An Incremental Structured Part Model for Object Recognition

Xiao Bai\textsuperscript{a}, Peng Ren\textsuperscript{b,*}, Huigang Zhang\textsuperscript{a}, Jun Zhou\textsuperscript{c}

\textsuperscript{a}School of Computer Science and Engineering, Beihang University, Beijing 100191, China.
\textsuperscript{b}College of Information and Control Engineering, China University of Petroleum (Huadong), Qingdao 266580, China.
\textsuperscript{c}School of Information and Communication Technology, Griffith University, Nathan QLD 4111, Australia.

Abstract

In this paper, we describe how to build an incremental structured part model for object recognition. The proposed method explores both global structural information and multiple local features of objects for object model characterization. It uses part models to represent structure nodes, which encode the local information of an object. The parts are learned through a segmentation and clustering process, and are used to form the part models in terms of multiple feature fusion and multi-class SVMs. The structured part model is then constructed by correlating different parts through a deformable configuration. Furthermore, we present an incremental learning strategy, which learns a part model by using only a small number of training samples. Annotated images with high entropies are used to update the trained model. The advantage of our method is that it captures the inherent connections of the semantic parts of objects and characterizes the structural

\*Corresponding author.

Email addresses: baixiao.buaa@googlemail.com (Xiao Bai), pengren@upc.edu.cn (Peng Ren), huigang2010@gmail.com (Huigang Zhang), jun.zhou@griffith.edu.au (Jun Zhou)

Preprint submitted to Neurocomputing February 10, 2015
relationships between them. The proposed approach is evaluated on two datasets and demonstrates advantages over several state-of-the-art part-based methods in the literature.

**Keywords:** Part models, Structural relationship, Incremental learning

### 1. Introduction

It is widely acknowledged that object structure plays an important role in visual recognition. Therefore, a batch of structural modeling methods have been reported in the literature. Endres et al. [1] have pooled part detections within bottom-up regions and used a boosted classifier with sigmoid weak learners for recognition scoring. Maji et al. [2] have presented a method for semi-supervised discovery of semantically meaningful parts from pairwise correspondence annotations. Yang et al. [3] have represented images as close-loop graphs with superpixels being nodes, and then conducted saliency detection in a two-stage scheme to distinguish foreground salient objects from background regions. The obtained structure has given a possible solution to overcome the shortcoming of missing spatial information in statistical methods.

Graph based model is one type of structural models that bridges the low-level local invariant features with the high-level visual information in images [4, 5]. A graph structure not only naturally captures the correlations between visual components but also enables algebraically or statistically rigorous formulations [6]. In this scenario, Huang et al. [7] have presented a framework for unsupervised image categorization in which images containing specific objects are taken as nodes in a hypergraph and the task of image clustering is formulated as the problem of hypergraph partition. Lee et al. [8] have developed a graph based algorithm...
that models the interaction between familiar categories and unknown regions to discover novel categories in unlabeled images.

Part-based model [9, 10] is a specific class of graph based methods for characterizing images, where graph nodes represent visual parts endorsed with semantic interpretations and graph edges represent the spatial relations between visual parts. For example, a graph may represent a human being with graph nodes representing the body parts (i.e. head, torso, arms and legs etc.) and graph edges representing spatial relationships between two body parts. Using the part-based information, Girshick et al. [11] have developed a flexible grammar model for detecting people under a wide range of partial occlusion, pose, and appearance variability. Xiang et al. [12] have proposed a context-sensitive multiple Markov random fields (MRFs) approach to model and learn visual semantic contexts. This work is then extended to a joint framework for both object detection and object poses and part layouts estimation [13]. Sudderth et al. [14] have extracted local interest operators as parts, and used Dirichlet processes to automatically learn the number of parts underlying each object category. Furthermore, Amit et al. [15] have developed a patchwork of a part model for characterizing the interaction of overlapping parts.

Though part-based strategy enables an effective characterization of objects, it has two major disadvantages: a) the model accuracy of part-based methods may not be guaranteed in case of insufficient amount of training images, and b) the trained model is not easy to be updated given new training samples. One possible solution to address these shortcomings is to develop hierarchical or incremental learning schemes which are adapted to model variation. Wang et al. [16] have proposed a novel coarse-to-fine hair segmentation method which integrates
coarse segmentation and accurate segmentation processes. Pedersoli et al. [17] have presented a hierarchical part based model and a corresponding coarse-to-fine inference procedure to achieve robust object detection. Li et al [18] have developed a coarse-to-fine learning framework in which a novel automatic dataset collecting and model learning scheme, OPTIMOL, has been developed for refining online picture selection in an incremental manner. However, most of the existing hierarchical or incremental learning methods tend to be task-specific and are not formulated in a principled way.

The aim of our work is to develop a generic incremental structured part-based model, which not only enables an extensive hierarchical object characterization but also allows the model to be updated with respect to new training samples. Given a small number of training images, our method can get the proposed structured part model. Different from alternative state-of-the-art part-based methods, our method learns object parts without human annotation. This is done through an image segmentation strategy and a clustering algorithm that use both appearance and location information of the candidate parts. Furthermore, we use an incremental learning strategy in our method, which makes our model flexible to the diversity of images and requires less images for training.

One noteworthy observation is that a most recently proposed method, namely submodular object recognition [19], has achieved the best classification performance on benchmark evaluation. Unlike the submodular model which conducts object recognition at pixel level, our framework emphasizes the exploration of object part relationships and the ability of model updates. Nevertheless, the interesting idea of submodular object recognition provides insights for most state-of-the-art models, including the proposed incremental structured part-based model,
to lift their performance.

This paper is extended from an earlier conference version in [20]. Comparing with the conference version, this paper has given complete and detailed description on the proposed method with in-depth analysis on why each step is required and effective. Furthermore, several new components have been introduced into the method. Firstly, we have introduced a segmentation method to the approach in the conference paper, so that object parts can be obtained automatically instead of by human labeling. Secondly, a clustering strategy has been proposed to get the part models through those disorganized segments. It is a new systematic solution to bound the segmentation method and the part model construction. The utilization of both appearance and location information of segmented parts ensures the accuracy of the part clustering results, which in turn builds more accurate part models. This clustering strategy is not limited to this paper as it can be applied to other tasks that need to get coincident parts from disorganized segments. Finally, we have significantly extended the experiments to demonstrate the effectiveness of each step of the proposed method.

2. Method Overview

In this section, we give an overview of the proposed method. A flow chart of the key steps to generate the structured part model is shown in Fig. 1.

The first key step of the proposed method is obtaining foreground objects and corresponding parts. To achieve this goal, a segmentation operation based on Graph Laplacian Energy [21] is used to achieve this goal. The segmentation operation is performed twice. In the first segmentation, foreground object is separated from the image background. In the second segmentation, selected objects is split
The next step is building the part model. A multi-feature fusion method is first used to represent parts as feature vectors. Then we use a clustering method to get the similar parts together and build a common model, which is based on the appearance and location information of these candidate parts. The similar parts in appearance and location will be clustered into the same category. We build the part model by finding the common features in the clusters. Here a multi-class support vector machine (SVM) is used to learn the part model. The details are described in Section 4.2.

Having obtained the part models, we effectively arrange them in a deformable configuration and establish a structured part model for representing objects. This
is done through characterizing the part relationships in the form of discrete part
models. Meanwhile, we also introduce a parameter that balances the contribution
of the part model and the structure model. It is flexible to the diversity of images
and also suitable to our incremental learning process.

Finally, an incremental learning strategy is proposed to update the initial model.
This step is to address the common dilemma that the number of training samples is
not sufficient to build a robust model, but large number of training samples lead to
considerable costs of human annotation. We introduce an incremental process for
the structured part model in Section 5. This method uses the entropies of model
outputs to decide which results can be used to update the model.

Through the above steps, the final incremental structured part model can be
built, and be used for object recognition. The modelling details are described in
the following sections.

3. Part Segmentation

One commonly accepted way for identifying parts for a part-based model is
to use human annotation which is slow and expensive. Instead, we introduce an
automatic part generation method starting by separating foreground objects from
background. This method is based on a two-level segmentation strategy to get the
candidate parts. We first segment one testing image into large regions, and select
the region associated with the foreground object. The selected object region is
then divided into object parts.

3.1. Object Segmentation from an Image

Image segmentation is a crucial step for object-oriented image analysis. In this
subsection, we describe how to segment an object from one image. This method
is based on hierarchical segmentation method proposed in [22], which exploits Graph Laplacian Energy (GLE, [21]) as a generic measure for segmentation. It can get reasonable amount of regions without need to set the region number manually.

Firstly, we calculate the gradients of the target image, and cut it into small connected regions via watershed segmentation [23]. These generated regions form the bottom level of our hierarchical structure. Secondly, these neighboring regions are merged into larger ones and generate a higher level of the hierarchical structure. The analytical method adopted here is GLE [21][22] in which each merging iteration merges the most similar pairs of neighboring regions. To efficiently implement the merging strategy on a large number of regions, a vector containing the simple RGB color and intensity gradient value is used to characterize a region.

We compute the normalized GLE value at each hierarchical level and use it to decide which level should be selected as a segmentation result. For a level with $M$ regions, we define the normalized GLE as

$$nGLE = \frac{m}{M} \sum_{i=1}^{M} \frac{LE(G_i)}{|m_i|}$$

(1)

where $G_i$ is the $i$th region consisting of $|m_i|$ watershed segmentations, $LE(G_i)$ is the corresponding GLE value, and $m$ is the total number of watershed segmentations. For detailed description for computing (1), please refer to [21] and [22].

A normalized GLE curve is displayed in Figure 2. This curve reaches the local minima at the levels exhibiting homogeneity. They correspond to the levels with lower normalized GLE compared to the adjacent levels. We use the greatest local minimum level (e.g. the level marked by the dash line in Figure 2) for obtaining the segmentation result, because it segments the image into the fewest
large components.

Through the above procedure, some large segmented components of a target image can be recovered. These components often correspond to the foreground objects and some background noises, as shown in the first segmentation results in Figure 3. We manually select the foreground components and form the foreground object.

![Figure 2: The normalized GLE value against the hierarchical level index. We choose the highest level that gives a local minimum as our segmentation result.](image)

3.2. Segmenting an Object into Parts

We further segment the foreground object components in order to get the object parts. The object parts are found by applying the Graph Laplacian Energy method described in Section 3.1 again to the foreground components. Specifically, we compute the GLE only for the object components and segment them by searching for the local minima of $nGLE$. In this step, each object component is treated as a hierarchical structure and the level corresponding to the greatest low-
est local minimum generates the parts for an object. The results of this process are illustrated in the second segmentation result in Figure 3.

The first segmentation described in Section 3.1 cuts an image into large components, which are foreground objects and background noises. The second segmentation described in Section 3.2 is performed on the foreground components to get the object parts. It should be mentioned that the obtained parts are not very precise due to the limitation of the segmentation method. However, the part description and model updating methods introduced in the following sections can help compensate this disadvantage. Specifically, in the next section, we will discuss clustering the object parts by using both appearance and location similarity. The clustered regions are considered as corresponding parts of the object and used to form the part model.
4. Structured Part Model

In this section, we first represent candidate parts using multi-feature fusion method (Section 4.1). Then, we use the parts to construct the separate part models (Section 4.2) and correlate them to form the structured part model (Section 4.3).

The proposed model is a combination of both statistical and structural pattern recognition methods. Different parts of objects in the same class normally share similar spatial relationships. For example, all cars have roof, body, front and tyre, and they follow similar spatial layouts. Therefore, we only need to recognize these four parts and model their spatial relationship in order to distinguish cars from other objects.

4.1. Part Description

As described in Section 3, we have obtained several parts of an object. Some part examples are displayed in Figure 4. Through this figure, we can see that the parts suffer from variations, such as color variation, shape variation, illumination variation, and viewpoint changes. Therefore, the part-based model construction algorithm must be robust enough to these variations.

We commence by representing each part as a feature vector through multi-feature fusion. From each of the segmented parts, SIFT, texture, color, and edge direction features are extracted. The SIFT features [24] are extracted in a grid-based manner, in which 128 dimensional features can be computed from magnitude and orientation values of samples in their neighborhoods. The texture descriptors [25] are computed at each pixel using a set of filter banks, and then modelled by the joint probability distribution of filter responses. To extract the color features, we use the LAB values [26] of densely sampled pixels. Using the
bag-of-words model [27], these three types of features are quantized into vectors with 1000, 256 and 128 dimensions, separately. Finally, the edge features are generated via standard Canny edge detector [28]. Then they are quantized into 8 unsigned bins according to their orientations.

Now we’ve got four independent feature types of the segmented parts. The next step is to concatenate them into a long vector through multi-feature fusion. We simply concatenate the four different types of features into a vector of length 1392 as shown in Figure 5. Note that the features of interest are collected from different domains (i.e. shape, color, texture), and thus each feature type keeps its own unique property and distribution, which is helpful for distinguishing objects.

4.2. Part Model Learning

In this subsection, we build a part-based model based on the object parts extracted from the training dataset. Here we propose a new part-based model con-
struction method, which uses both the appearance and the location information of the parts to establish our part-based models.

The method constitutes two main steps. First, all the extracted parts are represented as a vector of length 1392, as illustrated in Section 4.1. These vectors form the appearance information of the corresponding parts. Then the parts are grouped into $k$ clusters in terms of the semantic meanings that these clusters should have, such as roof, front, body, and tyre of a car. Here, $k$ denotes the number of part models for the object, which is set manually. After clustering the parts according to their appearance information, we get rough clusters with specific semantic meaning. However, there are still some false samples being clustered into a group that they do not belong to. One possible reason for this error is that although the clusters capture similar appearance of parts, such purely appearance-based characterization does not take the relative location of the parts into account. Furthermore, some parts in the clusters may be too rough for learning a part model. To address these problems, the location information is used to filter out these non-discriminative parts.

In the second step, each cluster is refined by using the location information to get more accurate part model. We assign each part with a vector pointed to the object centroid as illustrated by the blue arrows in Figure 1(E). To get the centroids, rectangle boxes around the segmented parts and object are drawn. Then
the centers of these boxes are chosen to be the centroids of corresponding parts and the object. The vector starts from the part centroids to the object center.

The similarity of these vectors can be measured by their lengths and angles. Concretely, we denote \( v_i \) as a two-dimensional vector specifying an anchor position for part \( i \) that is relative to the object centroid position, \( i = 1, 2, ..., k \) (\( k \) is the number of part clusters). Then the vectors’ similarity can be measured as
\[
s(v_i, v_j) = \exp(||v_i - v_j||/\sigma^2)
\]
By comparing the pairwise similarities, we can get a cluster center part, which has the highest similarity with all the other parts. Through the above process, those parts with low similarity with this cluster center will be eliminated.

Through the above process, we use both appearance and location similarity to measure different parts. The appearance similarity can be computed by using the 1392 dimensional feature vector described in Section 4.1. While the location similarity is achieved through the above vectors which represent relative positions between parts. These two criteria ensure that we can exactly get the ideal \( k \) part clusters. The selected parts within each cluster now agree both in appearance and location relative to the object centroid, as shown in Figure 6.

Up to now, we have got the part clusters of the object. These part clusters are then used for part-based model learning. Based on the part feature vectors introduced in Section 4.1, a multi-class support vector machine (SVM) is learned. Let \( k \) be the number of part classes, \( x_{ni} \) denotes the \( n \)-th training sample of part \( i \) \((i = 1, 2, ..., k)\), which is represented as a 1392 dimensional vector and \( y_{ni} \) denotes its part class label. The multi-class SVM generates a \( k \)-dimensional weight vector \( \{w^*_i\}_{i=1}^k \), with one weight for each class. Let \( W \) denote a matrix whose columns
To estimate $W$, we minimize the following loss function

$$W^* = \arg \min \sum_n \sum_{i=1}^k d(w_i^T x_{ni}, y_{ni}) + \gamma \sum_i \|w_i\|^2$$  

(2)

where $\gamma \geq 0$ is a tradeoff parameter that regularizes the model complexity, and is set to 0.8 by threefold cross-validation. $d(\cdot, \cdot)$ is the standard loss function of C-SVC method in LIBSVM [29].

By solving this optimization problem using LIBSVM, we can get the part classifiers of an object. Thus, when an unlabeled image is given, we can get the candidate parts and calculate their similarity scores according to the achieved $W^*$. Details can be found in the next subsection in which we also explore the structural relationship between these parts and construct the structured part model.

4.3. Structured Part Model Construction

In this section, we effectively arrange the achieved parts in a deformable configuration to represent an object. The structure model here is inspired by the
pictorial structure method [9].

Given an image, let $p_i(l_i)$ be a function measuring the degree of part similarity when part $i$ is placed at location $l_i$. Let $p_{ij}(l_i, l_j)$ be a function measuring the degree of deformation when part $i$ is placed at location $l_i$ and part $j$ is placed at location $l_j$. We formulate the problem of matching a structured part model to an image as a statistical function to be maximized

$$L^* = \arg \max_L \left( \sum_{i=1}^{k} p_i(l_i) + \lambda \sum_{(i,j) \in E} p_{ij}(l_i, l_j) \right)$$

where $L = \{l_1, l_2, ..., l_k\}$ is the set of locations of the $k$ object parts, $E$ is the set of all the part pairs. This function maximizes the sum of the matching similarities $p_i(l_i)$ of each individual part and the deformation similarity $p_{ij}(l_i, l_j)$ for a connected pair of parts. We refer to $\sum_{i=1}^{k} p_i(l_i)$ and $\sum_{(i,j) \in E} p_{ij}(l_i, l_j)$ as model 1 and model 2, respectively. Here model 1 can be thought of as a standard part model which is measured only by the results of parts matching of one image to the model. Model 2 can be considered as a structure model that only measures the location relationship between two parts. $\lambda$ is a parameter that balances the contribution of models 1 and 2. It leads to the generalization of [9] to a more flexible setting and is self-adaptive through the incremental process.

We use a sliding window method to detect parts in an image and to compute $p_i(l_i)$. The sliding window method can get candidate patches by searching the testing images at different scales. Then we can get the matching scores of these patches by using the learned multi-class SVM classifier $W^*$. The scores can be transformed into similarities $p_i(l_i)$ by fitting a sigmoid function [30]. To compute $p_{ij}(l_i, l_j)$, we use the same method as [9] to calculate the degree of deformation, and fit it to $(0, 1]$ via an exponential function.
The proposed structured part model combines both local features and global structure of objects. Furthermore, it balances between part identification and object structure such that it is robust to missing or misclassified parts in an image. Note that the learning process only uses a small number of training data and renders an efficient computation. To make the model more robust, we present an incremental learning algorithm in the next section.

5. Incremental Learning with Structured Part Model

Normally, a large number of representative images are required to train a robust model because of the large variation of test images. In our work, we aim at using a small number of training images to train the structured part model introduced in Section 4. To this end, we employ an incremental learning approach to propagate the initial knowledge.

5.1. Incremental Learning

Given a very small number of training images of an object class, our algorithm learns the optimal structured part-based model $L^*$ that best describes this class using the approaches introduced in Section 4. Here we present an incremental process to iteratively update $L^*$, which further improves the robustness of the proposed method.

We randomly separate testing images into several batches and feed them sequentially into the system. Each batch is processed in an iteration that updates the current model. After one iteration, the batch of images can be classified as positive images, negative images and candidate images used for model updating. Our incremental process is performed when a new batch comes in. It continuously classifies the incoming images and learns a more robust model.
Intuitively, we endow the matching similarity $p_i$ and deformation similarity $p_{ij}$ with a probabilistic meaning. Here the similarities $p_i$ and $p_{ij}$ are the same with that described and computed in Section 4.3. On each image batch, we compute the probability that the current optimal structured part-based model matches the images using (3). The model update is dependent on the image matching results. Images with low matching probability are discarded, while the rest are divided into two sets based on the entropy value generated from the following equation

$$H(I) = -\sum_i p_i \ln p_i - \lambda \sum_{(i,j)\in E} p_{ij} \ln p_{ij}$$

(4)

According to Shannon’s entropy theory, equation (4) relates to the amount of uncertainty of an image $I$. High entropy indicates high uncertainty of an image, which in turn suggests possible new structures. Thus, we choose those images with high entropies for model updates. Images with low entropies are classified to be positive images. In our experiments, we set 1.5 as the entropy threshold.

Above all, the proposed incremental learning method is an iterative process that continuously classifies images with high accuracy and learns a more robust object model. The basic process of our incremental learning method is illustrated in Figure 7.

5.2. Model Update Details

The model update follows the method introduced in Section 4. It allows refinement of the part classifiers and the corresponding structure model. When an image is classified to be with high entropy, it is used for model update. The updating details are described as follows.

First, for the samples used for updating, their parts that are matched with the previous model are already located through the previous iteration. Second, we get
the parts’ descriptors and the location relationships between these parts. Third, we use the above parts’ descriptors and their labels \((x_{ni}, y_{ni})\) to train new part models \(W^*\) based on equation (2). And we use the same method as [9] to calculate the degree of structure deformation \(p_{ij}(l_i, l_j)\), which leads to the structure model update.

At the same time, the weight parameter \(\lambda\) is updated iteratively to make the learned model more robust. In each iteration, the matching probabilities are calculated using \(L^*\). This can be achieved by setting \(\lambda\) to 0 and 100 (a large enough number) separately. Suppose that \(\varphi_i = \{x|x\text{ is an image belonging to the positive part using the model } i\}\), \(\varphi = \{x|x\text{ is an image belonging to the positive part using}\)
$L^*$, and $con_i$ represents the contribution of the model $i$ to $L^*$. Then

$$con_i = \frac{\#\{\varphi_1 \cap \varphi\}}{\#\{\varphi\}}, i = 1, 2$$  \hspace{1cm} (5)

$$\lambda = \frac{con_2}{con_1} = \frac{\#\{\varphi_2 \cap \varphi\}}{\#\{\varphi_1 \cap \varphi\}}$$  \hspace{1cm} (6)

The relation in equation (6) determines the weights of the part model and the structure model. By calculating $\lambda$ in each batch, a more refined model can be achieved. The incremental process updates the part models $W^*$, the structure deformation and the balance parameter $\lambda$ between them. The implementation of our algorithm is described in Algorithm 1.

**Algorithm 1** Incremental Structured Part Model

**Input:** Set of $N$ positive images ($N$ is a small number), set of new unlabeled images, part number $n$, and weight $\lambda=1$.

**Output:** Set of classified positive images, and the final Structured Part Model

**Initialize** The $n$ part in each training image (Section 3)

**Repeat**

**Learn** Calculate the features of each part in the latest input images and train the SVM models. (Section 4.1)

Learn the Structured Part Model. (Section 4.3)

**Classify** Classify images using the current Structured Part Model. (Section 4.3)

**Incremental** Use the images with high entropy for model update. (Section 5)

**until** User satisfied or images exhausted.
6. Experimental Results

In this section, we validate the effectiveness of the proposed incremental structured part model. All experiments were implemented on a desktop with an Intel Core 2 Duo 2.40-GHz processor. Firstly, we compare the performance of our model with the baselines in the case of one or more components missed, and show how each component of our method improves performance. Specially, we focus on the performance of our incremental process and show how it achieves good results with a small number of training images (Section 6.1). Secondly, we evaluate the proposed model for object recognition. Our method is compared with state-of-the-art classification methods such as models by Gritfin et al. [31], ScSPM [32], and LLC [33] etc.(Section 6.2) Finally, we also apply our model to detection tasks, and show its advantages over some other part-based models [34, 35]. Furthermore, we also test our method on pedestrian detection, and show its effectiveness on general detection tasks (Section 6.3).

The experiments are performed on two widely used datasets, Caltech-256 [31] and Pascal VOC 2007 [34]. The Caltech-256 dataset contains 30,607 images in 256 categories, with each class containing at least 80 images. The Pascal VOC 2007 dataset consists of 9,963 images from 20 classes. Objects in this dataset reside in cluttered scenes with a high degree of variation in viewing angle, illumination and object appearance. We resize each image to less than $300 \times 300$ pixels with the aspect ratio unchanged. We used all classes in these two datasets for the experiments.
6.1. Incremental structured part model evaluation

In the first experiment, we test the time consuming performance of the proposed incremental process. By analyzing the computing time of this process, we can terminate it at an optimal stage. This experiment is done on the Caltech-256 dataset. We choose 10 positive images from each category as the initial training data. For each category, we add 20 unlabeled images at each incremental iteration to update the model. The average classification accuracy and time consuming results are shown in Figure 8.

![Different Iteration Performance](image)

Figure 8: Performance of the incremental learning process on the Caltech-256 dataset. The horizontal axis represents the number of iterations. The vertical axis on the left is the average classification accuracy of all 256 categories. The vertical axis on the right corresponds to the average time consuming over the iteration number.

In Figure 8, the blue line represents how the average classification accuracy changes when iteration number increases, and the red line shows how time consumes with the growth of iteration number. Through the table, we can see the
classification accuracy does not always increase when the number of iteration increases. Nevertheless, the time consumption keeps increasing linearly. Hence, the update iteration can be terminated after a small number of steps to guarantee both efficiency and accuracy. In the following experiments, we manually set 5 as the number of the iteration.

Next we test the proposed model for image classification on both the Caltech-256 and Pascal VOC 2007 datasets. We randomly choose 5, 10, 15, 20, 25 and 30 training images per class to validate the effectiveness of the proposed method. We consider three baselines for comparison: 1) a standard part model, 2) a structure model, which is also the basic pictorial structure method [9], and 3) our structured part model without an incremental process. The results are shown in Figure 9. It is clear that our incremental structured part model outperforms the baselines by nearly 10 percent. The proposed model is very stable on both datasets when different sizes of training data are used. When the training sample number is 5, our method achieves classification accuracies that are nearly 10 and 20 percent greater than the alternatives, separately.

![Figure 9](image.png)

**Figure 9**: The average classification results of all the categories in the Caltech-256 dataset (*left*) and Pascal VOC 2007 dataset (*right*), when different training sizes are used.
Note that to train a robust model in the training stage, the non-incremental method requires a large number of training images. However, when the amount of training samples are limited, it cannot provide enough information for the model. And the classification results are not satisfying with limited amount of training images, as can be seen in the situations of 5 and 10 training images in Figure 9.

The reason for which our model achieves good performance under small number of training images is the effect of the incremental process. By choosing those images with high entropies, large amount of novel information can be acquired for model updating. The effect of the incremental process is three-fold. Firstly, it refines the multi-SVM part model after several incremental iterations. Secondly, this process can refine the structural model both in orientation and location relationships. Thirdly, each iteration updates the parameter $\lambda$ in (3), which leads to a refined global model.

Figure 10 shows some example images with high classification accuracy in the Caltech-256 dataset. We have also tackled those image data with missing parts. The results show that most of them can be classified correctly, which proves the robustness of the proposed method.

In Figure 11, we also show some example images with low classification accuracy in the Caltech-256 dataset. In these categories, our method did not perform well mainly for two reasons. First, the structural characteristics are not clear for some categories, and objects with different sizes are randomly distributed in the image, as shown in *dice* and *mussels* case. Second, some categories have significant intra-class variations, such as the *bat* and *bathtub* examples. The above two reasons make the category structure model not easy to build, leading to low classification accuracy.
In summary, we have tested our proposed model in this subsection and shown that only a small number of training images are required for training a refined model. In the next section, we will evaluate this structured part model and empirically investigate how it works to improve the performance of classification and detection.
6.2. Classification performance

In this subsection, we evaluate the performance of the proposed model for image classification and compare it with state-of-the-art classification methods. The classification performance is evaluated using the Average Precision (AP) measure. It computes the area under the Precision/Recall curve, in which higher score means better performance.

We first make this comparison against five other classification methods [34, 33, 36, 37, 19] on the Pascal 2007 dataset. The method presented in [34] is one of the best methods for Pascal 2007 classification, where complex local features and kernels are used to get good results. Locality-constrained linear coding (LLC [33]) utilizes the locality constraints to project each descriptor into its local-coordinate system, and then generates the final image representation by max pooling. Perronnin et al. [36] have improved the fisher kernel for large-scale image classification. Chen et al. [37] have introduced a hierarchical matching framework based on bag-of-words representation. Recently, Zhu et al. [19] have presented a novel object recognition framework based on the so-called submodular, generated by bottom-up processes and mid-level cues in an unsupervised manner. To make a fair comparison, all these methods are modelled on the standard training dataset provided by Pascal 2007.

Table 1 shows the classification accuracy on all 20 classes. The latest method proposed in [19] has achieved the highest accuracy in most classes. However, our method still achieve considerable results with other state-of-the-art classification methods, especially those with similar shapes such as bicycle and motorbike, cat and dog, cow and sheep. The results show that our semantic part model is capable of extracting features and their structural relationships in order to distinguish...
similar objects. It should be mentioned that the above classification methods do not involve in dealing with detection tasks. Whereas, our method can also achieve good detection outcome on some widely-used datasets which will be proved in Section 6.3. As the proposed method is based on the structural features, it is somehow influenced by the appearance and viewpoint of an object. For example, our method does not achieve the best accuracy on the car and person categories due to their within-class pose variations.

<table>
<thead>
<tr>
<th>Category</th>
<th>aero</th>
<th>bicyc</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASCAL 07 Best [34]</td>
<td>77.5</td>
<td>63.6</td>
<td>56.1</td>
<td>71.9</td>
<td>33.1</td>
<td>60.6</td>
<td>78</td>
<td>58.8</td>
<td>53.5</td>
<td>42.6</td>
</tr>
<tr>
<td>LLC [33]</td>
<td>74.8</td>
<td>65.2</td>
<td>50.7</td>
<td>70.9</td>
<td>28.7</td>
<td>68.8</td>
<td>78.5</td>
<td>61.7</td>
<td>54.3</td>
<td>48.6</td>
</tr>
<tr>
<td>Perronnin [36]</td>
<td>75.7</td>
<td>64.8</td>
<td>52.8</td>
<td>70.6</td>
<td>30.0</td>
<td>64.1</td>
<td>77.5</td>
<td>55.5</td>
<td>55.6</td>
<td>41.8</td>
</tr>
<tr>
<td>Chen [37]</td>
<td>76.7</td>
<td>74.7</td>
<td>53.8</td>
<td>72.1</td>
<td>40.4</td>
<td>71.7</td>
<td>83.6</td>
<td>66.5</td>
<td>52.5</td>
<td>57.5</td>
</tr>
<tr>
<td>Zhu [19]</td>
<td><strong>83.7</strong></td>
<td><strong>82.5</strong></td>
<td><strong>63.3</strong></td>
<td><strong>77.3</strong></td>
<td><strong>58.0</strong></td>
<td><strong>80.2</strong></td>
<td><strong>89.4</strong></td>
<td><strong>68.8</strong></td>
<td><strong>63.1</strong></td>
<td><strong>63.7</strong></td>
</tr>
<tr>
<td>Proposed method</td>
<td>77.1</td>
<td>73.0</td>
<td>54.8</td>
<td>75.2</td>
<td>37.2</td>
<td>70.3</td>
<td>72.4</td>
<td>65.7</td>
<td>60.6</td>
<td>50.8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>table</th>
<th>dog</th>
<th>horse</th>
<th>mbike</th>
<th>person</th>
<th>plant</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASCAL 07 Best [34]</td>
<td>54.9</td>
<td>45.8</td>
<td>77.5</td>
<td>64.0</td>
<td>85.9</td>
<td>36.3</td>
<td>44.7</td>
<td>50.9</td>
<td>79.2</td>
<td>53.2</td>
</tr>
<tr>
<td>LLC [33]</td>
<td>51.8</td>
<td>44.1</td>
<td>76.6</td>
<td>66.9</td>
<td>83.5</td>
<td>30.8</td>
<td>44.6</td>
<td>53.4</td>
<td>78.2</td>
<td>53.5</td>
</tr>
<tr>
<td>Perronnin [36]</td>
<td>56.3</td>
<td>41.7</td>
<td>76.3</td>
<td>64.4</td>
<td>82.7</td>
<td>28.3</td>
<td>39.7</td>
<td>56.6</td>
<td>79.7</td>
<td>51.5</td>
</tr>
<tr>
<td>Chen [37]</td>
<td>62.8</td>
<td>51.1</td>
<td>81.4</td>
<td>71.5</td>
<td>86.5</td>
<td>36.4</td>
<td>55.3</td>
<td>60.6</td>
<td>80.6</td>
<td>57.8</td>
</tr>
<tr>
<td>Zhu [19]</td>
<td><strong>67.4</strong></td>
<td><strong>53.5</strong></td>
<td><strong>86.4</strong></td>
<td><strong>82.7</strong></td>
<td><strong>90.5</strong></td>
<td><strong>48.4</strong></td>
<td><strong>62.0</strong></td>
<td><strong>67.9</strong></td>
<td><strong>87.2</strong></td>
<td><strong>71.5</strong></td>
</tr>
<tr>
<td>Proposed method</td>
<td>57.5</td>
<td>49.3</td>
<td>75.7</td>
<td>72.9</td>
<td>77.2</td>
<td>42.1</td>
<td>47.9</td>
<td>51.5</td>
<td>80.6</td>
<td>58.6</td>
</tr>
</tbody>
</table>

Furthermore, we also test the alternative methods on Caltech-256 dataset, in which we used 5, 15, and 30 training images per class. We compare our method with four alternative methods [31, 32, 33, 36]. Detailed results are given in Table 4, where our method leads the performance.
Table 2: Image classification results on Caltech-256 dataset.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>5 training</th>
<th>15 training</th>
<th>30 training</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gritfin et al. [31]</td>
<td>18.40</td>
<td>28.30</td>
<td>34.10</td>
</tr>
<tr>
<td>ScSPM [32]</td>
<td>-</td>
<td>27.73</td>
<td>34.02</td>
</tr>
<tr>
<td>LLC [33]</td>
<td>-</td>
<td>34.36</td>
<td><strong>41.19</strong></td>
</tr>
<tr>
<td>Perronnin [36]</td>
<td>-</td>
<td>34.70</td>
<td>40.80</td>
</tr>
<tr>
<td>Proposed method</td>
<td><strong>31.15</strong></td>
<td><strong>35.22</strong></td>
<td>36.87</td>
</tr>
</tbody>
</table>

6.3. Object Detection performance

In this experiment, we evaluate our system on the Pascal 2007 dataset for object detection. The proposed method is compared with the maximum and median results of Pascal VOC 2007 [34], the UoCTTI method [35], and Endres et al. [1], which are also based on part model. To guarantee consistency with PASCAL detection criterion, a detection is considered correct when it overlaps more than 50% with a groundtruth bounding box. The training samples are given by the Pascal 2007 training dataset. In our method, we use 5 samples for initial training, and other samples for incremental training.

Table 3 summarizes the AP for all the 20 categories in this dataset. For each class, the method obtaining the highest AP is marked in bold. As shown in Table 3, the proposed method has outperformed the median results of 18 categories. Furthermore, our method has achieved the highest AP score in 7 categories, and the second highest score in 8 categories.

The categories with higher scores share the common property that they all have clearly predictable visual properties, such as distinctive parts and relatively fixed spatial arrangement of parts. Thus our structured part model can effectively
characterize these properties and achieve good detection performance. For those categories with significant variation in structure (e.g. table and horse), the proposed method is less effective.

Table 3: Detection results on Pascal VOC 2007 dataset. AP measure (%) is shown for each category.

<table>
<thead>
<tr>
<th>Category</th>
<th>aero</th>
<th>bicyc</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASCAL 07 Max</td>
<td>26.2</td>
<td>40.9</td>
<td>9.8</td>
<td>9.4</td>
<td><strong>21.4</strong></td>
<td><strong>39.3</strong></td>
<td><strong>43.2</strong></td>
<td>24.0</td>
<td>12.8</td>
<td>14.0</td>
</tr>
<tr>
<td>PASCAL 07 Median</td>
<td>15.6</td>
<td>26.2</td>
<td>4.3</td>
<td>2.5</td>
<td>3.4</td>
<td>19.7</td>
<td>29.4</td>
<td>9.9</td>
<td>9.6</td>
<td>10.0</td>
</tr>
<tr>
<td>UoCTTI</td>
<td>20.6</td>
<td>36.9</td>
<td>9.3</td>
<td>9.4</td>
<td><strong>21.4</strong></td>
<td>23.2</td>
<td>34.6</td>
<td>9.8</td>
<td>12.8</td>
<td>14.0</td>
</tr>
<tr>
<td>Endres et al. [1]</td>
<td><strong>33.7</strong></td>
<td>36.4</td>
<td><strong>10.6</strong></td>
<td>9.5</td>
<td>18.3</td>
<td>37.2</td>
<td>35.6</td>
<td><strong>26.7</strong></td>
<td>8.5</td>
<td>12.5</td>
</tr>
<tr>
<td>Proposed method</td>
<td>21.1</td>
<td><strong>42.0</strong></td>
<td>8.8</td>
<td><strong>10.2</strong></td>
<td>21.0</td>
<td>33.7</td>
<td>39.4</td>
<td>17.5</td>
<td><strong>15.6</strong></td>
<td><strong>15.8</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>table</th>
<th>dog</th>
<th>horse</th>
<th>mbike</th>
<th>person</th>
<th>plant</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASCAL 07 Max</td>
<td>9.8</td>
<td>16.2</td>
<td><strong>33.5</strong></td>
<td>37.5</td>
<td>22.1</td>
<td>12.0</td>
<td><strong>17.5</strong></td>
<td>14.7</td>
<td><strong>33.4</strong></td>
<td>28.9</td>
</tr>
<tr>
<td>PASCAL 07 Median</td>
<td>6.3</td>
<td>9.5</td>
<td>19.8</td>
<td>22.4</td>
<td>12.1</td>
<td>4.5</td>
<td>4.7</td>
<td>9.6</td>
<td>16.3</td>
<td>23.9</td>
</tr>
<tr>
<td>UoCTTI</td>
<td>0.2</td>
<td>2.3</td>
<td>18.2</td>
<td>27.6</td>
<td>21.3</td>
<td>12.0</td>
<td>14.3</td>
<td>12.7</td>
<td>13.4</td>
<td>28.9</td>
</tr>
<tr>
<td>Endres et al. [1]</td>
<td><strong>11.4</strong></td>
<td><strong>18.9</strong></td>
<td>24.3</td>
<td>36.1</td>
<td>17.0</td>
<td>8.2</td>
<td>15.4</td>
<td><strong>15.2</strong></td>
<td>26.5</td>
<td>29.0</td>
</tr>
<tr>
<td>Proposed method</td>
<td>5.5</td>
<td>10.6</td>
<td>17.3</td>
<td><strong>40.9</strong></td>
<td>20.2</td>
<td><strong>13.1</strong></td>
<td>16.9</td>
<td>12.5</td>
<td>22.6</td>
<td><strong>32.6</strong></td>
</tr>
</tbody>
</table>

To further validate the effectiveness of our method, we also conduct experiments on pedestrian detection. The experiment is done on the INRIA-Person dataset [38], which has been used to evaluate alternative human detectors. Different from other person dataset, the INRIA dataset has fairly high resolution pedestrians. In the training part, there are 1218 negative images, and 614 positive images with 1208 pedestrians. In the testing part, there are 453 negative images, and 288 positive images with 566 pedestrians.

To analyze the effectiveness of the method, we compare our model with other pedestrian detectors [39, 35, 38, 40, 10, 41, 42]. All these methods are trained and tested on the INRIA-Person dataset. To make fair comparisons, we use the
same evaluation strategy proposed in [43]. A detection is considered correct when it overlaps more than 50% with a groundtruth bounding box. Different from alternative methods, we use only 5 labeled samples for initial training, and other samples for incremental training.

We use the false positives per image (FPPI) rate to evaluate the detection results. The final detection results are computed by averaging miss rate at 0.1 FPPI, which is shown in Table 4. Although our method does not specifically aim at pedestrian detection, it can still achieve good results on the dataset. Like those in [35] and [10], the structure model in our method ensures the overall integrity of pedestrian model. It provides more information to detect person correctly. On the other hand, the incremental learning process enables updates for the pedestrian model. The disadvantage of our method on pedestrian detection is that the multiple features are redundant to describe a person. When the image is with low resolution, it is better to represent a person by using simple features.

7. Conclusion

In this paper, we have introduced a general method for learning a novel incremental structured part model. We have commenced by automatically identifying object regions through a segmentation and clustering scheme without human annotation. We have combined information obtained both from semantic parts and their structural relation for constructing the model. Furthermore, we have developed an incremental process to refine the model iteratively, which makes the proposed method more flexible. Our method requires only a small number of training images to achieve good recognition performance. In future work, we will investigate more accurate models for training the part classifier and promote the
Table 4: Comparison of evaluated Pedestrian Detectors on INRIA-Person dataset.

<table>
<thead>
<tr>
<th>Pedestrian Detectors</th>
<th>Features</th>
<th>Learning</th>
<th>Miss Rate at 0.1 FPPI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>gradient hist.</td>
<td>color</td>
<td>texture</td>
</tr>
<tr>
<td>VJ [39]</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LATSVM-V1 [35]</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOG [38]</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOGLBP [40]</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>LATSVM-V2 [10]</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>FEATSYNTH [41]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FPDW [42]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Proposed method</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

performance of the whole model.

8. Acknowledgements

This work was supported in part by NSFC projects (No. 61370123, 61105002 and 61105005), Qingdao Fundamental Research Project (No. 13-1-4-256-jch), the Open Project Program of the National Laboratory of Pattern Recognition (NLPR), and Australian Research Councils DECRA Projects funding scheme (project ID DE120102948).

References


