Growing Interpretable Part Graphs on ConvNets via Multi-Shot Learning

Quanshi Zhang, Ruiming Cao, Ying Nian Wu, and Song-Chun Zhu

University of California, Los Angeles

Abstract

This paper proposes a learning strategy that extracts object-part concepts from a pre-trained convolutional neural network (CNN), in an attempt to 1) explore explicit semantics hidden in CNN units and 2) gradually grow a semantically interpretable graphical model on the pre-trained CNN for hierarchical object understanding. Given part annotations on very few (*e.g.* 3–12) objects, our method mines certain latent patterns from the pre-trained CNN and associates them with different semantic parts. We use a four-layer And-Or graph to organize the mined latent patterns, so as to clarify their internal semantic hierarchy. Our method is guided by a small number of part annotations, and it achieves superior performance (about 13%–107% improvement) in part center prediction on the PASCAL VOC and ImageNet datasets[1].

Introduction

Convolutional neural networks (LeCun et al. 1998; Krizhevsky, Sutskever, and Hinton 2012; He et al. 2016) (C-NNs) have achieved near human-level performance in object classification on some datasets. However, in real-world applications, we are still facing the following two important issues.

Firstly, given a CNN that is pre-trained for object classification, it is desirable to derive an interpretable graphical model to explain explicit semantics hidden inside the CN-N. Based on the interpretable model, we can go beyond the detection of object bounding boxes, and discover an object's latent structures with different part components from the pre-trained CNN representations.

Secondly, it is also desirable to learn from very few annotations. Unlike data-rich applications (*e.g.* pedestrian and vehicle detection), many visual tasks demand for modeling certain objects or certain object parts *on the fly*. For example, when people teach a robot to grasp the handle of a cup, they may not have enough time to annotate sufficient training samples of cup handles before the task. It is better to mine common knowledge of cup handles from a few examples on the fly.

Motivated by the above observations, in this paper, given a pre-trained CNN, we use very few (3–12) annotations

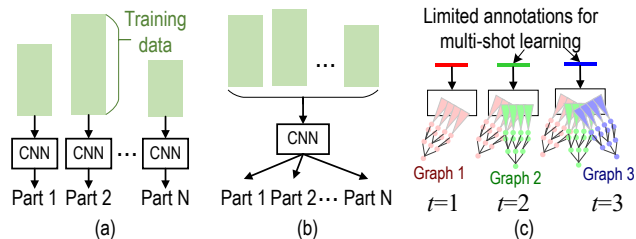[1]Codes here: *https://sites.google.com/site/cnnsemantics/*

Figure 1: Comparison of three learning strategies. (a) Individually learning/fine-tuning each part without sharing patterns between parts has large information redundancy in model representation. (b) Jointly learning/fine-tuning parts requires all parts to be simultaneously learned. (c) Given a small number (*e.g.* 3–12) of part annotations based on demands on the fly, we incrementally grow new semantic graphs on a pre-trained CNN, which associate certain CNN units with new parts.

to model a semantic part for the task of part localization. When a CNN is pre-trained using object-level annotations, we believe that its conv-layers have contained implicit representations of the objects. We call the implicit representations *latent patterns*, each corresponding to a component of the semantic part (namely a sub-part) or a contextual region *w.r.t.* the semantic part. *For each semantic part, our goal is to mine latent patterns from the conv-layers related to this part. We use an And-Or graph (AOG) to organize the mined latent patterns to represent the semantic hierarchy of the part.*

Input and output: Given a pre-trained CNN and a number of images for a certain category, we only annotate the semantic parts on a few images as input. We develop a method to grow a semantic And-or Graph (AOG) on the pre-trained CNN, which associates certain CNN units with the semantic part. Our method does not require massive annotations for learning, and can work with even a **single** part annotation. We can use the learned AOG to parse/localize object parts and their sub-parts for hierarchical object parsing.

Fig. 2 shows that the AOG has four layers. In the AOG, each OR node encodes its alternative representations as children, and each AND node is decomposed into its constituents.

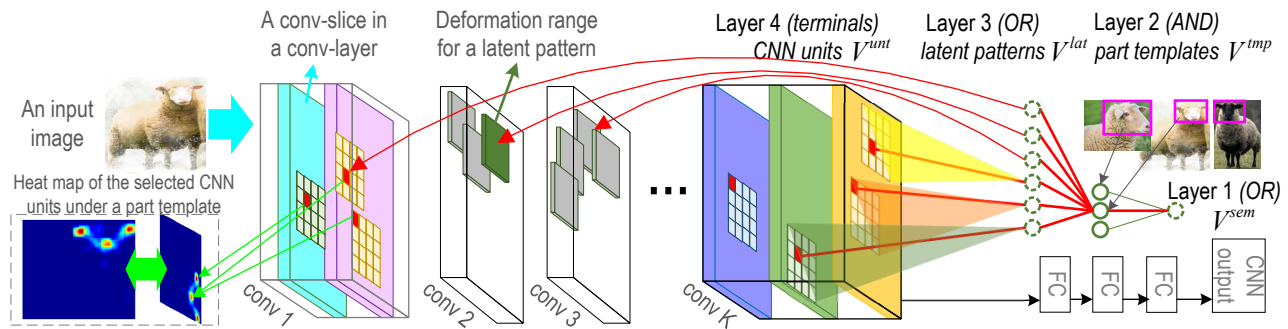- Layer 1: the top OR node for *semantic part* describes the

Figure 2: Semantic And-Or graph grown on the pre-trained CNN. The AOG associates CNN units with certain semantic parts (head, here). Red lines in the AOG indicate a parse graph for concept association. To visualize these latent patterns, we show the heat map (left) at the 5-th conv-layers in the VGG-16 network, which sums up the associated units (red squares) throughout all conv-slices. In Fig. 4, we reconstructed the dog head using the learned AOG to show its interpretability.

head of a sheep in Fig. 2. It lists a number of part templates as children.

- Layer 2: AND nodes for *part templates* correspond to different poses or local appearances for the part, *e.g.* a black sheep head from a front view and a white sheep head from side view.

- Layer 3: OR nodes for *latent patterns* describe sub-parts of the sheep head (*e.g.* a corner of the nose) or a contextual region (*e.g.* the neck region).

- Layer 4: terminal nodes are *CNN units*. A latent pattern naturally corresponds to a certain range of units within a conv-slice. It selects a CNN unit within this range to account for local shape deformation of this pattern.

Learning method and key benefits: The basic idea for growing AOG is to define a metric to distinguish reliable latent patterns from noisy neural activations in the conv-layers. We expect latent patterns with high reliability to 1) consistently represent certain sub-parts on the annotated object samples, 2) frequently appear in unannotated objects, and 3) keep stable spatial relationship with other latent patterns. We mine reliable latent patterns to construct the AOG. This learning method is related to previous studies of pursuing AOGs, which mined hierarchical object structures from Gabor wavelets on edges (Si and Zhu 2013) and HOG features (Zhang, Wu, and Zhu 2015). We extend such ideas to feature maps of neural networks.

Our method has the following three key benefits:

- **CNN semanticization:** We semanticize the pre-trained C-NN by connecting its units to an interpretable AOG. In recent years, people have shown a special interest in opening the black-box representation of the CNN. In this paper, we retrieve "implicit" patterns from the CNN, and use the AOG to associate each pattern with a certain "explicit" semantic part. We can regard the AOG as an interpretable representation of CNN patterns, which may contribute to the understanding of black-box knowledge organization in the CNN.

- **Multi-shot learning:** The idea of pattern mining also enables multi-shot learning from small data. Conventional end-to-end learning usually requires a large number of annotations to learn/finetune networks. In contrast, in our learning

scenario, all patterns in the CNN have been well pre-trained using object-level annotations. We only use very few (3–12) part annotations to retrieve certain latent patterns, instead of finetuning CNN parameters. For example, we use the annotation of a specific tiger head to mine latent patterns. The mined patterns are not over-fitted to the head annotation, but represent common head appearance among different tigers. Therefore, we can greatly reduce the number of part annotations for training.

- **Incremental learning:** we can incrementally enrich the knowledge of semantic parts. Given a pre-trained CNN, we can incrementally grow new neural connections from CNN units to a new AOG, in order to represent a new semantic part. It is important to maintain the generality of the pre-trained CNN during the learning procedure. *I.e.* we do not change/fine-tune the original convolutional weights within the CNN, when we grow new AOGs. This allows us to continuously add new semantic parts to the same CNN, without worrying about the model drift problem.

Contributions of this paper are summarized as follows.
1) From the perspective of model learning, given a few part annotations, we propose a method to incrementally grow interpretable AOGs on a pre-trained CNN to gradually model semantic parts of the object.
2) From the perspective of knowledge transferring, our method semanticizes a CNN by mining reliable latent patterns from noisy neural responses of the CNN and associating the implicit patterns with explicit semantic parts.
3) To the best of our knowledge, we can regard our method as the first study to achieve weakly supervised (*e.g.* 3–12 annotations) learning for part localization. Our method exhibits superior localization performance in experiments (about 13%–107% improvement in part center prediction).

Related work

Long-term learning & short-term learning: As reported in (Kumaran, Hassabis, and McClelland 2016), there are "two learning systems instantiated in mammalians:" 1) the neocortex gradually acquires sophisticated knowledge representation, and 2) the hippocampus quickly learns specifics of individual experiences. CNNs are typically trained using

big data, and contain rich appearance patterns of objects. If one compares CNNs to the neocortex, then the fast retrieval of latent patterns related to a semantic part can be compared to the short-term learning in hippocampus.

Semantics in the CNN: In order to explore the hidden semantics in the CNN, many studies have focused on the visualization of CNN units (Zeiler and Fergus 2014; Mahendran and Vedaldi 2015; Simonyan, Vedaldi, and Zisserman 2013; Dosovitskiy and Brox 2015; Aubry and Russell 2015) and analyzed their statistical features (Dosovitskiy et al. 2014; Szegedy et al. 2014; Yosinski et al. 2014; Lu 2015). Liu et al. (Liu, Shen, and van den Hengel 2015) extracted and visualized a subspace of CNN features.

Going beyond "passive" visualization, some studies "actively" extracted CNN units with certain semantics for different applications. Zhou *et al.* (Zhou et al. 2015; 2016) discovered latent "scene" semantics from CNN feature maps. Simon *et al.* discovered objects (Simon and Rodner 2015) in an unsupervised manner from CNN feature maps, and learned semantic parts in a supervised fashion (Simon, Rodner, and Denzler 2014). In our study, given very few part annotations, we mine CNN patterns that are related to the semantic part. Obtaining clear semantics makes it easier to transfer CNN patterns to other part-based tasks.

AOG for knowledge transfer: Transferring hidden patterns in the CNN to other tasks is important for neural networks. Typical research includes end-to-end fine-tuning and transferring CNN knowledge between different categories (Yosinski et al. 2014; Gallagher, Tang, and Tu 2015) and/or datasets (Ganin and Lempitsky 2015). In contrast, we believe that a good explanation and transparent representation of part knowledge will creates a new possibility of transferring part knowledge. As in (Zhu et al. 2008; Si and Zhu 2013), the AOG is suitable to represent the semantic hierarchy, which enables semantic-level interactions between human and neural networks.

Modeling "objects" vs. modeling "parts" in un-/weakly-supervised learning: Generally speaking, in terms of un-/weakly-supervised learning, modeling parts is usually more challenging than modeling entire objects. Given image-level labels (without object bounding boxes), object discovery (Oquab et al. 2015; Simon and Rodner 2015; Pathak, Kr ahenb uhl, and Darrell 2015) can be achieved by identifying common foreground patterns from noisy background. Closed boundaries and common object structure are also strong prior knowledge for object discovery.

In contrast to objects, semantic parts are hardly distinguishable from other common foreground patterns in an unsupervised manner. Some parts (*e.g.* the abdomen) do not have shape boundaries to determine their shape extent. Inspired by graph mining (Zhang, Wu, and Zhu 2015; Zhang et al. 2016; 2014), we mine common patterns from CNN activation maps in conv-layers to explain the part.

Part localization/detection vs. semanticizing CNN patterns: Part localization/detection is an important task in computer vision (Azizpour and Laptev 2012; Simon, Rodner, and Denzler 2014; Li et al. 2013; Chen et al. 2014). There are two key points to differentiate our study from conventional part-detection approaches. First, most methods for

detection, such as the CNN and the DPM (LeCun et al. 1998; Felzenszwalb, Girshick, and McAllester 2010; Tsogkas et al. 2015), limit their attention to the classification problem. In contrast, our effort is to clarify semantic meanings of implicit CNN patterns. Second, instead of summarizing knowledge from massive annotations, our method mines CNN semantics with very limited supervision.

And-Or graph for part parsing

In this section, we introduce the structure of the AOG and part parsing/localization based on the AOG. The AOG structure is suitable for clearly representing semantic hierarchy of a part. The method for mining latent patterns and building the AOG will be introduced in the next section. An AOG represents the semantic structure of a part at four layers.

Layer	Name	Node type	Notation
1	semantic part	OR node	V^{sem}
2	part template	AND node	$V^{\text{tmp}} \in \Omega^{\text{tmp}}$
3	latent pattern	OR node	$V^{\text{lat}} \in \Omega^{\text{lat}}$
4	CNN unit	Terminal node	$V^{\text{unt}} \in \Omega^{\text{unt}}$

Each OR node in the AOG represents a list of alternative appearance (or deformation) candidates. Each AND node is composed of a number of latent patterns to describe its subregions.

In Fig. 2, given CNN activation maps on an image I[2], we can use the AOG for part parsing. From a top-down perspective, the parsing procedure 1) identifies a part template for the semantic part; 2) parses an image region for the selected part template; 3) for each latent pattern under the part template, it selects a CNN unit within a certain deformation range to represent this pattern.

In this way, we select certain AOG nodes in a *parse graph* to explain sub-parts of the object (shown as red lines in Fig. 2). For each node V in the parse graph, we parse an image region Λ_V within image I[3]. We use $S_I(V)$ to denote an inference/parsing score, which measures the fitness between the parsed region Λ_V and V (as well as the sub-AOG under V).

Given an image I[2] and an AOG, the actual parsing procedure is solved by dynamic programming in a bottom-up manner, as follows.

Terminal nodes (CNN units): We first focus on parsing configurations of terminal nodes. Terminal nodes under a latent pattern are displaced in location candidates of this latent pattern. Each terminal node V^{unt} has a fixed image region $\Lambda_{V^{\text{unt}}}$: we propagate V^{unt}'s receptive field back to the image plane as $\Lambda_{V^{\text{unt}}}$. We compute V^{unt}'s inference score $S_I(V^{\text{unt}})$

---

[2]Because the CNN has demonstrated its superior performance in object detection, we assume that the target object can be well detected by the pre-trained CNN. Thus, to simplify the learning scenario, we crop I to only contain the object, resize it to the image size for CNN inputs, and only focus on the part localization task.

[3]Image regions of OR nodes are propagated from their children. Each terminal node has a fixed image region, and each part template (AND node) has a fixed region scale (will be introduced later). Thus, we only need infer the center position of each part template in (2) during part parsing.

based on both its neural response value and its displacement *w.r.t.* its parent (see appendix for details[4]).

Latent patterns: Then, we propagate parsing configurations from terminal nodes to latent patterns. Each latent pattern V^{lat} is an OR node. V^{lat} naturally corresponds to a square within a certain conv-slice in the output of a certain CNN conv-layer as its deformation range[5]. V^{lat} connects all the CNN units within the deformation range as children, which represent different deformation candidates. Given parsing configurations of its children CNN units as input, V^{lat} selects the child $\hat{V}^{\text{unt}}$ with the highest score as the true deformation configuration:

$$S_I(V^{\text{lat}}) = \max_{V^{\text{unt}} \in Child(V^{\text{lat}})} S_I(V^{\text{unt}}), \quad \hat{\Lambda}_{V^{\text{lat}}} = \Lambda_{\hat{V}^{\text{unt}}} \quad (1)$$

Part templates: Each part template V^{tmp} is an AND node, which uses its children (latent patterns) to represent its sub-part/contextual regions. Based on the relationship between V^{tmp} and its children, V^{tmp} uses its children's parsing configurations to parse its own image region $\Lambda_{V^{\text{tmp}}}$. Given parsing scores of children, V^{tmp} computes the image region $\hat{\Lambda}_{V^{\text{tmp}}}$ that maximizes its inference score.

$$S_I(V^{\text{tmp}}) = \max_{\Lambda_{V^{\text{tmp}}}} \sum_{V^{\text{lat}} \in Child(V^{\text{tmp}})} \left[S_I(V^{\text{lat}}) + S^{\text{inf}}(\Lambda_{V^{\text{tmp}}} | \hat{\Lambda}_{V^{\text{lat}}}) \right] \quad (2)$$

Just like typical part models (*e.g.* DPMs), the AND node uses each child's region V^{lat} to infer its own region. $S_I(V^{\text{lat}})$ measures the score of each child, and $S^{\text{inf}}(\Lambda_{V^{\text{tmp}}} | \hat{\Lambda}_{V^{\text{lat}}})$ measures spatial compatibility between V^{tmp} and each child V^{lat} in region parsing (see the appendix for formulations).

Semantic part: Finally, we propagate parsing configurations to the top node V^{sem}. V^{sem} is an OR node. It contains a list of alternative templates for the part. Just like OR nodes of latent patterns, V^{sem} selects the child $\hat{V}^{\text{tmp}}$ with the highest score as the true parsing configuration:

$$S_I(V^{\text{sem}}) = \max_{V^{\text{tmp}} \in Child(V^{\text{sem}})} S_I(V^{\text{tmp}}), \quad \hat{\Lambda}_{V^{\text{sem}}} = \hat{\Lambda}_{\hat{V}^{\text{tmp}}} \quad (3)$$

Learning: growing an And-Or graph

The basic idea of AOG growing is to distinguish reliable latent patterns from noisy neural responses in conv-layers and use reliable latent patterns to construct the AOG.

Training data: Let $\mathbf{I}$ denote an image set for a target category. Among all objects in $\mathbf{I}$, we label bounding boxes of the semantic part in a small number of images, $\mathbf{I}^{\text{ant}} = \{I_1, I_2, \ldots, I_M\} \subset \mathbf{I}$. In addition, we manually define a number of templates for the part. Thus, for each $I \in \mathbf{I}^{\text{ant}}$, we annotate $(\Lambda^*_{V^{\text{sem}}}, V^{\text{tmp}*})$, where $\Lambda^*_{V^{\text{sem}}}$ denotes the ground-truth bounding box of the part in I, and $V^{\text{tmp}*}$ specifies the ground-truth template ID for the part.

Which AOG parameters to learn: We can use human annotations to define the first two layers of the AOG. If people specify a total of m different part templates during the

annotation process, correspondingly, we can directly connect the top node with m part templates $\{V^{\text{tmp}*}\}$ as children. For each part template V^{tmp}, we fix a constant scale for its region $\Lambda_{V^{\text{tmp}}}$. *I.e.* if there are n ground-truth part boxes that are labeled for V^{tmp}, we compute the average scale among the n part boxes as the constant scale for $\Lambda_{V^{\text{tmp}}}$.

Thus, the key to AOG construction is to mine children latent patterns for each part template. We need to mine latent patterns from a total of K conv-layers. We select n_k latent patterns from the k-th ($k = 1, 2, \ldots, K$) conv-layer, where K and $\{n_k\}$ are hyper-parameters. Let each latent pattern V^{lat} in the k-th conv-layer correspond to a square deformation range[5], which is located in the $D_{V^{\text{lat}}}$-th conv-slice of the conv-layer. $\overline{\mathbf{P}}_{V^{\text{lat}}}$ denotes the center of the range. As analyzed in the appendix, we only need to estimate the parameters of $D_{V^{\text{lat}}}, \overline{\mathbf{P}}_{V^{\text{lat}}}$ for V^{lat}.

How to learn: We mine the latent patterns by estimating their best locations $D_{V^{\text{lat}}}, \overline{\mathbf{P}}_{V^{\text{lat}}} \in \boldsymbol{\theta}$ that maximize the following objective function.

$$\max_{\boldsymbol{\theta}} \Big\{ \underbrace{\text{mean}_{I \in \mathbf{I}^{\text{ant}}} \big[\mathbf{L}^{\text{gen}}(I, \boldsymbol{\theta}) + \mathbf{L}^{\text{dis}}(I, \boldsymbol{\theta}) \big]}_{\text{annotated images}} + \underbrace{\text{mean}_{I' \in \mathbf{I}} \sum_{V^{\text{lat}}} S^{\text{unsup}}_{I'}(V^{\text{lat}})}_{\text{unannotated images}} \Big\} \quad (4)$$

where $\boldsymbol{\theta}$ is the set of AOG parameters. First, let us focus on the first half of the equation, which learns from part annotations. $\mathbf{L}^{\text{gen}}(I, \boldsymbol{\theta})$ and $\mathbf{L}^{\text{dis}}(I, \boldsymbol{\theta})$ are the generative inference score and discriminative score of part parsing, respectively, on I. Given annotations $(\Lambda^*_{V^{\text{sem}}}, V^{\text{tmp}*})$ on I, we define

$$\begin{aligned} \mathbf{L}^{\text{gen}}(I, \boldsymbol{\theta}) &= S_I(V^{\text{sem}}) \approx S_I(V^{\text{tmp}*}) \\ \mathbf{L}^{\text{dis}}(I, \boldsymbol{\theta}) &= -\lambda_{V^{\text{tmp}*}} \|\hat{\mathbf{P}}_{V^{\text{sem}}} - \mathbf{P}^*_{V^{\text{sem}}}\| \end{aligned} \quad (5)$$

where the discriminative score measures the localization error between the parsed part region $\hat{\mathbf{P}}_{V^{\text{sem}}}$ and the ground truth $\mathbf{P}^*_{V^{\text{sem}}}$. We ignore the small probability of the AOG assigning an annotated image with an incorrect part template to simplify the computation of generative scores.

The second half of (4) learns from objects without part annotations.

$$S^{\text{unsup}}_{I'}(V^{\text{lat}}) = \lambda^{\text{unsup}} \big[S^{\text{rsp}}_{I'}(\hat{V}^{\text{unt}}) + S^{\text{loc}}_{I'}(\hat{V}^{\text{unt}}) - \lambda^{\text{close}} \|\Delta \mathbf{P}_{V^{\text{lat}}}\|^2 \big] \quad (6)$$

where latent pattern V^{lat} selects CNN unit $\hat{V}^{\text{unt}}$ as its deformation configuration on I'. The first term $S^{\text{rsp}}_{I'}(\hat{V}^{\text{unt}})$ denotes the neural response of the CNN unit $\hat{V}^{\text{unt}}$. The second term $S^{\text{loc}}_{I'}(\hat{V}^{\text{unt}}) = -\lambda^{\text{loc}} \|\hat{\mathbf{P}}_{V^{\text{unt}}} - \overline{\mathbf{P}}_{V^{\text{lat}}}\|^2$ measures the deformation level of the latent pattern. The third term measures the spatial closeness between the latent pattern and its parent V^{tmp}. We assume that 1) latent patterns that frequently appear among unannotated objects may potentially represent stable sub-parts and should have higher priorities; and that 2) latent patterns spatially closer to V^{tmp} are usually more reliable. Please see the appendix for details of $S^{\text{rsp}}_{I'}(\hat{V}^{\text{unt}})$ and scalar weights of λ^{unsup}, λ^{close}, and λ^{loc}.

When we set $\lambda_{V^{\text{tmp}*}}$ to a constant $\lambda^{\text{inf}} \sum_{k=1}^{K} n_k$, we can transform the learning objective in (4) as follows.

$$\forall V^{\text{tmp}} \in \Omega^{\text{tmp}}, \quad \max_{\boldsymbol{\theta}_{V^{\text{tmp}}}} \mathbf{L}, \quad \mathbf{L} = \sum_{V^{\text{lat}} \in Child(V^{\text{tmp}})} Score(V^{\text{lat}}) \quad (7)$$

where $Score(V^{\text{lat}}) = \text{mean}_{I \in \mathbf{I}_{V^{\text{tmp}}}} [S_I(V^{\text{lat}}) + S^{\text{inf}}(\Lambda^*_{V^{\text{sem}}} | \hat{\Lambda}_{V^{\text{lat}}})] + \text{mean}_{I' \in \mathbf{I}} S^{\text{unsup}}_{I'}(V^{\text{lat}})$. $\boldsymbol{\theta}_{V^{\text{tmp}}} \subset \boldsymbol{\theta}$ denotes the parameters for

---

[4]Please see the section of appendix for details.

[5]We set a constant deformation range for each latent pattern, which potentially covers 75×75 pxls on the image. Deformation ranges of different patterns in the same conv-slice may overlap.

Table 1: Average number of children

AOG Layer	#1 semantic part	#2 part template	#3 latent pattern
Children number	3	4575.8	136.4

the sub-AOG of V^{tmp}. We use $\mathbf{I}_{V^{\text{tmp}}} \subset \mathbf{I}^{\text{ant}}$ to denote the subset of images that are annotated with V^{tmp} as the ground-truth part template.

Learning the sub-AOG for each part template: Based on (7), we can mine the sub-AOG for each part template V^{tmp}, which uses this template's own annotations on images $I \in \mathbf{I}_{V^{\text{tmp}}} \subset \mathbf{I}^{\text{ant}}$, as follows.
1) We first enumerate all possible latent patterns corresponding to the k-th CNN conv-layer ($k = 1, \ldots, K$), by sampling all pattern locations *w.r.t.* $D_{V^{\text{lat}}}$ and $\overline{\mathbf{P}}_{V^{\text{lat}}}$.
2) Then, we sequentially compute $\hat{\Lambda}_{V^{\text{lat}}}$ and $Score(V^{\text{lat}})$ for each latent pattern.
3) Finally, we sequentially select a total of n_k latent patterns. In each step, we select $\hat{V}^{\text{lat}} = \text{argmax}_{V^{\text{lat}}} \Delta \mathbf{L}$. *I.e.* we select latent patterns with top-ranked values of $Score(V^{\text{lat}})$ as V^{tmp}'s children.

Experiments

Implementation details

We chose the 16-layer VGG network (VGG-16) (Simonyan and Zisserman 2015) that was pre-trained using the 1.3M images in the ImageNet ILSVRC 2012 dataset (Deng et al. 2009) for object classification. Then, given a target category, we used images in this category to fine-tune the original VGG-16 (based on the loss for classifying target objects and background). VGG-16 has 13 conv-layers and 3 fully connected layers. We chose the last 9 (from the 5-th to the 13-th) conv-layers as valid conv-layers, from which we selected units to build the AOG.

Note that during the learning process, we applied the following two techniques to further refine the AOG model. First, multiple latent patterns in the same conv-slice may have similar positions $\overline{\mathbf{P}}_{V^{\text{lat}}}$, and their deformation ranges may highly overlap with each other. Thus, we selected the latent pattern with the highest $Score(V^{\text{lat}})$ within each small range of $\epsilon \times \epsilon$ in this conv-slice, and removed other nearby patterns to obtain a spare AOG structure. Second, for each V^{tmp}, we estimated n_k, *i.e.* the best number of latent patterns in conv-layer k. We assumed that scores of all the latent patterns in the k-th conv-layer follow the distribution of $Score(V^{\text{lat}}) \sim \alpha \exp[-(\beta rank)^{0.5}] + \gamma$, where $rank$ denotes the score rank of V^{lat}. We found that when we set $n_k = \lceil 0.5/\beta \rceil$, the AOG usually had reliable performance.

Datasets

We tested our method on three benchmark datasets: the PASCAL VOC Part Dataset (Chen et al. 2014), the CUB200-2011 dataset (Wah et al. 2011), and the ILSVRC 2013 DET dataset (Deng et al. 2009). Just like in most part-localization studies (Chen et al. 2014), we also selected six animal categories—*bird, cat, cow, dog, horse*, and *sheep*—from the PASCAL Part Dataset for evaluation, which prevalently contain non-rigid shape deformation. The CUB200-2011 dataset contains 11.8K images of 200 bird species. As in (Branson, Perona, and Belongie 2011; Simon, Rodner, and Denzler 2014), we regarded these images as a single bird category by ignoring the species labels. All the above seven categories have ground-truth annotations of the *head* (it is the *forehead* part in the CUB200-2011 dataset) and *torso/back*. Thus, for each category, we learned two AOGs to model its *head* and *torso/back*, respectively.

In order to provide a more comprehensive evaluation of part localization, we built a larger object-part dataset based on the off-the-shelf ILSVRC 2013 DET dataset. We used 30 **animal** categories among all the 200 categories in the ILSVRC 2013 DET dataset. We annotated bounding boxes for the *heads* and *front legs/feet* in these animals as two common semantic parts for evaluation. In Experiments, we annotated 3–12 boxes for each part to build the AOG, and we used the rest images in the dataset as testing images.

Two experiments on multi-shot learning

We applied our method to all animal categories in the above three benchmark datasets. We designed two experiments to test our method in the scenarios of (1×3)-shot learning and (4×3)-shot learning, respectively. We applied the learned AOGs to part localization for evaluation.

Exp. 1, three-shot AOG construction: For each semantic part of an object category, we learn three different part templates. We annotated a single bounding box for each part template. Thus, we used a total of three annotations to build the AOG for this part.

Exp. 2, AOG construction with more annotations: We continuously added more part annotations to check the performance changes. Just as in Experiment 1, each part contains the same three part templates. For each part template, we annotated four parts in four different object images to build the corresponding AOG.

Baselines

We compared our method with the following nine baselines. The first baseline was the fast-RCNN (Girshick 2015). We directly used the fast-RCNN to detect the target parts on objects. To enable a fair comparison, we learned the fast-RCNN by first fine-tuning the VGG-16 network of the fast-RCNN using all object images in the target category and then training the fast-RCNN using the part annotations. The second baseline was the strongly supervised DPM (SS-DPM) (Azizpour and Laptev 2012), which was trained with part annotations for part localization. The third baseline was proposed in (Li et al. 2013), which trained a DPM component for each object pose to localize object parts (namely, PL-DPM). We used the graphical model proposed in (Chen et al. 2014) as the fourth baseline for part localization (PL-Graph). The fifth baseline, namely CNN-PDD, was proposed by (Simon, Rodner, and Denzler 2014), which selected certain conv-slices (channels) of the CNN to represent the target object part. The sixth baseline (VGG-PDD-finetuned) was an extension of CNN-PDD, which was con-

Table 2: Part localization performance

Dataset	Exp. 1: 3-shot learning				Exp. 2: 12-shot learning		
	Pascal		ImageNet		Pascal		ImageNet
Semantic part	Head	Torso	Head	F-legs	Head	Torso	Head
SS-DPM	1.5	7.8	7.0	5.1	3.4	10.8	18.8
PL-DPM	1.1	2.1	3.9	1.2	3.1	4.2	6.5
PL-Graph	8.6	19.8	15.9	6.3	7.0	23.5	25.1
Fast-RCNN	15.5	40.0	31.9	10.9	31.3	52.9	53.6
Ours	**29.2**	**50.7**	**40.1**	**23.2**	**34.6**	**59.5**	**55.0**

Because object parts may not have clear boundaries, many studies (Oquab et al. 2015; Simon, Rodner, and Denzler 2014) did not consider part scales in evaluation. Similarly, our method mainly localizes part center, and does not discriminatively learn a model for the regression of part bounding boxes, which is different from fast-RCNN methods. Instead, we simply fix a constant bounding-box scale for each part template, *i.e.* the average scale of part annotations for this part template. Nevertheless, our method still exhibits superior performance.

Table 3: Normalized distance of part localization. The performance was evaluated using the CUB200-2011 dataset.

Semantic part	Exp. 1: 3-shot learning		Exp. 2: 12-shot learning	
	Head	Torso	Head	Torso
SS-DPM	0.3469	0.2604	0.2925	0.2427
PL-DPM	0.3412	0.2329	0.3056	0.1998
PL-Graph	0.4889	0.4015	0.6093	0.3961
fc7+linearSVM	0.3120	0.2721	0.2906	0.2481
fc7+RBF-SVM	0.3666	0.2994	0.3351	0.2617
fc7+NearestNeighbor	0.4195	0.3337	0.4159	0.3319
CNN-PDD	0.2333	0.2205	0.3401	0.2188
VGG-PDD-finetune	0.3269	0.2251	0.3198	0.2090
Fast-RCNN	0.4131	0.3227	0.2245	0.2810
Ours	**0.1115**	**0.1388**	**0.0758**	**0.1368**

ducted based the VGG-16 network that was pre-fine-tuned using object images in the target category. Because in the scope of weakly supervised learning, "simple" methods are usually insensitive to the over-fitting problem, we designed the last three baselines as follows. Given the pre-trained VGG-16 network that was used in our method, we directly used this network to extract *fc7* features from image patches of the annotated parts, and learned a linear SVM and a RBF SVM to classify target parts and background. Then, given a testing image, the three baselines brutely searched part candidates from the image, and used the linear SVM, the RBF SVM, and the nearest-neighbor strategy, respectively, to detect the best part. All the baselines were conducted using the same set of annotations for a fair comparison.

Evaluation metric

As mentioned in (Chen et al. 2014), a fair evaluation of part localization requires to remove the factors of object detection. Therefore, we used object bounding boxes to crop objects from the original images as the testing samples. Note that detection-based baselines (*e.g.* fast-RCNN, PL-Graph) may produce several bounding boxes for the part. Just as in (Chen et al. 2014; Oquab et al. 2015), we took the most confident bounding box per image as the localization result.

Table 4: Part center prediction accuracy on the PASCAL VOC Part Dataset

		bird	cat	cow	dog	horse	sheep	**Avg.**
Head	SS-DPM	9.0	39.4	39.0	49.4	38.0	36.8	35.3
	PL-DPM	18.6	28.0	33.5	32.5	20.8	0	22.2
	PL-Graph	11.9	53.6	34.6	56.4	20.1	0	29.4
	Fast-RCNN	21.2	36.4	40.1	34.3	24.0	40.5	32.7
	Ours	**64.5**	**85.0**	**65.4**	**81.8**	**77.1**	**64.1**	**73.0**
Torso		bird	cat	cow	dog	horse	sheep	**Avg.**
	SS-DPM	46.0	49.6	71.0	56.6	27.9	82.9	55.7
	PL-DPM	68.0	38.0	78.2	20.2	75.5	**92.8**	62.1
	PL-Graph	61.2	52.9	82.4	58.6	64.1	92.4	68.6
	Fast-RCNN	52.2	70.3	86.5	43.8	57.6	91.3	66.9
	Ours	**89.2**	**79.2**	**89.6**	**83.2**	**93.1**	91.6	**87.7**

3-shot learning

|		bird	cat	cow	dog	horse	sheep	**Avg.**
Head	SS-DPM	13.5	47.8	30.8	51.8	38.0	30.2	35.3
	PL-DPM	23.3	54.5	31.3	53.7	16.5	28.1	34.6
	PL-Graph	21.0	34.2	25.3	55.6	25.8	22.3	30.7
	Fast-RCNN	44.0	61.3	53.3	60.8	48.0	59.5	54.5
	Ours	**68.7**	**85.5**	**67.0**	**78.6**	**79.9**	**71.1**	**75.1**
Torso		bird	cat	cow	dog	horse	sheep	**Avg.**
	SS-DPM	54.0	51.5	66.8	33.7	50.7	88.6	57.5
	PL-DPM	68.7	65.0	77.2	29.8	77.2	89.0	67.8
	PL-Graph	26.6	65.1	85.0	65.1	79.3	89.0	68.4
	Fast-RCNN	70.5	76.3	**90.7**	68.9	70.7	93.5	78.4
	Ours	**87.3**	**84.3**	89.1	**84.6**	**91.4**	**94.3**	**88.5**

12-shot learning
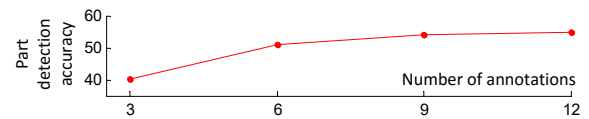
Figure 3: Performance with different numbers of annotations. We annotate 1–4 parts for each of the 3 part templates.

Given localization results of a part in a certain category, we used three evaluation metrics. 1) *Part detection*: a true part detection was identified based on the widely used "IOU ≥ 0.5" criterion (Girshick 2015); the part detection rate of this category was computed. 2) *Center prediction*: as in (Oquab et al. 2015), if the predicted part center was localized inside the true part bounding box, we considered it a correct center prediction; otherwise not. The average center prediction rate was computed among all objects in the category for evaluation. 3) The *normalized distance* in (Simon, Rodner, and Denzler 2014) is a standard metric to evaluate localization accuracy on the CUB200-2011 dataset. Because object parts may not have clear boundaries (*e.g.* the forehead of the bird), *center prediction* and *normalized distance* are more often used for evaluation of part localization.

Results and quantitative analysis

Table 1 lists the average children number of an AOG node at different layers. Fig. 4 shows the positions of the extracted latent pattern nodes, and part-localization results based on the AOGs. Given an image, we also used latent patterns in the AOG to reconstruct the corresponding semantic part based on the technique of (Dosovitskiy and Brox 2016), in order to show the interpretability of the AOG.

In Tables 2, 3, 4, 5, and 6, we compared the performance

Table 5: Part center prediction accuracy of 3-shot learning on the ILSVRC 2013 DET Animal-Part dataset.

		bird	frog	turt.	liza.	koala	lobs.	dog	fox	cat	lion	tiger	bear	rabb.	hams.	squi.	horse
Head	SS-DPM (Azizpour and Laptev 2012)	24.4	31.8	24.7	20.7	54.4	25.5	51.2	32.7	43.0	51.0	76.5	46.0	38.8	76.3	27.4	34.7
	PL-DPM (Li et al. 2013)	26.4	19.4	13.6	14.8	46.0	25.5	54.0	0	63.2	52.6	75.5	29.0	56.8	80.5	20.9	35.3
	PL-Graph (Chen et al. 2014)	25.4	43.5	18.8	16.3	56.7	27.0	56.9	37.4	52.3	67.0	77.0	48.0	63.9	84.7	45.6	39.5
	Fast-RCNN (Girshick 2015)	24.4	11.2	24.7	5.9	79.5	9.2	21.3	62.1	63.7	75.3	69.9	49.5	60.1	71.6	57.2	16.3
	Ours	**88.6**	**84.7**	**57.8**	**84.4**	**95.8**	**41.1**	**90.5**	**94.3**	**81.3**	**87.6**	**92.3**	**84.0**	**96.2**	**96.7**	**90.2**	**74.2**
		zebra	swine	hippo	catt.	sheep	ante.	camel	otter	arma.	monk.	elep.	red pa.	gia.pa.	gold.		**Avg.**
	SS-DPM (Azizpour and Laptev 2012)	42.9	56.7	64.4	50.8	52.1	48.8	35.3	36.9	28.3	57.0	73.0	50.9	77.7	29.0		45.4
	PL-DPM (Li et al. 2013)	30.5	48.2	53.2	47.0	7.4	17.1	39.5	32.1	21.2	59.7	61.2	49.6	1.8	18.0		36.7
	PL-Graph (Chen et al. 2014)	36.2	43.3	42.6	38.9	47.2	46.5	38.6	48.7	26.3	59.7	59.9	42.5	84.1	17.0		46.4
	Fast-RCNN (Girshick 2015)	25.4	53.7	58.5	56.8	59.5	64.5	41.4	43.9	31.8	52.2	77.6	56.6	65.0	18.5		46.9
	Ours	**80.8**	**73.2**	**79.3**	**80.0**	**88.3**	**93.1**	**92.6**	**88.8**	**42.9**	**94.1**	**85.5**	**80.7**	**93.6**	**83.0**		**83.2**
Front legs		bird	frog	turt.	liza.	koala	lobs.	dog	fox	cat	lion	tiger	bear	rabb.	hams.	squi.	horse
	SS-DPM (Azizpour and Laptev 2012)	15.4	41.8	51.9	33.0	51.1	63.1	31.2	35.7	28.5	38.6	51.8	52.2	25.3	43.8	27.1	33.8
	PL-DPM (Li et al. 2013)	10.5	31.6	46.8	8.3	47.8	64.1	24.6	14.7	13.1	34.6	18.4	37.7	2.7	25.8	18.8	33.8
	PL-Graph (Chen et al. 2014)	0	14.3	54.4	14.7	38.0	57.3	23.2	21.0	23.8	27.6	32.6	32.6	17.3	35.2	27.1	27.1
	Fast-RCNN (Girshick 2015)	30.2	29.6	43.0	7.3	62.0	60.2	21.7	23.8	24.6	28.3	46.8	57.2	16.0	24.2	8.2	34.6
	Ours	**58.6**	**77.6**	**67.1**	**37.6**	**82.6**	**77.7**	**85.5**	**73.4**	**62.3**	**80.3**	**77.3**	**78.3**	**66.7**	**60.9**	**63.8**	**70.7**
		zebra	swine	hippo	catt.	sheep	ante.	camel	otter	arma.	monk.	elep.	red pa.	gia.pa.	gold.		**Avg.**
	SS-DPM (Azizpour and Laptev 2012)	39.1	28.8	42.8	42.6	47.9	59.6	42.2	38.9	6.3	54.8	58.1	40.3	50.4	-		40.6
	PL-DPM (Li et al. 2013)	18.5	16.1	21.0	0	0	33.1	33.6	10.5	1.0	48.7	31.5	14.5	26.0	-		23.7
	PL-Graph (Chen et al. 2014)	17.9	27.1	23.2	16.5	54.2	34.9	33.6	27.4	12.5	54.8	54.8	14.5	38.2	-		29.5
	Fast-RCNN (Girshick 2015)	37.1	40.7	18.1	39.1	17.7	29.5	45.7	17.9	12.5	29.6	71.0	25.8	67.2	-		32.3
	Ours	**83.4**	**69.5**	**59.4**	**76.5**	**77.1**	**89.8**	**87.9**	**43.2**	**61.5**	**75.7**	**84.7**	**61.3**	**74.8**	-		**71.2**

Table 6: Part center prediction accuracy of 12-shot learning on the ILSVRC 2013 DET Animal-Part dataset.

		bird	frog	turt.	liza.	koala	lobs.	dog	fox	cat	lion	tiger	bear	rabb.	hams.	squi.	horse
Head	SS-DPM (Azizpour and Laptev 2012)	29.9	51.8	26.0	28.9	72.6	27.7	53.1	66.4	64.2	76.3	82.1	68.0	66.1	81.9	37.7	27.9
	PL-DPM (Li et al. 2013)	11.9	32.4	24.0	17.8	55.3	22.0	37.9	46.9	53.4	62.9	67.9	53.0	67.2	76.3	22.3	35.3
	PL-Graph (Chen et al. 2014)	37.3	27.1	10.4	21.5	55.8	5.0	68.7	38.9	75.6	77.3	63.3	55.5	67.2	69.8	36.7	28.4
	Fast-RCNN (Girshick 2015)	55.7	37.6	58.4	63.7	85.1	9.2	54.5	73.0	64.2	90.2	84.7	58.0	85.2	82.8	78.1	44.7
	Ours	**87.1**	**86.5**	**59.1**	**80.0**	**94.9**	**44.7**	**88.2**	**88.6**	**83.9**	**90.2**	**89.8**	**91.5**	**90.7**	**97.7**	**94.4**	**72.6**
		zebra	swine	hippo	catt.	sheep	ante.	camel	otter	arma.	monk.	elep.	red pa.	gia.pa.	gold.		**Avg.**
	SS-DPM (Azizpour and Laptev 2012)	46.3	67.1	61.7	52.4	66.3	60.4	41.4	49.2	42.4	62.9	78.3	71.1	87.7	31.5		56.0
	PL-DPM (Li et al. 2013)	34.5	51.8	54.3	42.7	35.6	30.9	33.0	46.5	15.7	48.9	60.5	45.2	74.5	19.0		42.7
	PL-Graph (Chen et al. 2014)	48.0	43.3	72.3	36.8	50.3	41.5	54.4	60.4	34.3	66.7	57.2	68.0	86.4	18.5		49.2
	Fast-RCNN (Girshick 2015)	**62.1**	67.7	**88.8**	76.2	76.1	83.9	70.7	66.3	65.2	73.1	83.6	83.3	79.1	86.5		69.6
	Ours	51.4	81.7	**82.4**	**82.4**	**87.1**	**87.1**	**91.2**	**92.5**	**74.7**	**93.7**	**90.1**	**92.1**	**93.2**	**95.0**		**84.5**

of different baselines. Our method exhibited much better performance than other baselines that suffered from over-fitting problems. In Fig. 3, we showed the performance curve when we increased the annotation number from 3 to 12. Note that the 12-shot learning only improved about 0.9%–2.9% of center prediction over the 3-shot learning. This demonstrat-ed that our method was efficient in mining CNN semantics, and the CNN units related to each part template had been roughly mined using just three annotations. In fact, we can further improve the performance by defining more part tem-plates, rather than by annotating more part boxes for existing part templates.

Conclusions and discussion

In this paper, we have presented a method for incremental-ly growing new neural connections on a pre-trained CNN to encode new semantic parts in a four-layer AOG. Given an demand for modeling a semantic part on the fly, our method can be conducted with a small number of part annotation-s (even a single box annotation for each part template). In addition, our method semanticizes CNN units by associat-ing them with certain semantic parts, and builds an AOG as a interpretable model to explain the semantic hierarchy of CNN units.

Because we reduce high-dimensional CNN activation-s to low-dimensional representation of parts/sub-parts, our method has high robustness and efficiency in multi-shot learning, and has exhibited superior performance to other baselines.
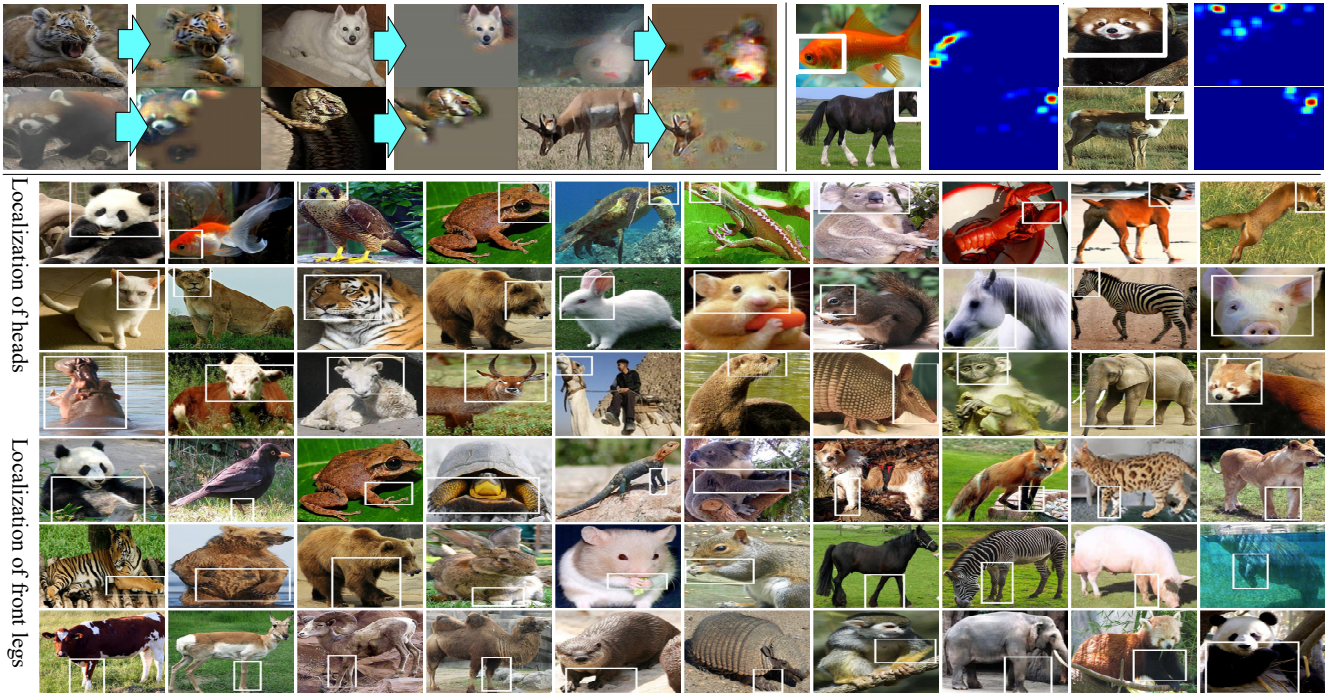
Acknowledgement

Figure 4: Image reconstruction based on the AOG (top left), heat maps corresponding to latent patterns at the 5-th conv-layers in the VGG-16 network (top right), and part localization performance to demonstrate the AOG interpretability (bottom).

Appendix

Parameters for latent patterns

In general, we use the notation of $\mathbf{P}_V$ to denote the central position of an image region Λ_V as follows.

$\mathbf{P}_{V^{\text{tmp}}}$	center of $\Lambda_{V^{\text{tmp}}}$
$\hat{\mathbf{P}}_{V^{\text{lat}}}$	center of $\hat{\Lambda}_{V^{\text{lat}}}$ obtained during part parsing
$\mathbf{P}_{V^{\text{unt}}}$	a constant center position of $\Lambda_{V^{\text{unt}}}$
$\overline{\mathbf{P}}_{V^{\text{lat}}}$	an AOG parameter, the center of the square deformation range of V^{lat}, i.e. V^{lat}'s "ideal" position without any deformation.
$\Delta\mathbf{P}_{V^{\text{lat}}}$	an AOG parameter, average displacement from V^{lat} to the parent V^{tmp}

Each latent pattern V^{lat} is defined by its location parameters $\{L_{V^{\text{lat}}}, D_{V^{\text{lat}}}, \overline{\mathbf{P}}_{V^{\text{lat}}}, \Delta\mathbf{P}_{V^{\text{lat}}}\} \subset \boldsymbol{\theta}$, where $\boldsymbol{\theta}$ is the set of AOG parameters. It means that a latent pattern V^{lat} uses a square[5] within the $D_{V^{\text{lat}}}$-th conv-slice/channel in the output of the $L_{V^{\text{lat}}}$-th CNN conv-layer as its deformation range. Each V^{lat} in the k-th conv-layer has a fixed value of $L_{V^{\text{lat}}} = k$. $\Delta\mathbf{P}_{V^{\text{lat}}}$ is used to compute $S^{\text{inf}}(\Lambda_{V^{\text{tmp}}}|\hat{\Lambda}_{V^{\text{lat}}})$. Given parameter $\overline{\mathbf{P}}_{V^{\text{lat}}}$, the displacement $\Delta\mathbf{P}_{V^{\text{lat}}}$ can be estimated as $\Delta\mathbf{P}_{V^{\text{lat}}} = \mathbf{P}^*_{V^{\text{tmp}}} - \overline{\mathbf{P}}_{V^{\text{lat}}}$, where $\overline{\mathbf{P}}^*_{V^{\text{tmp}}}$ denotes the average position among all ground-truth parts that are annotated for V^{tmp}. As a result, for each latent pattern V^{lat}, we only need to learn its conv-slice $D_{V^{\text{lat}}} \in \boldsymbol{\theta}$ and central position $\overline{\mathbf{P}}_{V^{\text{lat}}} \in \boldsymbol{\theta}$.

Scores of terminal nodes

The inference score for each terminal node V^{unt} under a latent pattern V^{lat} is formulated as

$$S_I(V^{\text{unt}}) = S_I^{\text{rsp}}(V^{\text{unt}}) + S_I^{\text{loc}}(V^{\text{unt}}) + S_I^{\text{pair}}(V^{\text{unt}})$$

$$S_I^{\text{rsp}}(V^{\text{unt}}) = \begin{cases} \lambda^{\text{rsp}} X(V^{\text{unt}}), & X(V^{\text{unt}}) > 0 \\ \lambda^{\text{rsp}} S_{none}, & X(V^{\text{unt}}) \leq 0 \end{cases}$$

$$S_I^{\text{pair}}(V^{\text{unt}}) = -\lambda^{\text{pair}} \underset{V^{\text{lat}}_{\text{upper}} \in \text{Neighbor}(V^{\text{lat}})}{\text{mean}} \left\| [\mathbf{P}_{V^{\text{unt}}} - \hat{\mathbf{P}}_{V^{\text{lat}}_{\text{upper}}}] - [\overline{\mathbf{P}}_{V^{\text{lat}}_{\text{upper}}} - \overline{\mathbf{P}}_{V^{\text{lat}}}] \right\|$$

The score of $S_I(V^{\text{unt}})$ consists of the following three terms: 1) $S_I^{\text{rsp}}(V^{\text{unt}})$ denotes the response value of the unit V^{unt}, when we input image I into the CNN. $X(V^{\text{unt}})$ denotes the normalized response value of V^{unt}; $S_{none} = -3$ is set for non-activated units. 2) When the parent V^{lat} selects V^{unt} as its location inference (i.e. $\hat{\Lambda}_{V^{\text{lat}}} \leftarrow \Lambda_{V^{\text{unt}}}$), $S_I^{\text{loc}}(V^{\text{unt}})$ measures the deformation level between V^{unt}'s location $\mathbf{P}_{V^{\text{unt}}}$ and V^{lat}'s ideal location $\overline{\mathbf{P}}_{V^{\text{lat}}}$. 3) $S_I^{\text{pair}}(V^{\text{unt}})$ indicates the spatial compatibility between neighboring latent patterns: we model the pairwise spatial relationship between latent patterns in the upper conv-layer and those in the current conv-layer. For each V^{unt} (with its parent V^{lat}) in conv-layer $L_{V^{\text{lat}}}$, we select 15 nearest latent patterns in conv-layer $L_{V^{\text{lat}}} + 1$, w.r.t. $\|\overline{\mathbf{P}}_{V^{\text{lat}}} - \overline{\mathbf{P}}_{V^{\text{lat}}_{\text{upper}}}\|$, as the neighboring latent patterns. We set constant weights $\lambda^{\text{rsp}} = 1.5, \lambda^{\text{loc}} = 1/3, \lambda^{\text{pair}} = 10.0, \lambda^{\text{unsup}} = 5.0$, and $\lambda^{\text{close}} = 0.4$ for all categories. Based on the above design, we first infer latent patterns corresponding to high conv-layers, and use the inference results to select units in low conv-layers.

Scores of AND nodes

$$S^{\text{inf}}(\Lambda_{V^{\text{tmp}}}|\hat{\Lambda}_{V^{\text{lat}}}) = -\lambda^{\text{inf}} \min\{\|\hat{\mathbf{P}}_{V^{\text{lat}}} + \Delta\mathbf{P}_{V^{\text{lat}}} - \mathbf{P}_{V^{\text{tmp}}}\|^2, d^2\}$$

where we set $d = 37$ pxls and $\lambda^{\text{inf}} = 5.0$.

References

Aubry, M., and Russell, B. C. 2015. Understanding deep features with computer-generated imagery. *In ICCV*.

Azizpour, H., and Laptev, I. 2012. Object detection using strongly-supervised deformable part models. *In ECCV*.

Branson, S.; Perona, P.; and Belongie, S. 2011. Strong supervision from weak annotation: Interactive training of deformable part models. *In ICCV*.

Chen, X.; Mottaghi, R.; Liu, X.; Fidler, S.; Urtasun, R.; and Yuille, A. 2014. Detect what you can: Detecting and representing objects using holistic models and body parts. *In CVPR*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. *In CVPR*.

Dosovitskiy, A., and Brox, T. 2015. Inverting visual representations with convolutional networks. *In arXiv:1506.02753*.

Dosovitskiy, A., and Brox, T. 2016. Inverting visual representations with convolutional networks. *In CVPR*.

Dosovitskiy, A.; Springenberg, J. T.; Riedmiller, M.; and Brox, T. 2014. Discriminative unsupervised feature learning with convolutional neural networks. *In NIPS*.

Felzenszwalb, P.; Girshick, R.; and McAllester, D. 2010. Cascade object detection with deformable part models. *In CVPR*.

Gallagher, P.; Tang, S.; and Tu, Z. 2015. What happened to my dog in that network: unraveling top-down generators in convolutional neural netowrks. *In arXiv:1511.07125v1*.

Ganin, Y., and Lempitsky, V. 2015. Unsupervised domain adaptation in backpropagation. *In ICML*.

Girshick, R. 2015. Fast r-cnn. *In ICCV*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. *In CVPR*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. 2012. Imagenet classification with deep convolutional neural networks. *In NIPS*.

Kumaran, D.; Hassabis, D.; and McClelland, J. L. 2016. What learning systems do intelligent agents need? complementary learning systems theory updated. *In Trends in Cognitive Sciences* 20(7):512–534.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *In Proceedings of the IEEE*.

Li, B.; Hu, W.; Wu, T.; and Zhu, S.-C. 2013. Modeling occlusion by discriminative and-or structures. *In ICCV*.

Liu, L.; Shen, C.; and van den Hengel, A. 2015. The treasure beneath convolutional layers: Cross-convolutional-layer pooling for image classification. *In CVPR*.

Lu, Y. 2015. Unsupervised learning on neural network outputs. *In arXiv:1506.00990v9*.

Mahendran, A., and Vedaldi, A. 2015. Understanding deep image representations by inverting them. *In CVPR*.

Oquab, M.; Bottou, L.; Laptev, I.; and Sivic, J. 2015. Is object localization for free? weakly-supervised learning with convolutional neural networks. *In CVPR*.

Pathak, D.; Kr ahenb uhl, P.; and Darrell, T. 2015. Constrained convolutional neural networks for weakly supervised segmentation. *In ICCV*.

Si, Z., and Zhu, S.-C. 2013. Learning and-or templates for object recognition and detection. *In PAMI*.

Simon, M., and Rodner, E. 2015. Neural activation constellations: Unsupervised part model discovery with convolutional networks. *In ICCV*.

Simon, M.; Rodner, E.; and Denzler, J. 2014. Part detector discovery in deep convolutional neural networks. *In ACCV*.

Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. *In ICLR*.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *In arXiv:1312.6034v2*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. *In arXiv:1312.6199v4*.

Tsogkas, S.; Kokkinos, I.; Papandreou, G.; and Vedaldi, A. 2015. Deep learning for semantic part segmentation with high-level guidance. *In arXiv:1505.02438v2*.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, In California Institute of Technology.

Yosinski, J.; Clune, J.; Bengio, Y.; and Lipson, H. 2014. How transferable are features in deep neural networks? *In NIPS*.

Zeiler, M. D., and Fergus, R. 2014. Visualizing and understanding convolutional networks. *In ECCV*.

Zhang, Q.; Song, X.; Shao, X.; Zhao, H.; and Shibasaki, R. 2014. Attributed graph mining and matching: An attempt to define and extract soft attributed patterns. *In CVPR*.

Zhang, Q.; Song, X.; Shao, X.; Zhao, H.; and Shibasaki, R. 2016. Object discovery: soft attributed graph mining. *In PAMI* 38(3).

Zhang, Q.; Wu, Y.-N.; and Zhu, S.-C. 2015. Mining and-or graphs for graph matching and object discovery. *In ICCV*.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2015. Object detectors emerge in deep scene cnns. *In ICRL*.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. *In CVPR*.

Zhu, L.; Chen, Y.; Lu, Y.; Lin, C.; and Yuille, A. 2008. Max-margin and/or graph learning for parsing the human body. *In CVPR*.