01
TernausNet+	+	83.42	72.03	70.63	55.83	20.36	60.45	89.84	46.22	65.84	58.84	12.57	54.66
Naha et al. [11]	+	83.54	75.99	73.13	59.14	21.64	62.69	**91.09**	64.93	74.20	**63.63**	22.21	63.21
Ours (Classifier)	+	**84.48**	77.34	74.39	56.96	21.66	62.96	89.79	66.15	72.49	63.44	23.08	62.98
Ours (Template)	+	84.35	**78.16**	**74.64**	**59.45**	**25.50**	**64.42**	90.15	**66.59**	**74.66**	63.47	**24.24**	**63.82**

Table 4. Evaluation on AwA and COCO dataset with **novel** classes in terms of 4-part parsing. BG denotes as background.

Figure 5. Qualitative results on AwA Part dataset.

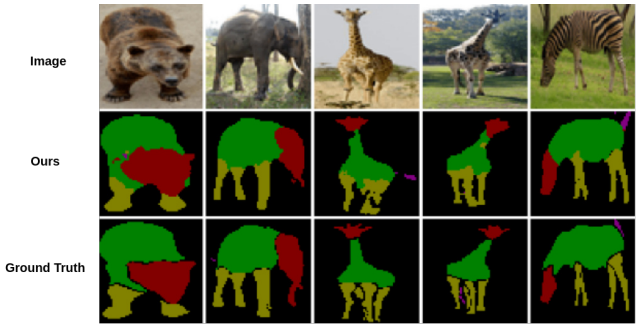

Figure 6. Qualitative results on COCO Part dataset.

model and [11] for the four-part segmentation task on the *dog* class is around 5.41 percentage points, compared to 6.45 for the five-part segmentation task. Similarly for the object class *cat,* the difference between [11] and our model is 5.34 percentage points when the number of parts is four but becomes 7.44 when the number of parts is five. This indicates that our model can distinguish fine-grained parts much more effectively than the baseline models.

4.5. Evaluation on AwA Part Dataset

For the AwA Part dataset, we train the network on the full Pascal Part dataset and then use the objects in this dataset for testing. As shown in Table 4, our model performs best in terms of mean IOU but the performance gains are not as high as on Pascal Part. This indicates that when there is enough training data and the target classes have similar shapes as the source classes, any segmentation model can perform relatively well. Table 4 also shows that the difference in performance between the template matching and the regular classification approaches for *tail* segmentation is not nearly as high as before. This suggests that the cosine similarity-based classifier performs very similarly to the regular classifier when enough training data is available. Qualitative results for this dataset are in Figure 5.

4.6. Evaluation on COCO Part Dataset

We again use the models trained on the full Pascal Part dataset to estimate part segmentations for objects in COCO Part. The unusually long neck of *giraffe* and the trunk of the *elephant* classes create significant challenges to the baseline models for this dataset. But our model still outperforms the baseline models in terms of mean IOU, as shown in Table 4. Qualitative results on this dataset can be seen in Figure 6.

5. Conclusion

In this paper, we introduce the novel problem of cross-class part segmentation using keypoint guidance. Our proposed approach utilizes keypoint annotations for transferring part annotations from a small labeled known quadruped set to any novel quadruped animal with the same number of body parts. We show that by using an effective transfer learning mechanism, such generalization can be achieved even when the amount of training labels is very small. We also show that the existing model can achieve the same level of generalization with a larger training dataset but performs much worse when the labeled examples are few. We hope our work will inspire more work on the cross-class part transfer task for other domains as well.

Acknowledgments. This work was supported in part by the National Science Foundation (CAREER IIS-1253549), the Office of Naval Research (N00014-19-1-2655), and by Indiana University through the Emerging Areas of Research Initiative Learning: Brains, Machines and Children.

References

[1] A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei. What's the point: Semantic segmentation with point supervision. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 549–565. Springer, 2016.

[2] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1335–1344, 2016.

[3] H.-S. Fang, G. Lu, X. Fang, J. Xie, Y.-W. Tai, and C. Lu. Weakly and semi supervised human body part parsing via pose-guided knowledge transfer. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[4] S. Gidaris and N. Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375, 2018.

[5] K. Gong, X. Liang, Y. Li, Y. Chen, M. Yang, and L. Lin. Instance-level human parsing via part grouping network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 770–785, 2018.

[6] K. Gong, X. Liang, D. Zhang, X. Shen, and L. Lin. Look into person: Self-supervised structure-sensitive learning and a new benchmark for human parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 932–940, 2017.

[7] A. Khoreva, R. Benenson, J. Hosang, M. Hein, and B. Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 876–885, 2017.

[8] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, 2013.

[9] D. Lin, J. Dai, J. Jia, K. He, and J. Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3159–3167, 2016.

[10] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[11] S. Naha, Q. Xiao, P. Banik, M. Alimoor Reza, and D. J. Crandall. Pose-guided knowledge transfer for object part segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop on Visual Learning with Limited Labels*, pages 906–907, 2020.

[12] X. Nie, J. Feng, and S. Yan. Mutual learning to adapt for joint human parsing and pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 502–517, 2018.

[13] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

[14] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[15] T. Ruan, T. Liu, Z. Huang, Y. Wei, S. Wei, and Y. Zhao. Devil in the details: Towards accurate single and multiple human parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4814–4821, 2019.

[16] B. Russell, A. Torralba, K. Murphy, and W. T. Freeman. Labelme:a database and web-based tool for image annotation. *International Journal on Computer Vision (IJCV)*, 2008.

[17] A. A. Shvets, A. Rakhlin, A. A. Kalinin, and V. I. Iglovikov. Automatic instrument segmentation in robot-assisted surgery using deep learning. In *Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 624–628. IEEE, 2018.

[18] M. Sun and S. Savarese. Articulated part-based model for joint object detection and pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 723–730, 2011.

[19] K. Wang, J. H. Liew, Y. Zou, D. Zhou, and J. Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9197–9206, 2019.

[20] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. L. Yuille. Joint object and part segmentation using deep learned potentials. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1573–1581, 2015.

[21] F. Xia, P. Wang, X. Chen, and A. L. Yuille. Joint multi-person pose estimation and semantic part segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6769–6778, 2017.

[22] F. Xia, J. Zhu, P. Wang, and A. L. Yuille. Pose-guided human parsing by an and/or graph using pose-context features. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[23] Z. Yang, Y. Li, L. Yang, N. Zhang, and J. Luo. Weakly supervised body part parsing with pose based part priors. *arXiv preprint arXiv:1907.13051*, 2019.

[24] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 834–849. Springer, 2014.

[25] X. Zhang, Y. Wei, Y. Yang, and T. Huang. Sg-one: Similarity guidance network for one-shot semantic segmentation. *arXiv preprint arXiv:1810.09091*, 2018.

[26] J. Zhao, J. Li, X. Nie, F. Zhao, Y. Chen, Z. Wang, J. Feng, and S. Yan. Self-supervised neural aggregation networks for human parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 7–15, 2017.