Detecting Semantic Parts on Partially Occluded Objects

Jianyu Wang*[1]
wjyouch@gmail.com
Cihang Xie*[2]
cihangxie306@gmail.com
Zhishuai Zhang*[2]
zhshuai.zhang@gmail.com
Jun Zhu[1]
zhujun.sjtu@gmail.com
Lingxi Xie(✉)[2]
198808xc@gmail.com
Alan L. Yuille[2]
alan.l.yuille@gmail.com

[1] Baidu Research (USA),
Sunnyvale, CA 94089 USA

[2] Department of Computer Science,
The Johns Hopkins University,
Baltimore, MD 21218 USA

* This work was done when the first author was a Ph.D. student at UCLA. The first three authors contributed equally to this work.

Abstract

In this paper, we address the task of detecting semantic parts on partially occluded objects. We consider a scenario where the model is trained using non-occluded images but tested on occluded images. The motivation is that there are infinite number of occlusion patterns in real world, which cannot be fully covered in the training data. So the models should be inherently robust and adaptive to occlusions instead of fitting / learning the occlusion patterns in the training data. Our approach detects semantic parts by accumulating the confidence of local visual cues. Specifically, the method uses a simple voting method, based on log-likelihood ratio tests and spatial constraints, to combine the evidence of local cues. These cues are called *visual concepts*, which are derived by clustering the internal states of deep networks. We evaluate our voting scheme on the VehicleSemanticPart dataset with dense part annotations. We randomly place two, three or four irrelevant objects onto the target object to generate testing images with various occlusions. Experiments show that our algorithm outperforms several competitors in semantic part detection when occlusions are present.

1 Introduction

"The absence of evidence is not the evidence of absence." — A PROVERB

Deep neural networks [16] have been successful on a wide range of vision tasks and in particular on object detection [11][24]. There have, however, been much fewer studies on semantic part detection. Here, a *semantic part* refers to a fraction of an object which can be verbally described, like a wheel of a car or a wing of an airplane. Detecting semantic parts of objects is a very important task, which enables us to parse the object and reason about its

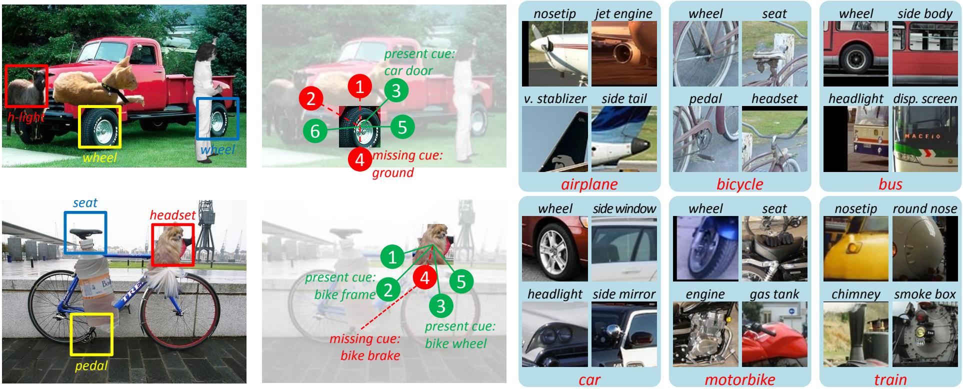

Figure 1: Left: our goal is to detect semantic parts, in particular under occlusion. Red, blue and yellow boxes indicate fully-, partially- and non-occluded semantic parts respectively. The proposed voting method is able to switch on/off visual cues (green/red disks) for semantic parts detection. Right: typical semantic parts on six types of rigid objects from VehicleSemanticPart dataset [29]. Some semantic parts (*e.g.*, *wheel*) can appear in different classes, while some others (*e.g.*, *chimney*) only appear in one class. This figure is best viewed in color.

properties. More importantly, humans are able to recognize occluded objects by looking at parts, *e.g.*, simply seeing a *car wheel* is often enough to infer the presence of the entire *car*.

In this paper we study the problem of detecting the semantic parts of partially occluded objects. There are in general two strategies for occlusion handling: making the model inherently robust to occlusions or fitting the model to occlusion patterns in the training set. We argue that the latter strategy is inferior because there are infinite number of occlusion patterns in real world, so the training set only contains a biased subset of occlusion patterns. Therefore, we consider the scenario, where the part detector is trained using non-occluded images but tested on occluded images. In other words, the distribution of testing images are very different from that of training images. This problem setting favors learning the models which are naturally robust and adaptive to occlusions instead of over-fitting the occlusion patterns in the training data.

Figure 1 illustrates the task we are going to address and some typical examples during testing. Since some of the target semantic parts are partially occluded, the state-of-the-art holistic object detectors such as Faster-RCNN [24] are sometimes unable to provide satisfying results due to their lack of ability to deal with occlusions. When the testing image contains an occlusion pattern which does not appear in the training set, such detectors can fail in finding a proper object proposal and/or making classification based on the detected region-of-interest. For example, if a *car wheel* is occluded by a large *table*, there might be no proposal covering the *wheel* (low recall in objectness detection), and the classifier may also be confused even if a perfect proposal is given (low accuracy in object recognition). This inspires us to detect parts by accumulating local visual cues, instead of directly learning a holistic template as what Faster-RCNN [24] is essentially doing.

We start with the recent work [29] which showed that deep networks have internal representations, which are called *visual concepts*. They are related to semantic parts and can be used for part detection. In this paper, we show that visual concepts can be combined together to detect semantic parts. We design a novel voting scheme, which is built upon some simple techniques such as log-likelihood ratio tests and spatial pooling. In the training phase on non-occluded objects, we find the relationship between each semantic part and its supporting visual concepts, and take into account their relative positions to model the spatial contexts. In the testing phase on occluded objects, these clues are integrated together to detect partially or even fully occluded semantic parts. As shown in Figure 1, our voting algorithm is adaptive to different contexts, since it enjoys the flexibility of switching on/off visual cues and avoids using negative cues.

We evaluate our algorithm on VehicleSemanticPart dataset [29] (see Figure 1), which provides dense labeling of more than 100 semantic parts over 6 object classes. In order to create the test set with various occlusion patterns, we randomly superimpose two, three or four irrelevant objects (named occluders) onto the target object. These occluders are manually labeled object segments from the PASCAL-Parts dataset [6]. We also control the occlusion ratio by computing the fraction of occluded pixels on the target object. Experiments reveal the advantage over several competitors in detection accuracy (measured by mean AP) under the scenario where the target object is partially occluded. Our approach, while being able to deal with occlusions, does not need to be trained on occluded images.

This paper is organized as follows. Section 2 discusses related work. Our voting algorithm is described in Section 3. Section 4 describes the experiments which validate our approach, and Section 5 concludes this work.

2 Related Work

Object detection is a fundamental task in computer vision. As the fast development of deep neural networks [16][26][13], this field has been recently dominated by one type of pipeline [11][24], which first generates a set of object proposals [1][27], and then predicts the object class of each proposal. This framework has significantly outperformed conventional approaches, which are based on handcrafted features [7] and deformable part models [9][3].

Visual concepts [29] are obtained by clustering the intermediate neural responses of deep networks. It is shown that on rigid objects, image patches corresponding to the same visual concept are often visually similar, and that visual concepts are fairly effective in detecting keypoints in the PASCAL3D+ dataset [30]. Our studies are built on previous studies which showed that filters in deep networks often exhibited preferences in stimuli [33].

There are some works about detecting parts using deep networks. In [4], they used deep features as unary term and built graphical models to assemble parts into entire human. [32] applied R-CNN [11] to detecting parts, which were later used for fine-grained categorization. [21] used deep feature with SVM for keypoint detection. By contrast, some works used CNN features to discover mid-level visual elements in an unsupervised way, which are then used for object / scene classification [20][25][31].

Occlusion is a common difficulty in object detection [14] or segmentation [19]. [18] used And-or-Graph (AOG) to model the occlusion patterns for car detection. Part-based models is very useful for detecting partially occluded objects [5][18]. Our method is also part-based but applied to detecting semantic parts.

Our voting method is similar to [17][22][23]. But, we incorporate log-likelihood ratio test [2] and spatial constraint [12] as key components into the voting method, which is new. Also we address a new problem of detecting semantic parts under occlusions, where the training images and testing images are quite different.

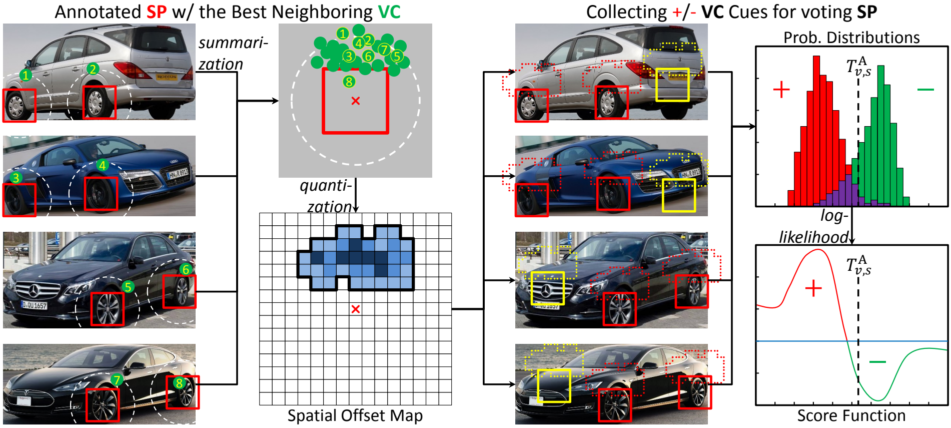

Figure 2: Illustration of the training phase of one $(\mathrm{VC}_v, \mathrm{SP}_s)$ pair. Only one negative point is shown in each image (marked in a yellow frame in the third column), though the negative set is several times larger than the positive set. Each green dot indicates an offset $\Delta p^{\star}$ (see Section 3.2.1), and the spatial offset map contains the frequencies of these offsets. The automatic determination of negative points, the probability distributions and the score functions are described in Section 3.2.2.

3 Our Algorithm

3.1 Notations

We first introduce some notations. Each semantic part SP_s has an index $s \in \{1, 2, \ldots, |\mathcal{S}|\}$, where $\mathcal{S}$ is a pre-defined set of all semantic parts. Let q denote a position at the input image lattice, *i.e.*, $q \in \mathcal{L}_0$. Denote a position at the *pool-4* layer as $p \in \mathcal{L}_4$, then a feature vector can be written as $\mathbf{f}(\mathbf{I}_p) \in \mathbb{R}^{512}$ (*e.g.*, VGG-16). Most often, we need to consider the relationship between two positions on two layers $p \in \mathcal{L}_4$ and $q \in \mathcal{L}_0$. Let $\mathcal{L}_0(p)$ denote the exact mapping from $\mathcal{L}_4$ to $\mathcal{L}_0$. Inversely, let $\mathcal{L}_4(q) = \arg\min_p \{\mathrm{Dist}(q, \mathcal{L}_0(p))\}$ denote the closest position at the $\mathcal{L}_4$ layer grid that corresponds to q. We denote the neighborhood of $q \in \mathcal{L}_0$ on the $\mathcal{L}_4$ layer as $\mathcal{N}(q) \subset \mathcal{L}_4$, which is defined as $\mathcal{N}(q) = \{p \in \mathcal{L}_4 \mid \mathrm{Dist}(q, \mathcal{L}_0(p)) < \gamma_{\mathrm{th}}\}$. The neighborhood threshold γ_{th} is set to be 120 pixels and will be discussed later in Section 4.2.

Following [29], we extract *pool-4* layer features using VGG-16 [26] from a set of training images, and use K-Means to cluster them into a set $\mathcal{V}$ of visual concepts, which corresponds to certain types of visual cues that appear on the image. Here, each visual concept VC_v has an index $v \in \{1, 2, \ldots, |\mathcal{V}|\}$. The v-th clustering center is denoted as $\mathbf{f}_v \in \mathbb{R}^{512}$.

Our algorithm is composed of a training phase and a testing phase, detailed in the following subsections. The training phase is illustrated in Figure 2. We perform training and testing on each semantic part individually.

3.2 The Training Phase

The training phase starts with cropping each object according to the ground-truth object bounding box, and rescaling it so that the short edge contains 224 pixels.

3.2.1 Spatial Relationship between Visual Concepts and Semantic Parts

Our goal is to use visual concepts to detect semantic parts. Therefore, it is important to model the spatial relationship of each $(\mathrm{VC}_v,\mathrm{SP}_s)$ pair. Intuitively, a semantic part can be located via its neighboring visual concepts. If a visual concept VC_v serves as a good visual cue to locate a semantic part SP_s, then the SP_s may only appear at some specified positions relative to VC_v. For example, if VC_v represents the upper part of a *wheel*, we shall expect the semantic part (*wheel*) to appear slightly below the position of VC_v. Motivated by this, we define a *spatial offset map* for each $(\mathrm{VC}_v,\mathrm{SP}_s)$ pair. An offset map $\mathcal{H}_{v,s}$ is a set of frequencies of offsets Δp, indicating the most likely spatial relationship between VC_v and SP_s, *i.e.*, if VC_v at position $p \in \mathcal{L}_4$ supports SP_s, SP_s may appear around $\mathcal{L}_0(p+\Delta p)$ in the image lattice. Since the neighborhood threshold γ_{th} is 120 pixels and the spatial stride of $\mathcal{L}_4$ is 16 pixels, $\mathcal{H}_{v,s}$ is a subset of $\{-7,-6,\ldots,7\}\times\{-7,-6,\ldots,7\}$, or equivalently, a 15×15 grid.

To estimate the offset map $\mathcal{H}_{v,s}$, we perform spatial statistics for each $(\mathrm{VC}_v,\mathrm{SP}_s)$ pair. We find each annotated ground-truth position q, and compute the position $p^\star$ in its $\mathcal{L}_4$ neighborhood which best stimulate VC_v, *i.e.*, $p^\star = \arg\min_{p\in\mathcal{N}(q)} \left\|\mathbf{f}(\mathbf{I}_p)-\mathbf{f}_v\right\|$. Then $\Delta p^\star = \mathcal{L}_4(q)-p^\star$ is added to a score table. After all annotated semantic parts are considered, each offset in the score table gets a frequency $\mathrm{Fr}(\Delta p)\in[0,1]$. The offsets with above-average frequencies compose the offset map $\mathcal{H}_{v,s}$. We rewrite a neighborhood $\mathcal{N}(q)$ equipped with the offset map $(\mathrm{VC}_v,\mathrm{SP}_s)$ as $\mathcal{N}_{v,s}(q)$, which contains all positions $\{p+\Delta p\in\mathcal{N}(q)\mid\Delta p\in\mathcal{H}_{v,s}\}$.

Some typical offset maps are shown in Figure 3. We can see that the concentration ratio of an offset map can reflect, at least to some extent, whether a visual concept is good for supporting or detecting the specified semantic part. In the next step, we shall integrate these spatial cues to obtain a score function.

3.2.2 Probabilistic Distributions, Supporting Visual Concepts and Log-likelihoods

We quantify the evidence that a visual concept $\mathrm{VC}_v\in\mathcal{V}$ gives for detecting a semantic part $\mathrm{SP}_s\in\mathcal{S}$ and also its ability to localize SP_s. We study this for all possible pairs $(\mathrm{VC}_v,\mathrm{SP}_s)$. We find, not surprisingly, that a subset of visual concepts are helpful for detecting a semantic part while others are not. We judge the quality of VC_v in detecting SP_s by measuring its ability to distinguish positive and negative visual cues. This is done by estimating the distribution of Euclidean distance between VC_v and SP_s.

For each SP_s, we select a positive training set $\mathcal{T}_s^+$ composing of those annotated positions q, and a negative set $\mathcal{T}_s^-$ composing of a number of positions which are far away from any ground-truth positions. For each VC_v, we perform statistics on the positive and negative samples, based on the previously defined neighborhoods $\mathcal{N}_{v,s}(q)$, and compute the following conditional distributions:

$$\mathrm{F}_{v,s}^{+}(r)=\frac{\mathrm{d}}{\mathrm{d}r}\Pr\left[\min_{p\in\mathcal{N}_{v,s}(q)}\left\|\mathbf{f}(\mathbf{I}_p)-\mathbf{f}_v\right\|\leqslant r\mid q\in\mathcal{T}_s^{+}\right], \tag{1}$$

$$\mathrm{F}_{v,s}^{-}(r)=\frac{\mathrm{d}}{\mathrm{d}r}\Pr\left[\min_{p\in\mathcal{N}_{v,s}(q)}\left\|\mathbf{f}(\mathbf{I}_p)-\mathbf{f}_v\right\|\leqslant r\mid q\in\mathcal{T}_s^{-}\right]. \tag{2}$$

Here, the first distribution $\mathrm{F}_{v,s}^{+}(r)$ is the *target* distribution, giving the activation pattern for VC_v if there is a semantic part SP_s nearby. The intuition is that if $(\mathrm{VC}_v,\mathrm{SP}_s)$ is a good pair, then the probability $\mathrm{F}_{v,s}^{+}(r)$ will be peaked close to $r=0$, (*i.e.*, there will be some feature vectors within $\mathcal{N}_{v,s}(q)$ that cause VC_v to activate). The second distribution, $\mathrm{F}_{v,s}^{-}(r)$ is the *reference* distribution which specifies the response of the feature vector if the semantic part is not present. This is needed [15] to quantify the chance that we get a good match (*i.e.*, a small value of r) when the semantic part is not present. In practice, we model the distribution using a histogram. Some typical feature distributions are shown in Figure 3. Note that the overlap between the target and reference distributions largely reflects the quality of a $(\mathrm{VC}_v, \mathrm{SP}_s)$ pair.

With the probabilistic distributions, we can find the set of supporting visual concepts $\mathcal{V}_s \subseteq \mathcal{V}$ for each semantic part SP_s. This is determined by first computing the threshold $T_{v,s}^{\mathrm{A}}$ that makes the false-negative rate $\mathrm{FNR}_{v,s} = 5\%$. This threshold will be used in the testing phase to judge if VC_v fires at some grid position for SP_s. Note that we set a small FNR so that the positive samples are mostly preserved. Although this may introduce some false positives, the voting model allows us to filter them out at a higher level. Then, the top-K visual concepts with the minimum false-positive rates $\mathrm{FPR}_{v,s}$ are selected to support SP_s. We fix $K = 45$, *i.e.*, $|\mathcal{V}_s| = 45$ for all $s = 1, 2, \ldots, |\mathcal{S}|$. We set a relatively large K (in comparison to $N \approx 200$), so that when some of the supporting visual concepts are absent because of occlusion, it is still possible to detect the semantic part via the present ones.

Not all supporting visual concepts are equally good. We use the *log-likelihood ratio test* [2] to define a *score function*:

$$\mathrm{Score}_{v,s}(r) = \log \frac{\mathrm{F}_{v,s}^{+}(r) + \varepsilon}{\mathrm{F}_{v,s}^{-}(r) + \varepsilon}. \tag{3}$$

Here, $\varepsilon = 10^{-7}$ is a small floating point number to avoid invalid arithmetic operations. The score function determines the evidence (either positive or negative) with respect to the feature distance, *i.e.*, $r = \left\|\mathbf{f}(\mathbf{I}_p) - \mathbf{f}_v\right\|$. The visualization of these scores are shown in Figure 3.

In summary, the following information is learned in the training process. For each semantic part SP_s, a set of supporting visual concepts is learned. For each $(\mathrm{VC}_v, \mathrm{SP}_s)$ pair, we obtain the voting offset map $\mathcal{H}_{v,s}$, the activation threshold $T_{v,s}^{\mathrm{A}}$, and the score function $\mathrm{Score}_{v,s}(r)$. These will be used in the testing stage.

3.3 The Testing Phase

Now, given a testing image $\mathbf{I}$, our goal is to detect all semantic parts on it. As in the training phase, each semantic part SP_s is processed individually. Recall that we have obtained the set of supporting visual concepts $\mathcal{V}_s$. For each supporting VC_v, an individual voting process is performed on the entire image.

The voting process starts with extracting CNN features on the *pool-4* layer. Let $\mathbf{f}(\mathbf{I}_p)$ be a feature vector at a position $p \in \mathcal{L}_4$. We test if this feature vector activates VC_v by checking if the feature distance $\left\|\mathbf{f}(\mathbf{I}_p) - \mathbf{f}_v\right\|$ is smaller than the activation threshold $T_{v,s}^{\mathrm{A}}$ obtained in the training process. If it is activated, we make use of the score function $\mathrm{Score}_{v,s}(r)$, and substitute with $r_v(\mathbf{I}_p) = \left\|\mathbf{f}(\mathbf{I}_p) - \mathbf{f}_v\right\|$. This term, $\mathrm{Score}_{v,s}(r_v(\mathbf{I}_p))$, or $\mathrm{Score}_{v,s}(\mathbf{I}_p)$ for short, is the evidence that $\mathbf{f}(\mathbf{I}_p)$ votes for SP_s. It is added to various positions at the *pool-4* layer determined by the offset map $\mathcal{H}_{v,s}$. Besides, recall that $\mathcal{H}_{v,s}$ is a set of offset vectors, each of them, denoted as Δp, is equipped with a frequency $\mathrm{Fr}(\Delta p)$. The final score which is added to the position $p + \Delta p$ is computed as:

$$\mathrm{Vote}_{v,s}(p + \Delta p) = (1 - \beta)\,\mathrm{Score}_{v,s}(\mathbf{I}_p) + \beta \log \frac{\mathrm{Fr}(\Delta p)}{U}. \tag{4}$$

The first term ensures that there is high evidence of VC_v firing, and the second term acts as the spatial penalty ensuring that this VC_v fires on the right position specified by the offset

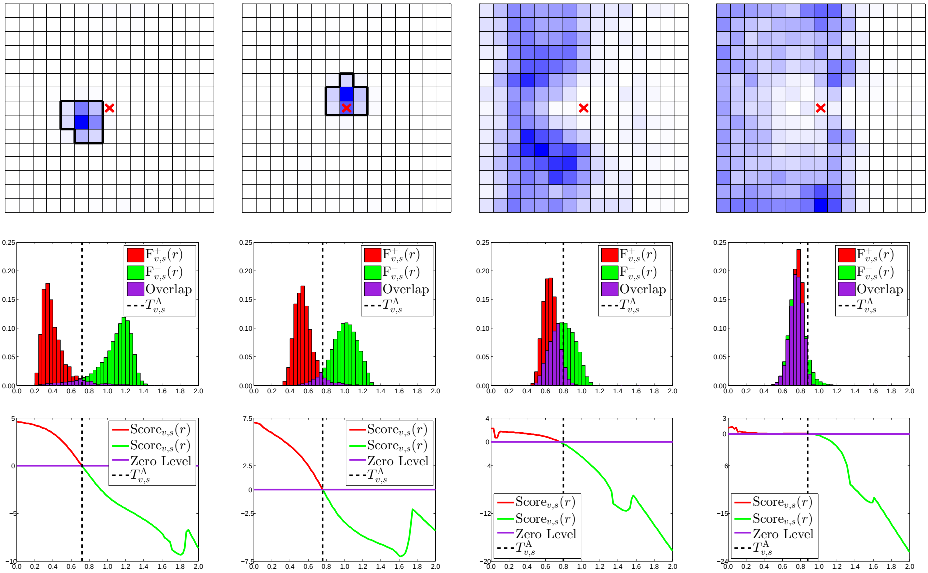

Figure 3: The visual cues obtained in the training process (best viewed in color PDF). From left to right, we visualize the results of using VC_1, VC_8, VC_{60} and VC_{166} to support SP_1. Of these, VC_1 and VC_8 are good supporters, VC_{60} is moderate, while VC_{166} is poor. Top row: the voting offset maps $\mathcal{H}_{v,s}$. Middle row: the target distributions $\mathrm{F}^{+}_{v,s}(r)$ (in red and purple) and the reference distributions $\mathrm{F}^{-}_{v,s}(r)$ (in green and purple). The overlap of these distributions reflects whether this VC is good at detecting this SP (smaller overlap is better). Bottom row: the score functions $\mathrm{Score}_{v,s}(r)$ (red and green parts indicate positive and negative scores, respectively).

map $\mathcal{H}_{v,s}$. Here we set $\beta = 0.7$, and define $\log \frac{\mathrm{Fr}(\Delta p)}{U} = -\infty$ when $\frac{\mathrm{Fr}(\Delta p)}{U} = 0$. U is a constant which is the average frequency over the entire offset map $\mathcal{H}_{v,s}$.

After all activated positions of VC_v are considered, we combine the voting results by preserving the maximal response at each position p. If the maximal response is negative at one position, it is set to 0 to avoid introducing negative cues, *i.e.*, that a visual concept is allowed to support a semantic part, but not allowed to inhibit it. The final score for detecting SP_s involves summing up the voting results of all supporting visual concepts:

$$\mathrm{Score}_s(p) = \sum_{\mathrm{VC}_v \in \mathcal{V}_s} \max\left\{0, \mathrm{Vote}_{v,s}(p)\right\}. \tag{5}$$

This score map is computed at the *pool-4* layer. It is then resized to the original image size using 2D spline interpolation.

3.3.1 The Multi-Scale Testing Strategy

Note that the training process is performed with the object bounding boxes provided, which limits our algorithm's ability of detecting a semantic part at a different scale from the training case. To deal with this issue, we design a multi-scale voting scheme.

This scheme is only used in testing, *i.e.*, no extra training is required. Given a testing image, we resize it to 10 different scales, with the short edge containing 224, 272, 320,

Object	Natural Detection				Oracle Detection			Scale Pred. Loss		
	VC	**SV**	**FR**	**VT**	**VC**	**SV**	**VT**	**VC**	**SV**	**VT**
airplane	10.1	18.2	**44.9**	30.6	18.5	25.9	**41.1**	0.27	0.22	**0.21**
bicycle	48.0	58.1	**78.4**	77.8	61.8	73.8	**81.6**	0.20	0.20	**0.13**
bus	6.8	26.0	**65.3**	58.1	27.8	39.6	**60.3**	0.32	0.21	**0.14**
car	18.4	27.4	**68.4**	63.4	28.1	37.9	**65.8**	0.23	0.21	**0.11**
motorbike	10.0	18.6	47.7	**53.4**	34.0	43.8	**58.7**	0.35	0.31	**0.18**
train	1.7	7.2	**42.9**	35.5	13.6	21.1	**51.4**	0.40	0.29	**0.22**
mean	15.8	25.9	**58.0**	53.1	30.6	40.4	**59.8**	0.30	0.24	**0.17**

Table 1: Detection accuracy (mean AP, %) and scale prediction loss without occlusion.

400, 480, 560, 640, 752, 864 and 976 pixels, respectively. A larger number of scales may lead to better detection results but the computational cost becomes more expensive. Then, we run the testing process at each scale and get 10 score maps. For each SP_s, we find the highest detection score among all score maps at different scales. We denote the scale producing the highest score as Sc_s. This provides evidence for the proper scale of the image. The final scale of the image is obtained by averaging all such evidences, *i.e.*, computing $\mathrm{Sc}^\star = \frac{1}{\mathcal{S}}\sum_{s\in\mathcal{S}}\mathrm{Sc}_s$. We use average rather than max operation in order to take multi-scale information and improve the robustness of our approach. Finally, we resize the image based on the predicted scale $\mathrm{Sc}^\star$, and run the detection process again. We will show in Section 4.2 that this simple method works well.

4 Experiments

4.1 Settings

Dataset. We use VehicleSemanticPart dataset [29] for training. It contains non-occluded images with dense part labeling of 6 objects, *i.e.*, *airplane*, *bicycle*, *bus*, *car*, *motorbike* and *train*. Some typical semantic parts are shown in Figure 1. For fair comparison, all the algorithms are trained on the bounding-box images without occlusions. As for the test set, we randomly superimpose two, three or four irrelevant objects (named *occluders*) onto the target object. **We use such synthesized images because they are easy to generate and the actual position of the occluded parts can be accurately annotated.** We also control the occlusion ratio by computing the fraction of occluded pixels on the target object.

Criterion. We evaluate our algorithm in two cases, *i.e.*, whether the target object is partially occluded. We follow the most popular criteria [8], where a detected semantic part is true-positive if it matches a ground-truth annotation, *i.e.*, the Intersection-over-Union (IoU) ratio between two boxes is not smaller than 0.5. Duplicate detection is counted as false-positive.

Baselines. In all experiments, our algorithm (denoted as **VT**) is compared to three baseline approaches, *i.e.*, single visual concept detection [29], Faster-RCNN [24], and SVM+LLC. The first one (denoted as **VC**) follows the exact implementation in [29]. The second one (denoted as **FR**) involves re-training a Faster-RCNN [24] model for each of the six classes, *i.e.*, each semantic part is considered as an “object category”. In the third baseline (denoted as **SV**), we follows the standard Bag-of-Visual-Words (BoVW) model, and train a binary SVM classifier for detecting each semantic part. We first encode each CNN feature $\mathbf{f}(\mathbf{I}_p)$ into a $|\mathcal{V}|$-dimensional vector $\mathbf{v}_p$ using Locality-sensitive Linear Coding (LLC) [28]. The

	2 Occ's, $0.2 \leqslant r < 0.4$			3 Occ's, $0.4 \leqslant r < 0.6$			4 Occ's, $0.6 \leqslant r < 0.8$		
Object	**SV**	**FR**	**VT**	**SV**	**FR**	**VT**	**SV**	**FR**	**VT**
airplane	12.0	**26.8**	23.2	9.7	**20.5**	19.3	7.5	**15.8**	15.1
bicycle	44.6	65.7	**71.7**	33.7	54.2	**66.3**	15.6	37.7	**54.3**
bus	12.3	**41.3**	31.3	7.3	**32.5**	19.3	3.6	**21.4**	9.5
car	13.4	**35.9**	**35.9**	7.7	22.0	**23.6**	4.5	**14.2**	13.8
motorbike	11.4	35.9	**44.1**	7.9	28.8	**34.7**	5.0	19.1	**24.1**
train	4.6	20.0	**21.7**	3.4	**11.1**	8.4	2.0	**7.2**	3.7
mean	16.4	37.6	**38.0**	11.6	28.2	**28.6**	6.4	19.2	**20.1**

Table 2: Detection accuracy (mean AP, %) when the object is partially occluded. Three levels of occlusion are considered.

number of bases in LLC is set to be 45, *i.e.*, the number of supporting visual concepts for each semantic part. Then, following the flowchart in Section 3.2.1, we select a positive set $\mathcal{T}_s^+$ and a negative set $\mathcal{T}_s^-$, and compute the feature vector $\mathbf{v}_p$ at each position and train a SVM classifier. At the testing stage, we compute the feature vector at each position, feed it to the SVM, and finally compose the detection score map with the confidence scores provided by the binary SVM. We do not consider some part-based models [10][9], as they are not based on deep network features, and thus do not produce state-of-the-art detection accuracy.

4.2 Semantic Part Detection without Occlusion

We first assume that the target object is not occluded by any irrelevant objects. Results of our algorithm and its competitors are summarized in Table 1. Our voting algorithm achieves comparable detection accuracy to Faster-RCNN [24], one of the state-of-the-art object detectors. Note that Faster-RCNN [24] depends on some discriminative information, while our voting algorithm relies only on integrating visual cues from visual concepts. As we will see later, our algorithm works better than Faster-RCNN [24] on the occlusion cases. Since [29] merely uses single visual concepts for detection, it produces significantly lower accuracy than our approach. SVM+LLC uses a strong classifier, but produces unsatisfying performance due to the lack of considering context information.

Scale Prediction. Since all methods are trained on cropped bounding-box images, they lack the ability of detecting semantic parts at a different scale from the training case. For Faster-RCNN [24], we construct an image pyramid, where the short side of an image will be resized to 5 scales, *i.e.*, $\{600, 688, 800, 976, 1200\}$, to fuse detection results at test time. However, we cap the longest side at 3000 pixels to avoid exceeding GPU memory. For other methods, we applied the same scale prediction algorithm, which is described in Section 3.3.1. To illustrate the importance of scale prediction algorithm, we present an *oracle* detection option, which resizes each image according to the ground-truth bounding box size, *i.e.*, after rescaling, the short edge of the target object becomes 224, as in the training case. As we can see in Table 1, this improves the performance of all three competitors significantly.

To analyze the accuracy of scale prediction, we perform a diagnostic experiment. For each testing image, we use the ground-truth bounding box to compute the *actual size* it should be rescaled into. For example, if a *car* occupies a 150×100 region in a 200×250 image, given that 150×100 is rescaled to 336×224 (as in training), the full image should be rescaled to 448×560. If the short edge is a for the actual size and b in scale prediction, then the loss in rescaling is computed as $\ln(\max\{a,b\} / \min\{a,b\})$. A perfect prediction has a loss of 0. Results are summarized in Table 1. The voting algorithm predicts the object scale more accurately, leading to a higher overall detection accuracy, and the least accuracy drop without using the oracle information.

Diagnosis. We diagnose the voting algorithm by analyzing the contribution of some modules. Two options are investigated with oracle information. First, changing the number of supporting visual concepts. We decrease $|\mathcal{V}_s|$ from 45 to 30, 20 and 10, and observe 1.0%, 3.3% and 7.8% mean accuracy drop, respectively. On the other hand, increasing it from 45 to 60 does not impact the performance much (0.4% improvement). Similar phenomena are also observed when occlusion is present. Therefore, we conclude that the voting method requires a sufficient number of supporting visual concepts, but using too many of them may introduce redundant information which increases the computational overhead. Second, We consider another smaller neighborhood threshold $\gamma_{\mathrm{th}} = 56$, which leads to a spatial offset map of size 7×7. This causes 4.5% accuracy drop, arguably caused by the lack of long-distance visual cues which is useful especially when occlusion is present.

4.3 Semantic Part Detection under Occlusion

Next, we investigate the case that the target object is partially occluded. We construct 3 datasets with different occluder numbers and occlusion ratios. We still apply the multi-scale detection described in Section 3.3.1. Note that all models are trained on the non-occlusion dataset, and we will report the results with occluded training objects in the future.

Results are shown in Table 2. To save space, we ignore the performance of **VC** [29] since it is the weakest one among all baseline methods. As occlusion becomes heavier, we observe significant accuracy drop. Note that all these methods are trained on the non-occlusion dataset. Since our voting algorithm has the ability of inferring the occluded parts via its contexts, the accuracy drop is the smallest. Consequently, its advantage in detection accuracy becomes more significant over other competitors.

5 Conclusions and Future Work

We address the task of detecting semantic parts under occlusions. We design a novel framework which involves modeling spatial relationship, finding supporting visual concepts, performing log-likelihood ratio tests, and summarizing visual cues with a voting process. Experiments verify that our algorithm works better than previous work when the target object is partially occluded. Our algorithm also enjoys the advantage of being explainable, which is difficult to achieve in the state-of-the-art holistic object detectors.

In the future, we will extend our approach to detect the entire object under occlusion. In our preliminary experiments, we have already observed that proposal-based methods such as Faster-RCNN are not robust to heavy occlusion on the entire object. This can be dealt with in two different ideas, *i.e.*, merging the detected semantic parts in a bottom-up manner, or using a pre-defined template to organize the detected semantic parts. We will also try to construct a dataset with real occluded images, and a dataset with more object classes.

Acknowledgements. This work is supported by the Intelligence Advanced Research Projects Activity (IARPA) via DoI/IBC contract number D16PC00007, and also by the NSF STC award CCF-1231216. We thank Dr. Vittal Premachandran for his enormous help in this project. We also thank Dr. Wei Shen, Dr. Ehsan Jahangiri, Dr. Yan Wang, Weichao Qiu, Chenxi Liu, Zhuotun Zhu, Siyuan Qiao and Yuyin Zhou for instructive discussions.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the Objectness of Image Windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2189–2202, 2012.

[2] Y. Amit. *2D Object Detection and Recognition: Models, Algorithms, and Networks*. MIT Press, 2002.

[3] H. Azizpour and I. Laptev. Object Detection Using Strongly-supervised Deformable Part Models. *European Conference on Computer Vision*, 2012.

[4] X. Chen and A. Yuille. Articulated Pose Estimation by a Graphical Model with Image Dependent Pairwise Relations. *Advances in Neural Information Processing Systems*, 2014.

[5] X. Chen and A. Yuille. Parsing Occluded People by Flexible Compositions. *Computer Vision and Pattern Recognition*, 2015.

[6] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille. Detect What You Can: Detecting and Representing Objects Using Holistic Models and Body Parts. *Computer Vision and Pattern Recognition*, 2014.

[7] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. *Computer Vision and Pattern Recognition*, 2005.

[8] M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge, 2010.

[9] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[10] S. Fidler and A. Leonardis. Towards Scalable Representations of Object Categories: Learning a Hierarchy of Parts. *Computer Vision and Pattern Recognition*, 2007.

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *Computer Vision and Pattern Recognition*, 2014.

[12] W.E.L. Grimson and D. Huttenlocher. *Object Recognition by Computer: the Role of Geometric Constraints*. MIT Press, 1991.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. *Computer Vision and Pattern Recognition*, 2016.

[14] A. Kar, S. Tulsiani, J. Carreira, and J. Malik. Amodal Completion and Size Constancy in Natural Scenes. *International Conference on Computer Vision*, 2015.

[15] S. Konishi, A. Yuille, J. Coughlan, and S.C. Zhu. Fundamental Bounds on Edge Detection: An Information Theoretic Evaluation of Different Edge Cues. *Computer Vision and Pattern Recognition*, 1999.

[16] A. Krizhevsky, I. Sutskever, and G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, 2012.

[17] B. Leibe, A. Leonardis, and B. Schiele. Combined Object Categorization and Segmentation with an Implicit Shape Model. *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[18] B. Li, T. Wu, and S.C. Zhu. Integrating Context and Occlusion for Car Detection by Hierarchical And-Or Model. *European Conference on Computer Vision*, 2014.

[19] K. Li and J. Malik. Amodal instance segmentation. *European Conference on Computer Vision*, 2016.

[20] Y. Li, L. Liu, C. Shen, and A. van den Hengel. Mid-level Deep Pattern Mining. *Computer Vision and Pattern Recognition*, 2015.

[21] J.L. Long, N. Zhang, and T. Darrell. Do Convnets Learn Correspondence? *Advances in Neural Information Processing Systems*, 2014.

[22] S. Maji and J. Malik. Object Detection Using a Max-Margin Hough Transform. *Computer Vision and Pattern Recognition*, 2009.

[23] R. Okada. Discriminative Generalized Hough Transform for Object Dectection. *International Conference on Computer Vision*, 2009.

[24] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems*, 2015.

[25] M. Simon and E. Rodner. Neural Activation Constellations: Unsupervised Part Model Discovery with Convolutional Networks. *International Conference on Computer Vision*, 2015.

[26] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations*, 2014.

[27] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, and A.W.M. Smeulders. Selective Search for Object Recognition. *International Journal of Computer Vision*, 104(2): 154–171, 2013.

[28] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-Constrained Linear Coding for Image Classification. *Computer Vision and Pattern Recognition*, 2010.

[29] J. Wang, Z. Zhang, C. Xie, V. Premachandran, and A. Yuille. Unsupervised Learning of Object Semantic Parts from Internal States of CNNs by Population Encoding. *arXiv preprint arXiv:1511.06855*, 2015.

[30] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond PASCAL: A Benchmark for 3D Object Detection in the Wild. *IEEE Winter Conference on Applications of Computer Vision*, 2014.

[31] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang. The Application of Two-level Attention Models in Deep Convolutional Neural Network for Fine-grained Image Classification. *Computer Vision and Pattern Recognition*, 2015.

[32] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based R-CNNs for Fine-Grained Category Detection. *European Conference on Computer Vision*, 2014.

[33] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object Detectors Emerge in Deep Scene CNNs. *International Conference on Learning Representations*, 2015.