Efficient Fine-grained Classification and Part Localization Using One Compact Network

Xiyang Dai
University of Maryland
College Park, MD
xdai@umiacs.umd.edu

Ben Southall
Nhon Trinh
SRI International
Princeton, NJ
ben.southall, nhon.trinh, bogdan.matei@sri.com

Bogdan Matei

Abstract

Fine-grained classification of objects such as vehicles, natural objects and other classes is an important problem in visual recognition. It is a challenging task because small and localized differences between similar looking objects indicate the specific fine-grained label. At the same time, accurate classification needs to discount spurious changes in appearance caused by occlusions, partial views and proximity to other clutter objects in scenes. A key contributor to fine-grained recognition are discriminative parts and regions of objects. Past work has often attempted to solve the problems of classification and part localization separately resulting in complex models and ad-hoc algorithms, leading to low performance in accuracy and processing time. We propose a novel multi-task deep network architecture that jointly optimizes both localization of parts and fine-grained class labels by learning from training data. The localization and classification sub-networks share most of the weights, yet have dedicated convolutional layers to capture finer level class specific information. We design our model as memory and computational efficient so that can be easily embedded in mobile applications. We demonstrate the effectiveness of our approach through experiments that achieve a new state-of-the-art 93.1% performance on the Stanford Cars-196 dataset, with a significantly smaller multi-task network (30M parameters) and significantly faster testing speed (78 FPS) compared to recent published results.

1. Introduction

Fine-grained classification amongst various classes of man-made and natural objects is currently an active area of research because of numerous practical applications. For instance, recognizing make-models of vehicles can improve tracking of vehicles across non-overlapping camera views, or searching for a given vehicle in a forensic investigation using captured video from multiple locales. Similarly, recognizing types of birds can enable their counting in an area or understanding patterns of migration. While related to generic image classification, fine-grained classification is a significantly distinct problem because it must focus on small, localized intra-class differences (e.g., make-models of vehicles, birds species) instead of inter-class differences that are often easier to account for (e.g., vehicles vs. birds). Accounting for differences between similar looking objects within a class to recognize specific instances, while being robust to spurious changes caused by occlusions and overlap with surrounding objects makes this task very challenging.

An important aspect of the solution to the fine-grained classification is locating discriminative parts of objects. If a learning network can be designed to focus on parts such as the position of car headlights, then it will learn representations that distinguish makes and model of cars based upon different shapes of headlights. Also, it may enable us to solve more challenging tasks, such as "locate a Honda Civic 2006 with a dent on the left front door", since we have an understanding of parts semantics and their relative geometry. However, it is challenging to solve part localization and fine-grained classification simultaneously because the former is geometric in nature while the latter is a labeling problem. Previous work either solved these two problems separately or fused them together in complicated frameworks.

We describe a multi-task deep learning approach to simultaneously solve part localization and fine-grained classification, and demonstrate the mutual benefits of the multi-task approach. Our network architecture (see Figure 1) and training procedure are less complex than competing methods while achieving better results with significant speedup and with smaller memory footprint. In order to reduce the difference in the nature of localization and classification problems, We model part localization as a multi-class classification problem by representing parts as a label mask that annotates part locations, as opposed to the traditional regression of geometric coordinates of parts. This narrows

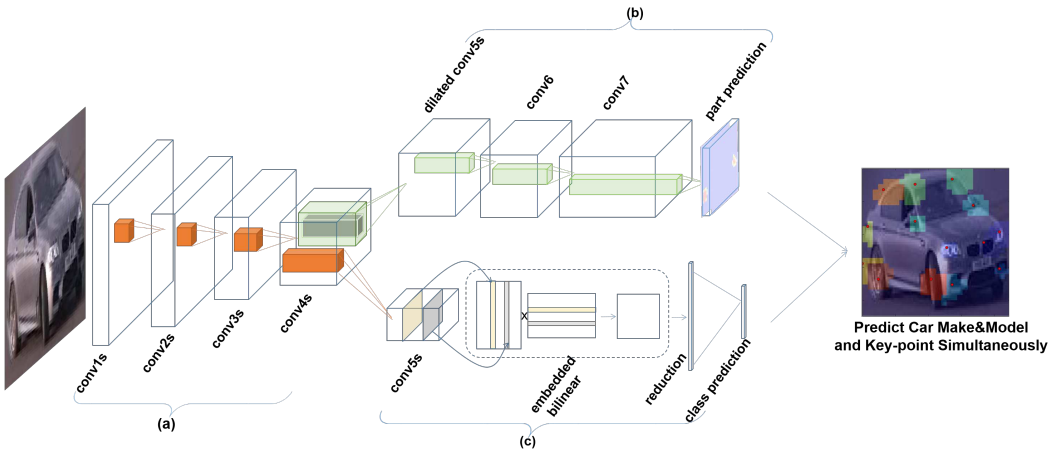

Figure 1: Overview of our deep multi-task learning architecture applied for joint part localization and fine-grained classification. The localization network (b) and fine-grained classification network (c) share the first four levels of convolution (a) and also have task-specific dedicated convolution layers. The whole architecture is trained end-to-end. Our multi-task network can simultaneously predict fine-grained class and part locations.

the gap between part localization and fine-grained classification. It allows us to share a significant number of network parameters between the fine-grained classification and the part localization tasks and enables us to take advantage of pre-trained models as a starting point for optimization. A set of fully-convolutional layers produce the part label mask while a mixture of convolutional and embedded bilinear pooling layers are used for fine-grained classification. The part localization loss and fine-grained classification loss are combined, enabling an end-to-end multi-task data-driven training of all network parameters. Our contributions are three-fold:

- We propose a novel multi-task deep learning and fusion architecture that have both shared and dedicated convolutional layers for simultaneous part labeling and make-model classification.
- The accuracy of our approach is competitive to state-of-the-art methods on both car and bird domains.
- Our network architecture is more compact (30 M parameters) and runs much faster (78 FPS) than competitors, enabling real-time, mobile applications.

Our paper is arranged as follows: Section 2 compares our approach to related work. Section 3 details the design of our deep multi-task architecture. Section 4 shows analysis of experiments comparing our approach with state-of-the-art methods in terms of task performance and computational efficiency. Conclusions are presented in Section 5.

2. Related Work

In this section, we introduce related work from the fields of fine-grained classification and part localization.

Fine-grained classification, using discriminative parts, has been the subject of active research. [3] built a human-in-loop classification game that revealed the importance of discriminative parts for the fine-grained task. [1] first demonstrated the usefulness of part based one-versus-one discriminative features. [9] tried to detect parts using several unsupervised part detectors. Later, [10] improved such work by proposing a co-segmentation based method that can generate discriminative parts without using part annotations. Most recently, [26] showed promising improvements in the use of parts by including geometric constraints between triplets of discriminative parts.

Meanwhile, fine-grained classification performance has seen improvements owing to developments in deep network architectures. [19] showed that simply using off-the-self CNN features could lead to significant improvement over traditional hand-crafted features. [4] combined deep convolutional activation features with a deformable parts model to further improve fine-grained performance. More recently, part information has been used more directly in the training process for deep networks. [2] applied deep convolutional networks to batches of image patches that were located and normalized by pose. Similarly, [28] borrowed the idea of region-based CNN and fine-tuned it on object parts. These two methods showed the great potential of merging part information into deep network models. However, they required ground truth part annotations during testing for good performance. To solve the part localization prob-

lem, [20] proposed a multi-proposal consensus network to predict part locations. Recently, there is a trend toward trying to solve part localization and fine-grained classification simultaneously. [13] proposed a valve linkage function that connected part localization, alignment and classification together. However, this approach attempted only to discriminate the heads of birds from their bodies. [14] proposed a network containing two streams of appearance models combined with a bilinear pooling layer, arguing that manually defined parts were sub-optimal and the bilinear pooling approach could explore optimal parts implicitly. Although the approach benefited from data-driven end-to-end learning of the deep network's parameters, the degree to which parts were discovered by the network was hard to interpret. [11], demonstrated the benefits of training with larger amounts of data (and a larger number of classes) to a fine-grained recognition task, even in the presence of noisy training labels. We consider our work separate from that. The major purpose of this paper is to illustrate the benefits of a compact network that can efficiently perform fine-grained classification and part localization simultaneously. Most recently, [7] proposed a framework that solved localized network and fine-grained classification together, which is most similar to this paper. However, this approach solved part localization and fine-grained classification separately and used a region based feature pooling around the parts to merge the discriminative information from parts. Although it showed the effectiveness of localizing and utilizing of the parts, the framework does not support end-to-end learning. Our approach also tries to solve part localization and fine-grained classification at the same time. However, unlike previous approaches, our model (and its associated training approach) is constructed to explicitly share information between parts localization and fine-grained classification; the architecture itself forces weight-sharing between the two tasks, and our training approach ensures that the part localization and fine-grained classification tasks influence one another. The network and training regime allows us to perform seamless end-to-end training, and makes our approach efficient.

3. Model

We start by introducing the overall architecture of our model to perform fine-grained classification and part localization simultaneously. The detailed analysis of our build components is demonstrated in the further subsections.

3.1. Representation Learning with Deep Multi-task Architecture

Multi-task learning has proven to be effective in several computer vision tasks. Deep networks with end-to-end training are well-suited for multiple tasks because they learn generic representations in early layers prior to specialization in later stages of the network. Recent work [28, 15, 18] has shown promising results by attaching multiple shallow and task-specific output layers to the final fully connected layer, and training the network to minimize combined losses for each task output. These approaches mainly change the last fully connected layers dedicated to specific tasks while not learning any low or mid-level representations that could influence the accuracy of multiple tasks. Our approach differs from previous vanilla multi-task models in: (i) its design of common low and mid-level representation layers for multiple tasks, (ii) application of a careful analysis to select layers that are suitable for sharing between the two tasks, and (iii) designing task-specific deep sub-networks to capture task-specific representations. The end-to-end training for multiple tasks ensures that the task-specific representations benefit from task-specific tuning, while the shared representations are jointly influenced by the needs of both tasks. Such architecture enables joint learning of part localization and fine-grained classification effectively and efficiently.

Choosing a Base Model. Our architecture is based upon a VGG-16 network pre-trained on ImageNet [21]. We choose VGG due to several reasons. First, VGG mainly utilizes 3 by 3 convolutional kernels that can be efficiently computed. Meanwhile, the the layers of a VGG network captured low and mid-level representations are easily interpreted. Since we explicitly rely on learning common representations, VGG serves as an appropriate based model for our fusion architecture. With portability in mind, we skip state-of-the-art Inception and ResNet models on purpose, although such models may further improve the performance.

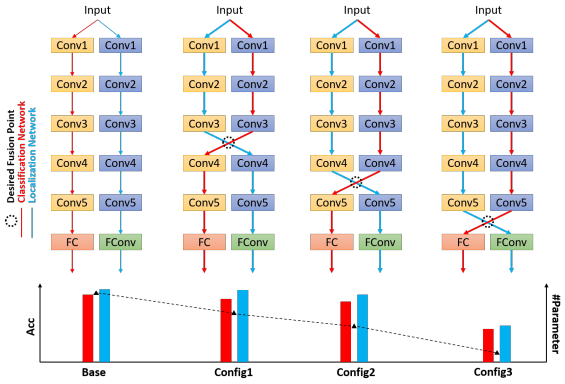

Figure 2: The experimental setups to determine which layers can be shared between tasks. Blue means layers are for localization. Red are trained for classification. We switch different weights between two network in different configurations, and retrain the later layers. This leads us to check the probability of fusion at desired point

Determining Multi-task Structure. Previous approaches [28, 15, 18] implement multi-task learning by

sharing all network parameters up until the final fully-connected (FC) layers; new FC layers are added for each task. Given the representational power of a single FC layer, this strategy has limited effectiveness. Our multi-task architecture uses deep sub-networks for different tasks, sharing earlier convolutional layers. Specifically, our part localization network and our fine-grained classification network architectures (described above) use the same *architecture* (VGG-16) in their first five convolutional layers. We aim to fuse the two networks together such that they use the same *weights* for a number of shared layers; if we share only the earliest layers, we may render ineffective gradient flows from task-specific sub-networks to the fused layers. If we share too many layers, we may degrade performance of the later task-specific sub-networks. To find the appropriate number of layers to share, we conducted a series of experiments. We first of all trained our localization network and classification networks separately to serve as a baseline (shown in Figure 2 as "Base") for measuring performance on the two tasks. Next, we switch the weights of the first three, four and five convolutional layers (shown in Figure 2 as "Config1, Config2, Config3"), and retrain the latter stages of the networks (this experiment is possible owing to the common *architecture* of the first layers of the two task networks) . Swapping weights like this, and then measuring task performance, allows us to establish which weights can be shared across tasks. For example, we found the performance of both task-specific networks drops significantly when we aggressively switch all convolution layers ("Config3"). The performance of both tasks with weights switched as in "Config2" indicated that features learned by the first four convolutional layers in both task-specific networks can be applied effectively to the other task, so our final architecture shares these weights. The whole parameter setup of our deep multi-task architecture is described in Table 1.

Multi-task training. We jointly train our multi-task architecture in an end-to-end fashion using a typical multi-task fusion loss:

$$\mathcal{L}_{fuse} = \mathcal{L}_{loc} + \lambda \mathcal{L}_{cls}\,, \tag{1}$$

where λ is a weight factor to control the influence of each task during joint training. We determine an appropriate value in our experiments below. Our architecture (figure 1) shares layers between the two tasks, the detailed parameter setup is shown in Table 1. We describe the detailed designs of dedicated layers for each tasks in the following sections.

3.2. Localize Parts and Key-points with Pseudo-masking

Key-point (and, equivalently, part) localization has been widely studied for the purpose of pose estimation [17, 22, 16] and has largely been solved by learning a regression layer at the end of a deep network to extract the x, y coordinates of trained key-points. These papers have demonstrated that the regression task is sufficiently different from the image classification task that networks must be trained from scratch, rather than fine-tuned from a pre-trained network; this not only increases the amount of training data required, but also extends training time. Motivated by our desire to share layers between the localization task and the fine-grained classification task, and inspired by recent successes in semantic segmentation, rather than model part localization as a regression problem, we instead model it as a multi-class part classification problem. This decision allows us to fine-tune from a pre-trained classification network and enables weight-sharing with the fine-grained classification network. Our experiments show that this design decision also provides excellent part localization performance.

Architecture. Our parts localization approach is based upon mask generation; to support this we modify the VGG-16 architecture to be fully convolutional in the latter stages such that our output mask has dimensions 28×28. Specifically, we keep the first four convolution layers unchanged, except for dropping spatial pooling after conv4. We modify conv5 to use dilated (or atrous) convolutions, also without downsampling. In addition, we change the fully connected layers into fully convolutional layers with kernels in spatial size of 1×1. These modification allow us to reuse the same pretrained VGG-16 network weights, but output a 28×28 spatial part localization mask instead - similar to semantic segmentation, this mask is a labeled image, with pixels marked as background, or as containing a particular model part, as appropriate. The detailed configuration can be viewed in Table 1(b).

Learning. Given a specific part $p_c \in P = \{p_1, p_2, \cdots, p_K\}$, where c is the part class (such as "Front left light" of car) of total K classes ($K+1$ is the background class label), with normalized spatial coordinate $x \in [0,1], y \in [0,1]$, we want our localization network to generate a $m \times m$ spatial map S that predicts $S_{u,v} = c$, with $u = \lfloor x \cdot m \rfloor$ and $v = \lfloor y \cdot m \rfloor$, where $\lfloor x \rfloor$ is the truncating operator to an integer from a real number. However, considering the the area of part locations on this spatial map will be significant smaller $k \ll m^2$, which causes the learning process highly imbalanced (background vs. key-points ratio will be $\frac{m^2-k}{k} \to 1$), we apply a pseudo-masking around ground-truth part locations to make the learning easier. We define our pseudo masking strategy as

$$M_{i,j} = \begin{cases} K+1, \text{if} \quad \min dist((i,j),(x_c,y_c)) \geq t \\ \underset{c}{\operatorname{argmin}}\, dist((i,j),(x_c,y_c)), \quad \text{otherwise} \end{cases}, \tag{2}$$

where $dist(\cdot)$ is a function that measures the distance; t is a trade-off to control the background/key-point ratio, we use $t = 0.1m$.

(a) Shared layers

	1	2	3	4	5	6	7
layer	2 × conv	max	2 × conv	max	3 × conv	max	3 × conv
filter-stride-padding	3-1-1	2-2-0	3-1-1	2-2-0	3-1-1	2-2-0	3-1-1
#channel	64	64	128	128	256	256	512
activation	relu	idn	relu	idn	relu	idn	relu
size	224×224	112×112	112×112	56×56	56×56	28×28	28×28

(b) Dedicated localization layers

	8a	9a	10a	11a	12a	13a	14a	15a
layer	max	3 × dconv	max	avg	dconv	conv	conv	fc
filter-stride-padding	3-1-1	3-2-1	3-1-1	3-1-1	3-12-1	1-1-0	1-1-0	-
#channel	512	512	512	512	512	1024	#part	1
dilation	-	2	-	-	12	-	-	-
activation	idn	relu	idn	idn	relu	relu	relu	softmax
size	28×28	28×28	28×28	28×28	28×28	28×28	28×28	28×28

(c) Dedicated classification layers

	8b	9b	10b	11b
layer	max	3 × conv	embedding	fc
filter-stride-padding	2-2-0	3-1-1	-	-
#channel	512	512	1	1
activation	idn	relu	idn	softmax
size	14×14	14×14	8192	#class

Table 1: The detail configuration of our final multi-task architecture. The attributes of the column "conv", "max", "avg", "dconv", "embedding", "fc" represent "convolution", "max pooling", "average pooling", "dilated convolution", "feature embedding" and "fully connected layer".

Our loss function is:

$$\mathcal{L}_{loc} = -\sum_{i=1}^{m}\sum_{j=1}^{m} \log \frac{f(i,j,M_{i,j})}{\sum_{c=1}^{K+1} f(i,j,c)}\,, \tag{3}$$

where $f(\cdot)$ represents the network. The loss includes the background class $K+1$.

Inference. Although our localization network predicts a pseudo part map, we can still recover the accurate key-point coordinate by exploring the probability maps underneath. Differing from the approach of [7], we don't need to set up a threshold to decide the existence of specific part location,since it is already handled by our pseudo masking strategy. Given a $m \times m$ prediction map S and a $m \times m \times (K+1)$ probability map $Prob$ extracted from last fully convolutional layer of the part-localization network, the coordinate i_c, j_c of a part c can be inferred by:

$$(i_c, j_c) = \underset{i,j}{\operatorname{argmax}} \left\{ Prob_{i,j,c} \cdot \mathbf{1}_{S_{i,j}=c} \right\}, \tag{4}$$

where $\mathbf{1}_a$ is the indicator function which is 1 if condition a is true and 0 otherwise. Our localization network design has multiple advantages:

- Our localization network shares the same amount of weights as VGG-16, hence can be fine-tuned using existing models
- Our network requires a small 224×224 input size , but yet generates a large part prediction mask
- Most importantly, we model the localization task as a classification problem, which enables straightforward fusion with the fine-grained classification task

3.3. Fine-grained Classification with Feature Embedding

We now describe the detailed implementations for our dedicated fine-grained classification layers.

Embedded bilinear pooling. Bilinear pooling has been proven effective to represent the feature variations from multiple streams of features. In a departure from previous work [14] that utilizes two different networks to conduct bilinear pooling, we demonstrate the effectiveness of using embedded bilinear pooling within a single network. Given a $w \times h \times ch$ shaped feature map F generated from a network, embedded bilinear pooling can be calculated as:

$$\mathsf{E} = \sum_{i=1}^{w}\sum_{j=1}^{h} \mathbf{F}_{i,j} \cdot \mathbf{F}_{i,j}^{\top}\,, \tag{5}$$

where $\mathbf{F}_{i,j}$ is a ch-dimensional column vector. This form is closely related to region covariance [23], that captures second order statistics between features, and can improve the classification performance.

Dimensionality reduction. Embedded bilinear pooling reduces the training parameters from $w \times h \times ch \times l$ to $ch \times ch \times l$, where l is the number of hidden units of fully connected layer for prediction. However, this is still a large number that can overwhelm the whole training process, and may also lead to inefficient utilization of learned weights. We use compact bilinear pooling [6, 5], a projection method that further reduces the number of dimensions in our feature vector while preserving desirable properties of the feature. Given mapping vectors $h \in \mathbb{N}^d$ where each entry is uniformly sampled from $\{1, 2, \cdots, c\}$, and $s \in \{+1, -1\}^d$ where each entry is sampled with either $+1$ or -1 with equal probability, the sketch function is defined as:

$$\Psi(x, s, h) = [C_1, C_2, \cdots, C_c] \tag{6}$$

where

$$C_j = \sum_{i:h(i)=j} s(i) \cdot x(i) \tag{7}$$

To reduce the dimensionality of bilinear features, the $ch \times ch$ size bilinear feature E is first vectorized to $x \in \mathbb{R}^d$ where $d = ch \times ch$ and further projected to a lower c-dimensional vector $\hat{\mathsf{E}} \in \mathbb{R}^c$ by:

$$\hat{\mathsf{E}} = \mathcal{F}^{-1}(\mathcal{F}(\Psi(x, s, h)) \circ \mathcal{F}(\Psi(x, s', h'))) \tag{8}$$

where s' and h' are drawn similarly to s and h, $\circ$ operator represents element-wise multiplication, and $\mathcal{F}$ represents the Fast Fourier Transformation. The result of the tensor sketching process is a lower-dimensional version of E, $\hat{\mathsf{E}}$; the number of dimensions in E and $\hat{\mathsf{E}}$ used in our experiments are detailed in Table 1(c).

Classification loss. Finally, the reduced features $\hat{\mathsf{E}}$ can be mapped to our C fine-grained classes using a small fully connected layer $fc(\cdot)$, trained using multinomial logistic loss:

$$\mathcal{L}_{cls} = -\sum_{i=1}^{C} \log \frac{fc(\hat{\mathsf{E}}, i)}{\sum_{c=1}^{C} fc(\hat{\mathsf{E}}, c)} . \tag{9}$$

Our classification network design has multiple advantages:

- We greatly reduce the number of parameters by replacing fully connected layers with an embedded bilinear layer.
- We are able to explore the second order information between feature maps through our embedded bilinear layer.
- We further reduce number of parameters required by introducing a random mapping technique.

4. Experiments

We first compare the performance of our method on the fine-grained classification task versus other leading methods considering both the performance and efficiency (speed and memory usage). Next, we demonstrate the effectiveness of part localization. We do an ablation study to illustrate that multi-task architecture improves classification performance at the end.

4.1. Dataset and Implementation Details

Dataset. We evaluate our approach on two standard fine-grained benchmarks. The Stanford Cars-196 [12] dataset contains 196 classes of car categories described by make, model and year, and has a total of 16185 images. This dataset is challenging due to: the large variation of car model, pose, and color; and often minor differences between models. We use the provided car bounding boxes during training and testing. This dataset does not provide information about parts, hence we manually annotated 30 images per class with 18 parts, such as "front right light", "rear left bumper" and so on. Our second dataset is Caltech-UCSD Birds (CUB-200-2011) [24], which contains 200 bird spices with 11788 images captured in the wild. Each image is annotated with a bounding box and 15 body parts such as "head", "breast", etc. This dataset is used as a cross-domain reference that shows our approach is easily applied to tasks from quite different domains.

Implementation. We implement our proposed multi-task network using a customized Caffe [8] package. Each input image is cropped to the object's bounding box and then resized to 224×224. We adopt a 3-step process to speed up the training process: 1) we freeze the weights of the part localization sub-network and fine-tune the classification network (including layers shared with the localization task); 2) we freeze the classification network weights and fine-tune the localization network (including the shared weights); 3) We fine-tune the whole network together with small learning rate and high momentum. The training approach is a gradual specialization of the base VGG-16 weights by incrementally adapting the network to new tasks. Step 1 adapts the classification network to the new image domain. Step 2, assisted by adapation of the shared layers in step 1, adapts the part-localization sub-network to domain-specific parts. Step 3 tunes the entire network via the multi-task loss, enabling task-to-task feedback via the shared layers.

4.2. Efficiency and Performance

We compare our method with several recent alternative methods [13, 29, 10, 14, 7] that aimed to improve fine-grained classification problem by solving part localization. Each method is evaluated with speed and memory usage on

Method	Setup	Input Resolution	#Parameters	Speed	Accuracy
Deep-LAC[13]	Parts	227	–	–	80.3
Part-RCNN[29]	BBox + Parts	227	60M	<1 FPS	76.4
Parts[10]	BBox	224	135M	<1 FPS	82.8
Bilinear[D,M][14]	BBox	448	70M	8 FPS	85.1
Stacked-CNN[7]	BBox + Parts	448	115M	20 FPS	76.7
Ours	**BBox + Parts**	**224**	**30M**	**78 FPS**	**84.3**

Table 2: Comparison with state-of-art methods on efficiency. The number of parameters estimation is calculated as Caffe weight size.

Parts	throat	beak	crown	forehead	right eye	nape	left eye	back
Stacked-CNN[7]	0.908	0.894	0.894	0.885	0.861	0.857	0.850	0.807
Ours	**0.963**	**0.952**	**0.950**	**0.960**	**0.939**	**0.937**	**0.943**	**0.867**
Parts	breast	belly	right leg	tail	left leg	right wing	left wing	Overall
Stacked-CNN[7]	0.799	0.794	0.775	0.760	0.750	0.678	0.670	0.866
Ours	**0.877**	**0.858**	**0.752**	**0.841**	**0.740**	**0.775**	**0.755**	**0.874**

Table 3: APK comparison with state-of-art methods on the CUB_200_2011.

a few key factors: input resolution, setup, number of training parameters, testing speed and accuracy, as shown in Table 2. All the experiments were conducted on a single Titan X (Maxwell). Our implementation runs significantly faster than all other methods, achieving about four times speed up compared to the previous fastest method. The efficiency gain mainly comes from the benefits of effective network fusion. Meanwhile, our classification performance is still competitive to state-of-the-art methods.

We further evaluate our method on Stanford Car196 dataset with the state-of-art methods, such as [12, 25, 9, 27, 14, 26, 10] on the Cars-196 dataset. Table 4 shows results; our multi-task approach achieves 93.1% top-1 accuracy and is better compared to the previous best [10] with a 0.3% improvement. Considering the minor improvement published recently, we think our improvement is noticeable. In addition, our approach is much more computationally efficient and solve part localization at the same time.

Method	Accuracy
BB-3D-G[12]	67.6
LLC[25]	69.5
ELLF[9]	73.9
CNN Finetuned[27]	83.1
FT-HAR-CNN[27]	86.3
Bilinear[D,M] [14]	91.7
BoT[26]	92.5
Parts[10]	92.8
Ours	**93.1**

Table 4: Comparison with state-of-art methods on the Stanford Car-196.

We also evaluate our part localization performance using the APK (average precision of key points) metric. This metric considers a predicted key-point (part location) to be correct if it lies in a radius of $\alpha\times$(image size) around the ground-truth location. We use $\alpha = 0.1$ following [7] to compare the results on CUB_200_2011. As shown in Table 3, the localization performance is significantly better than the previous approach [7] in all part categories except "right leg" and "left leg". Qualitative part localization results are shown in Figure 3. Our approach is capable of precisely pinpointing parts across a range of pose and aspect variations for both birds and cars.

4.3. Ablation Study

We first analyze the influence of the parameter λ that controls the balance between part localization and classification loss during training, and then illustrate the positive influence of multi-task training on the classification task. Figure 4 shows the evolution of test loss during training for both classification and part localization, with $\lambda = \{0.1, 0.2, 0.5, 1\}$ and using a small fixed learning rate over 70 epochs. Our experiments indicate that as λ increases, the gradient flow from the part localization sub-network overwhelms the training of the classification sub-network. Our best performance is with $\lambda = 0.2$.

We now evaluate the effectiveness of our multi-task architecture. We use a fine-tuned VGG-16/ImageNet as a baseline model, which archives a reasonable performance on both the CUB_200_2011 and Stanford Cars-196 datasets. In order to compare the performance before and after multi-task training, we disable the localization sub-network during training to form a classification standalone result. As shown in Table 5, our multi-task architecture performs better than the baseline and standalone-trained model on both

(a) (b)

Figure 3: The visualization selected results of key-point localizations on both Stanford Car-196 (a) and CUB-200-2011 (b). Solid dots represent the key-point location we predict. The transparent mask presents the part map we predict.

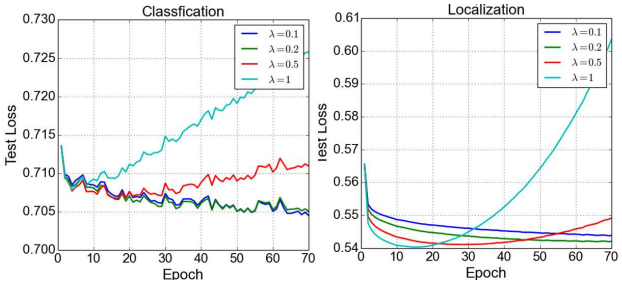

Figure 4: The influence of different λ value chosen on the multi-task training.

datasets. This demonstrates the effectiveness of our multi-task training.

Base	Class	Part	Stanford Car196	CUB_200_2011
✓			89.1	80.0
✓	✓		92.7	83.9
✓	✓	✓	93.1	84.3

Table 5: Evaluating the effectiveness of the multi-task training on datasets .

5. Conclusion and Future Work

In this paper, we propose a compact multi-task architecture that simultaneously performs part localization and fine-grained classification. We are able to fuse the localization network and classification network effectively and efficiently. Experiments demonstrate that our approach is general, being competitive on both the Stanford Cars-196 and the Cub-200-2011 birds datasets. Furthermore, our proposed network is both significantly smaller and faster than previous state-of-the-art methods, which makes real-time mobile applications possible.

With the success of such compact multi-task architecture, our future work will mainly focus on applying this architecture on car specific tasks, such as car re-identification and searching. Enhancement of the techniques presented in this paper will help solve multiple problems such as crowd-sourcing car re-identification problem with wearable devices and specific instance recognition problems using features such as a ”dent” in a specific location.

Acknowledgments

The authors wish to thank the Combating Terrorism Technical Support Office (CTTSO) for providing the financial support to this project.

References

[1] T. Berg and P. N. Belhumeur. POOF: Part-Based One-vs-One Features for fine-grained categorization, face verification, and attribute estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013. 2

[2] S. Branson, G. V. Horn, S. Belongie, and P. Perona. Bird species categorization using pose normalized deep convolutional nets. In *British Machine Vision Conference (BMVC)*, Nottingham, 2014. 2

[3] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013. 2

[4] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference in Machine Learning (ICML)*, 2014. 2

[5] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *CoRR*, abs/1606.01847, 2016. 6

[6] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell. Compact bilinear pooling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6

[7] S. Huang, Z. Xu, D. Tao, and Y. Zhang. Part-stacked CNN for fine-grained visual categorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3, 5, 6, 7

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 6

[9] J. Krause, T. Gebru, J. Deng, L.-J. Li, and L. Fei-Fei. Learning features and parts for fine-grained recognition. In *International Conference on Pattern Recognition (ICPR)*, Stockholm, Sweden, August 2014. 2, 7

[10] J. Krause, H. Jin, J. Yang, and L. Fei-Fei. Fine-grained recognition without part annotations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 2, 6, 7

[11] J. Krause, B. Sapp, A. Howard, H. Zhou, A. Toshev, T. Duerig, J. Philbin, and F. Li. The unreasonable effectiveness of noisy data for fine-grained recognition. *CoRR*, abs/1511.06789, 2015. 3

[12] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 6, 7

[13] D. Lin, X. Shen, C. Lu, and J. Jia. Deep lac: Deep localization, alignment and classification for fine-grained recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1666–1674, June 2015. 3, 6, 7

[14] T.-Y. Lin, A. RoyChowdhury, and S. Maji. Bilinear cnns for fine-grained visual recognition. In *International Conference on Computer Vision (ICCV)*, 2015. 3, 5, 6, 7

[15] X. Liu, J. Gao, X. He, L. Deng, K. Duh, and Y.-Y. Wang. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015. 3

[16] T. Pfister. *Advancing Human Pose and Gesture Recognition*. PhD thesis, University of Oxford, 2015. 4

[17] T. Pfister, K. Simonyan, J. Charles, and A. Zisserman. Deep convolutional neural networks for efficient pose estimation in gesture videos. In *Asian Conference on Computer Vision (ACCV)*, 2014. 4

[18] R. Ranjan, V. M. Patel, and R. Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *CoRR*, abs/1603.01249, 2016. 3

[19] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 512–519, Washington, DC, USA, 2014. IEEE Computer Society. 2

[20] K. J. Shih, A. Mallya, S. Singh, and D. Hoiem. Part localization using multi-proposal consensus for fine-grained categorization. In *British Machine Vision Conference (BMVC)*, 2015. 3

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 3

[22] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2014, Columbus, OH, USA, June 23-28, 2014*, pages 1653–1660, 2014. 4

[23] O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *9th European Conference on Computer Vision (ECCV), Graz, Austria, May*, pages 589–600, 2006. 5

[24] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011. 6

[25] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 7

[26] Y. Wang, J. Choi, V. I. Morariu, and L. S. Davis. Mining discriminative triplets of patches for fine-grained classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 7

[27] S. Xie, T. Yang, X. Wang, and Y. Lin. Hyper-class augmented and regularized deep learning for fine-grained image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2645–2654, June 2015. 7

[28] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based rcnn for fine-grained detection. In *13th European Conference on Computer Vision (ECCV)*, 2014. 2, 3

[29] Z. Zhang, P. Luo, C. C. Loy, and X. Tang. Facial landmark detection by deep multi-task learning. In *13th European Conference on Computer Vision (ECCV)*, 2014. 6, 7