

# **OBJECT DETECTION WITH LARGE INTRA-CLASS VARIATION**

---

A Thesis presented to  
the Faculty of the Graduate School  
at the University of Missouri

---

In Partial Fulfillment  
of the Requirements for the Degree  
Master of Science

---

by  
GUANG CHEN  
Dr. Tony X Han  
DECEMBER 2011

The undersigned, appointed by the Dean of the Graduate School, have examined the thesis entitled:

OBJECT DETECTION WITH LARGE INTRA-CLASS VARIATION

presented by Guang Chen,  
a candidate for the degree of Master of Science and hereby certify that, in their opinion,  
it is worthy of acceptance.

Tony X Han

---

Dr.Tony X Han

Yunxin Zhao

---

Dr. Yunxin Zhao

Zhihai He

---

Dr.Zhihai He

## **ACKNOWLEDGMENTS**

I would like to express my appreciation to my committee: Dr. Tony X Han, Dr. Yunxin Zhao, Dr. Zhihai He. Thanks for giving me the opportunity to present my research work of the past two years. Special thanks to Dr. Han for his time, patience, and understanding. It has been an honor to work with you. Also, thanks to the group members: Xiaoyu Wang, Xutao Ly, Wei Gong, Shuai Tang, Yan Li, Miao Sun. I have learned so much from you. There are not enough words to describe your priceless advices. Finally, it is a big pleasure to thank my parents who give me the moral support, without their help I might not have done my research and projects so well and on time.

# TABLE OF CONTENTS

<b>ACKNOWLEDGMENTS</b>	<b>ii</b>
<b>LIST OF FIGURES</b>	<b>v</b>
<b>ABSTRACT</b>	<b>viii</b>
<b>CHAPTER</b>	
<b>1 Introduction</b>	<b>1</b>
<b>2 Min-Max Model Adaptation</b>	<b>6</b>
2.1 Introduction	6
2.2 Cross-Entropy Error as the Adaptation Cost	7
2.3 Model Adaptation as Min-Max Optimization	9
2.3.1 Constructing a random cover from the adaptation dataset	10
2.3.2 Min-Max objective function	10
2.3.3 Min-Max optimization	12
2.4 Experiments	13
2.4.1 Image representation and experimental setup	14
2.4.2 Performance comparison between different adaptation methods	15
2.4.3 INRIA dataset	16
2.4.4 TUD-pedestrian dataset	17
2.4.5 PASCAL dataset	17
2.4.6 ImageNet dataset	19
2.4.7 Experiments discussion	20
2.5 Conclusions	21
<b>3 Divide with Global Classifier, Conquer with Local Adaptation</b>	<b>23</b>

3.1	Introduction . . . . .	23
3.2	Divide by Global Classification . . . . .	25
3.3	Clustering - Adjusting Model Complexity . . . . .	26
3.3.1	Distance Metrics . . . . .	27
3.3.2	Clustering Algorithm . . . . .	27
3.3.3	Parameter Selection & Fast Approximation . . . . .	29
3.4	Local Adaptation - Enhancing Learning Capacity . . . . .	31
3.5	Experiments . . . . .	32
3.5.1	Experimental Setup . . . . .	34
3.5.2	In-House dataset . . . . .	35
3.5.3	Caltech dataset . . . . .	36
3.5.4	INRIA dataset . . . . .	37
3.5.5	Computational Complexity in Testing . . . . .	39
3.6	Conclusion and Future Work . . . . .	39
<b>4</b>	<b>Conclusion . . . . .</b>	<b>41</b>
	<b>BIBLIOGRAPHY . . . . .</b>	<b>42</b>

## LIST OF FIGURES

Figure		Page
2.1	The construction of the random cover of the adaptation dataset. Each small block represents $S^{(n)}_i$ , an element of the partition. The $G_j$ 's could have different size and overlap with each other. . . . .	11
2.2	From left to right, the first box gives the examples we use to train our generic human detector, all coming from INRIA dataset; the second to the fourth boxes give positive examples sampled from three adaptation dataset: the INRIA dataset, the TUD dataset, and the PASCAL dataset. . . . .	14
2.3	Evaluation of the effect of data cover number $N$ in our algorithm, experiment is on PASCAL dataset. We randomly set $N$ , and draw the curve when $N = 2, 4, 5, 10, 15$ , all comparing with Taylor adaptation method with optimal $\lambda$ . . . .	18
2.4	First two rows show positive and negative examples to train our bird detector, all extracted from ImageNet dataset; The third row gives some positive examples we labeled for our parrot dataset . . . . .	19
2.5	Evaluation of our method on multiple datasets, compared with other adaptation method. In the curve of PASCAL dataset, the standard deviation bar represents the effect of data cover number $N$ on the performance of our algorithm. . . . .	21

2.6	Performance comparison between detectors, all the number are detection scores. The first row is from old detector; the second row is from adapted detector generated by Taylor expansion adaptation method with the optimal adaptation rate; the third row is from detector adapted by our proposed algorithm. In each dataset, we samples 3 positive and 3 negative examples. The score difference between positive and negative examples is larger, the performance is better. . . . .	22
3.1	A Toy Example of Two-class (“o” vs “x”) Classification Using Our Approach: A global classifier (blue solid line) and its boundaries (blue dotted lines) divide the data space into easy regions and hard regions. The ambiguous data in hard regions are clustered according to the data distribution, which automatically adjusts the model complexity. Each cluster of samples are classified using locally adapted classifier that avoids under training. The hybrid learning algorithm autonomously strikes a balance between model complexity and learning capacity. . . . .	24
3.2	Sample results of our automatic clustering on ambiguous sample data. Each row corresponds to a particular cluster, showing similar shape, background, and appearance. . . . .	26
3.3	Evaluation of the proposed algorithm on In-House (1 <sup>st</sup> row ) and Caltech (2 <sup>nd</sup> row) datasets. All results are plotted as miss-rate w.r.t. false-alarm-rate in FPPW. The first column details performances of local linear SVM learning with $k$ -means clustering, where $k$ varies from 1 to 2000; The second column shows performance achieved by proposed clustering methods, compared with the best results in the first column; The final column compares performances achieved by 2 different local learning methods. . . .	33

3.4	Comparison between proposed algorithm and state of the art on Caltech <i>training</i> dataset, adopting the same experimental setting and evaluation methodology as [1], plotted as miss-rate w.r.t. FPPI. The precision values at FPPI=1 are shown for easy comparison. . . . .	36
3.5	Evaluation of local SVM learning after k-means clustering on INRIA dataset. When $k = 1$ or 2 the performance increases slightly. However, further increasing $k$ deteriorates the performance quickly. . . . .	38
3.6	Comparison of detection results on Caltech dataset, between one-stage global classifier without local learning, local learning by linear SVM, and local learning by Min-Max adaptation. The last approach achieves both the best detection rate and the lowest false alarm rate. . . . .	39



## ABSTRACT

For object detection, the state-of-the-art performance is achieved through supervised learning. The performances of object detectors of this kind are mainly determined by two factors: features and underlying classification algorithms. In this work, we aim at improving the performance of object detectors from the aspect of classification algorithm. Observing the fact that classifiers used for object detection are task dependent and data driven, we developed a hybrid learning algorithm combining global classification and local adaptations, which automatically adjusts model complexity according to data distribution. We divide data samples into two groups, easy samples and ambiguous samples, using a learned global classifier. A local adaptation approach based on spectral clustering and proposed Min-Max model adaptation is then applied to further process the ambiguous samples. The proposed algorithm automatically determines model complexity of the local learning algorithm according to the distribution of ambiguous samples. By autonomously striking a balance between model complexity and learning capacity, the proposed hybrid learning algorithm incarnates a human detector outperforming the state-of-the-art algorithms on a couple of benchmark datasets [2, 1] and a self-collected pedestrian dataset. Besides, the proposed Min-Max model adaptation algorithm also successfully improve the performance of an offline-trained classifier on-site by adapting the classifier towards newly acquired data, without worries about the tuning the adaptation rate parameter, which affects the performance gain substantially. Taking the object detection as a testbed, we implement an adapted object detector based on binary classification. Under different adaptation scenarios and different datasets including PASCAL, ImageNet, INRIA, and TUD-Pedestrian, the proposed adaption method achieves significant performance gain and is compared favorably with the state-of-the-art adaptation method with the fine tuned adaptation rate.

# Chapter 1

## Introduction

Among various approaches to object detection, the sliding window approach [3, 2] dominates due to its good performance [4, 5, 6, 7, 8, 9], efficiency [3, 10], parallelizability, and easy implementation. The sliding-window-based detectors treat the object detection as a classification problem: The whole image is densely scanned from the top left to the bottom right with rectangular scanning windows of different sizes. For each possible scanned rectangle, certain features such as edge histogram, texture histogram, or wavelet coefficients, are extracted and fed to a offline trained classifier using labeled training data. The classifier is trained to classify any rectangle bounding an object of interest as a positive sample and to classify all other rectangles as negative samples.

The performances of sliding-window-based detectors are mainly determined by two factors: the feature and the underlying classification algorithm. In this work, we aim at improving the performance of object detectors from the aspect of the underlying classification algorithm. Many supervised learning algorithms such as various boosting algorithms [3, 11, 12], SVM of different flavors including linear, kernel, multi-kernel, latent, structured, etc. [2, 7, 9, 4, 8], and Convolutional Neural Networks (CNN) [13], have been applied to object detection during the past decade.

However the performance of supervised<sub>1</sub> learning algorithms heavily depends on the

labeled training dataset. For a specific classification task, a generic classifier trained with the data collected from various environments may only achieve fair performance because it has to accommodate the extensive dataset. Whereas the classifier trained using only the data sampled from the testing environment tends to overfit the training data and perform poorly if the variation of the testing dataset is big. This is the trade-off we have to balance in practical applications; we can adapt a generic classifier to a specific task, but we have to tune the adaptation rate according to different application scenarios.

Although the performance of an offline-trained classifier can be improved on-site by adapting the classifier, the performance gain is substantially affected by the adaptation rate. Poor selection of the adaptation rate may worsen the performance of the original classifier. Therefore, automatic model adaptation for classification tasks is an important problem with great application value.

Various model adaptation approaches [14, 15, 16, 17, 18, 19, 20] have been proposed under the scenarios of online learning for detection/tracking tasks. Zhang *et.al.* [21] elegantly formulate the model adaption problem as an optimization problem on the combined cost function of the old dataset and the new dataset. The cost function of the old dataset is approximated with its second order Taylor expansion to alleviate the storage burden and computational complexity. However, in their work, the adaption rate is an empirical parameter affects the adaptation performance significantly and demands careful fine tuning.

To avoid tuning the important adaptation rate parameter, we propose a conservative model adaptation method by considering the worst case during the adaptation process. We first construct a random cover of the set of the adaptation data from its partition. For each element in the cover (*i.e.* a portion of the whole adaptation data set), we define the cross-entropy error function in the form of logistic regression. The element in the cover with the maximum cross-entropy error corresponds to the worst case in the adaptation. Therefore we can convert the conservative model adaptation into the classic min-max optimization problem: finding the adaptation parameters that minimize the maximum of the cross-entropy

errors of the cover. Taking the object detection as a testbed, we implement an adapted object detector based on binary classification. Under different adaptation scenarios and different datasets including PASCAL, ImageNet, INRIA, and TUD-Pedestrian, the proposed adaption method achieves significant performance gain and is compared favorably with the state-of-the-art adaptation method [21] equipped with the fine tuned adaptation rate. Without the need of tuning the adaptation rates, the proposed conservative model adaptation method can be extended to other adaptive classification tasks.

With the help of proposed Min-Max model adaptation algorithm, we move on to address another learning problem: striking a good balance between model complexity and learning capacity. To ensure the detector has enough learning capacity to learn from training data and can be generalized well, people frequently resort to the Occam's razor principle [22, 23] to select underlying classifiers: we want to pick up a classifier, as simple as possible, with good performance on training data. With a spectrum of classifiers with different model complexity, is it possible to automatically pick up a classifier with appropriate complexity and to learn the corresponding model parameters? When the distribution of data in the input space is uneven, local learning algorithms can adjust the learning capacity locally to improve the overall performance. Zhang *et.al.* [24] proposed SVM-KNN that successfully tackles the problem of high variance of the data complexity in the input space, at the expense of high computational complexity. Similar idea was introduced in [25]. These local learning algorithms are superior in adjusting the learning capacity according to the local data distribution.

However, the local learning algorithms have three difficulties in real world applications: **First**, probing the local data distribution is very expensive. For example, both of the local learning algorithms [25, 24] rely on the K-Nearest Neighbor (KNN) algorithm to guide the local classifier. This probing procedure limits the application of the local learning algorithms in large scale learning practice such as object detection. **Second**, the localities depend on data distributions: a region with a simple distribution should be covered with a

relatively small number of local classifiers whereas a region with a complicated distribution should be covered with a large number of local classifiers. The “ $K$ ” in KNN algorithm is a constant and cannot fulfill such an adaptive task. **Third**, the performance of a local classifier relies on the population of the cluster. Complicated distributions may lead to low-population clusters, making the corresponding local classifier under trained.

To tackle the difficulties above, we developed a hybrid learning algorithm combining the global classification and the local adaptations, which automatically adjusts the model complexity according to the data distribution. As sketched in Figure 3.1, we divide the data samples into two groups, easy samples and ambiguous samples, using a learned global classifier. A local adaptation approach based on spectral clustering and Min-Max model adaptation is then applied to further process the ambiguous samples. The idea behind the proposed hybrid algorithm is straightforward: 1) Easy regions do not need local learning; 2) The local classifier can leverage on global classifier to avoid under-training; 3) Data in hard regions can be automatically clustered based on their affinity matrix using accelerated spectral clustering. Taking human detection as a testbed, under different scenarios and datasets, including Caltech pedestrian dataset [1], self-collected large pedestrian dataset, and INRIA dataset [2], the proposed hybrid learning method achieves significant performance gain. Compared with 11 state-of-the-art algorithms [1] on Caltech, the proposed approaches achieves the highest detection rate, outperforming the deformable part based algorithm [4] 17% at FPPI=1. Additionally, without the need of tuning parameters, the proposed algorithm automatically generates the optimum group (verified by brute force enumeration) of local classifiers for different scenarios, which makes the algorithm easily be extended to different object detection tasks.

Our contributions are in four-folds: 1) we propose a model adaptation algorithm that Without the need of tuning the adaptation rates, the algorithm achieves significant performance gain and is compared favorably with the state-of-the-art adaptation method with the fine tuned adaptation rate. 2) we develop a hybrid learning algorithm that enables the

application of local learning algorithm in large-scale tasks; 3) the proposed hybrid learning method can automatically adjust the model complexity according to the distribution of the training data; 4) the proposed scheme of local adaptation from global classifier avoids the common under-training problem for local classifier: we gain significant performance enhancement in human detection over traditional algorithms, with very little increment in computational cost.

# Chapter 2

## Min-Max Model Adaptation

### 2.1 Introduction

The performance of supervised learning algorithms including various flavors of SVM [22, 20], boosting [26, 27, 28, 3], neural networks [29, 13], heavily depends on the labeled training dataset. For a specific classification task, a generic classifier trained with the data collected from various environments may only achieve fair performance because it has to accommodate the extensive dataset. Whereas the classifier trained using only the data sampled from the testing environment tends to overfit the training data and perform poorly if the variation of the testing dataset is big. This is the trade-off we have to balance in practical applications; we can adapt a generic classifier to a specific task, but we have to tune the adaptation rate according to different application scenarios.

Although the performance of an offline-trained classifier can be improved on-site by adapting the classifier, the performance gain is substantially affected by the adaptation rate. Poor selection of the adaptation rate may worsen the performance of the original classifier. Therefore, automatic model adaptation for classification tasks is an important problem with great application value.

To solve this problem, we propose a conservative model adaptation method by considering the worst case during the adaptation process. We first construct a random cover of the set of the adaptation data from its partition. For each element in the cover (*i.e.* a portion of the whole adaptation data set), we define the cross-entropy error function in the form of logistic regression. The element in the cover with the maximum cross-entropy error corresponds to the worst case in the adaptation. Therefore we can convert the conservative model adaptation into the classic min-max optimization problem: finding the adaptation parameters that minimize the maximum of the cross-entropy errors of the cover. Taking the object detection as a testbed, we implement an adapted object detector based on binary classification. Under different adaptation scenarios and different datasets including PASCAL, ImageNet, INRIA, and TUD-Pedestrian, the proposed adaptation method achieves significant performance gain and is compared favorably with the state-of-the-art adaptation method with the fine tuned adaptation rate. Without the need of tuning the adaptation rates, the proposed conservative model adaptation method can be extended to other adaptive classification tasks.

The remainder of the paper is organized as follows. Sec. 2.2 formulates the adaptation cost as the cross-entropy error function. Converting the adaptation in the worst case into a classical min-max optimization problem is discussed in details in sec. 2.3. Experimental results are presented and discussed in Sec. 2.4. Finally we draw the conclusion in Sec. 2.5.

## 2.2 Cross-Entropy Error as the Adaptation Cost

To train a classifier, positive and negative examples are given as a training dataset  $\mathcal{S} = \{(\mathbf{x}_k, y_k), k = 1, \dots, K, (\mathbf{x}_k, y_k) \in \mathcal{X} \times \mathcal{Y}\}$ , in which  $y_k$  is the label of the example  $x_k$ . If  $\mathbf{x}_k$  is a positive example,  $y_k = 1$  and  $y_k = 0$  if it is a negative example. A parametric learning algorithm is then applied on  $\mathcal{S}$ , to find the decision function:



$$\mathcal{Z} = F(\mathcal{X}|\mathbf{w}), \quad (2.1)$$

where  $\mathbf{w}$  is the parameter vector of the trained classifier. In Eq.(2.1), each  $z \in \mathcal{Z}$  is a mapped label of the corresponding example  $\mathbf{x} \in \mathcal{X}$ . Usually,  $\mathbf{w}$  should be optimized according to a cost function defined to measure the classifier performance over the training dataset:

$$C(\mathcal{Z}, \mathcal{S}) = C(F(\mathcal{X}|\mathbf{w}), \mathcal{S}). \quad (2.2)$$

During the model adaptation, the parameter vector of the classifier trained on *old* dataset  $\mathcal{S}^{(o)}$  is used as the initial parameter vector, denoted as  $\mathbf{w}^{(o)}$ . With a labeled dataset  $\mathcal{S}^{(n)}$  (also called as adaptation dataset) collected from a new environment, we want to obtain an adapted classifier with the updated parameter vector  $\mathbf{w}^{(n)}$ , which performs better in the new environment. That is to find the parameter vector minimizing the cost function on the new dataset  $\mathcal{S}^{(n)}$ :

$$\mathbf{w}^{(n)} = \arg \min_{\mathbf{w}} C^{(n)}(\mathbf{w}) \quad (2.3)$$

There exist many formulations to define the cost function, among which we choose *logistic regression* [30] for its good performance, applicability, and popularity. Specifically, for a data set  $\mathcal{S} = \{(\mathbf{x}_k, y_k), k = 1, \dots, K\}$ , the likelihood of an example  $x_k$  being a positive example is:

$$p_k = \frac{1}{1 + \exp\{-Score(\mathbf{x}_k, \mathbf{w})\}}, \quad (2.4)$$

where the  $Score(\mathbf{x}_k, \mathbf{w})$  is the confidence score of the example  $\mathbf{x}_k$  computed by the classifier with the parameter vector  $\mathbf{w}$ . Therefore, the likelihood function of the whole data set

can be written as:

$$P = \prod_{k=1}^K p_k^{y_k} (1 - p_k)^{1-y_k}. \quad (2.5)$$

We define the cost function by taking the negative logarithm of the likelihood; this definition leads to the *cross-entropy* error function:

$$C = -\frac{1}{K} \ln P = -\frac{1}{K} \sum_{k=1}^K \{y_k \ln p_k + (1 - y_k) \ln(1 - p_k)\}. \quad (2.6)$$

The entries of the gradient vector  $\nabla C(\mathbf{w})$  and the Hessian matrix  $\mathbf{H}_C(\mathbf{w})$  of the cost function can be computed as:

$$\frac{\partial C}{\partial w_j} = \frac{1}{K} \sum_{k=1}^K \mathbf{x}_k(j)(p_k - t_k), \quad (2.7)$$

and

$$\frac{\partial^2 C}{\partial w_i \partial w_j} = \frac{1}{K} \sum_{k=1}^K p_k(1 - p_k) \mathbf{x}_k(i) \mathbf{x}_k(j), \quad (2.8)$$

where  $\mathbf{x}_k(i)$  is the  $i$ th entry of the feature vector  $\mathbf{x}_k$ .

## 2.3 Model Adaptation as Min-Max Optimization

The model adaptation problem aims at adapting a previously trained detector, to fit new data with a distribution different from the one of the data used to train the original detector. The distribution of the new data determines the adaptation rate. If the variation within the new data is small, *i.e.* the new data are relatively similar to each other, we should set a high adaptation rate. On the other hand, if the variation within the new data is big, that means fast adaptation to the new data is risky since we cannot estimate the new data distribution robustly; we have to set a low adaptation rate to avoid overfitting. But how do we estimate the variation of new data? We can randomly split the whole new data into many overlapped subsets and watch the differences among these subsets: big differences

indicate large variation and vice versa.

### 2.3.1 Constructing a random cover from the adaptation dataset

To estimate the data variation of the adaptation set  $\mathcal{S}^{(n)}$ , we randomly divide it into  $M$  small partitions:

$$\mathcal{S}^{(n)} = \bigcup_{i=1}^M \mathcal{S}^{(n)_i}, \quad (2.9)$$

constrained by the non-intersection condition:

$$\mathcal{S}^{(n)_i} \cap \mathcal{S}^{(n)_j} = \emptyset, \quad \text{for } i \neq j, \quad (2.10)$$

where  $i, j = 1, \dots, M$ .

With this random partition, we construct an  $N$ -element random cover of  $\mathcal{S}^{(n)}$ , denoted as  $\{G_1, G_2, \dots, G_N\}$ , as illustrated in Figure 2.1:

$$G_j = \bigcup_{i \in E_j} \mathcal{S}^{(n)_i}, \quad (2.11)$$

for  $j = 1, \dots, N-1$ , and

$$G_N = \mathcal{S}^{(n)} - \bigcup_{j=1}^{N-1} G_j, \quad (2.12)$$

where  $E_j \subset \{1, \dots, M\}$  is a random subset.

This random cover construction avoids the complete enumeration of the power set of the partition (*i.e.*, the uniform sampling), which leads to huge computation.

### 2.3.2 Min-Max objective function

Since direct evaluate the data distribution on each element (*i.e.*, each  $G_j$ ) of the random cover is very diff cult, we resort to evaluate the cost function on the  $G_j$ . We denote the cost

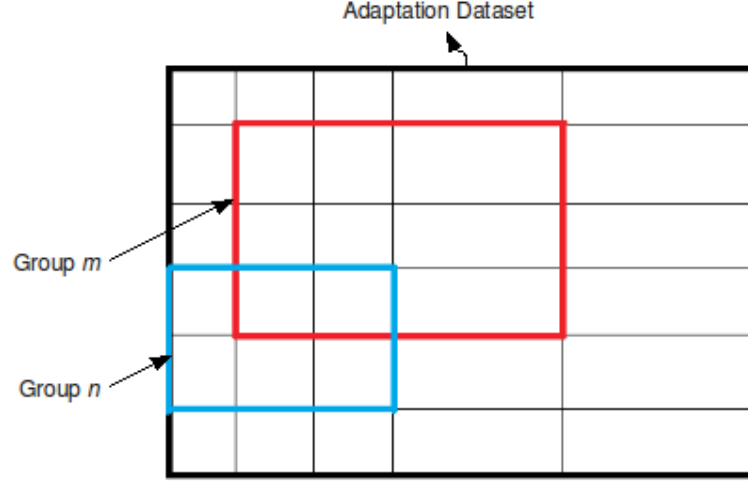


Figure 2.1: The construction of the random cover of the adaptation dataset. Each small block represents  $S^{(n)}_i$ , an element of the partition. The  $G_j$ 's could have different size and overlap with each other.

functions on the old dataset and the adaptation dataset as  $C_{(o)}(\mathbf{w})$  and  $C_{(n)}(\mathbf{w})$  respectively. For each  $G_j$ , the corresponding cost function is denoted as  $C_j(\mathbf{w})$ . Their logistic regression formulation is given in Eq. (2.6).

Different from previous methods which update parameter vector  $\mathbf{w}$  through minimizing the  $C_{(n)}(\mathbf{w})$ , we propose a *conservative adaptation approach*, to guarantee that the adapted detector performs relatively well even when the adaptation dataset has scattered data distribution. During the adaptation, we focus on updating the parameter vector  $w$  to improve the detector's performance on the  $G_j$  with the largest cost function, *i.e.*, the biggest cross-entropy error. Therefore, we formulate the conservative adaptation into a *min-max* problem:

$$\mathbf{w} = \arg \min_{\mathbf{w}} \left( \max_j [C_j(\mathbf{w})] \right). \quad (2.13)$$

To solve this min-max problem, at each adaptation iteration, we define the cost function on the adaptation dataset as

$$E(\mathbf{w}) = \max_j [C_j(\mathbf{w})]. \quad (2.14)$$

### 2.3.3 Min-Max optimization

Noticing the positivity of  $C_j(\mathbf{w})$ 's in Eq.(2.13), we can compute  $\max[C_j(\mathbf{w})]$  as the infinity norm of the cost function vector,  $\mathbf{C}(\mathbf{w}) = (C_1(\mathbf{w}) \ C_2(\mathbf{w}) \ \dots \ C_N(\mathbf{w}))^T$ . Therefore, we have

$$\max_j [C_j(\mathbf{w})] = \|\mathbf{C}(\mathbf{w})\|_\infty = \lim_{q \rightarrow +\infty} \left( \sum_{j=1}^N [C_j(\mathbf{w})]^q \right)^{\frac{1}{q}}. \quad (2.15)$$

Therefore Eq. (2.13) can be approximated with a large  $q$ :

$$\mathbf{w} = \arg \min_{\mathbf{w}} (E(\mathbf{w})) \simeq \arg \min_{\mathbf{w}} \left( \sum_{j=1}^N [C_j(\mathbf{w})]^q \right). \quad (2.16)$$

To update the parameter vector  $\mathbf{w}$ , we use *Newton-Raphson* [30] method to compute the minimizer of the cost function above:

$$\mathbf{w}^{[i+1]} = \mathbf{w}^{[i]} - \mathbf{H}_E^{-1}(\mathbf{w}^{[i]}) \cdot \nabla E(\mathbf{w}^{[i]}), \quad (2.17)$$

where  $\nabla E(\mathbf{w})$  and  $\mathbf{H}_E(\mathbf{w})$  are the gradient and the Hessian matrix of the cost function  $E(\mathbf{w})$ . They can be computed as:

$$\nabla E(\mathbf{w}^{[i]}) = q \sum_{j=1}^N \left( [C_j(\mathbf{w}^{[i]})]^{q-1} \cdot \nabla C_j(\mathbf{w}^{[i]}) \right) \quad (2.18)$$

$$\begin{aligned} \mathbf{H}_E(\mathbf{w}^{[i]}) = & q \sum_{j=1}^N \left( [C_j(\mathbf{w}^{[i]})]^{q-1} \cdot \mathbf{H}_{C_j}(\mathbf{w}^{[i]}) \right. \\ & \left. + (q-1) [C_j(\mathbf{w}^{[i]})]^{q-2} \cdot [\nabla C_j(\mathbf{w}^{[i]})]^2 \right). \end{aligned} \quad (2.19)$$

We use the parameter vector  $\mathbf{w}^{(o)}$  of the classifier trained on the old dataset to initialize the iterative optimization process:  $\mathbf{w}^{[0]} = \mathbf{w}^{(o)}$ . The iterative optimizing process terminates

at a certain threshold  $\xi$ :

$$\sqrt{\nabla E(\mathbf{w}^{[i]})^T \cdot \mathbf{H}_E^{-1}(\mathbf{w}^{[i]}) \cdot \nabla E(\mathbf{w}^{[i]})} < \xi. \quad (2.20)$$

Our algorithm is summarized in Algorithm(1).

---

**Algorithm 1:** The Min-Max Based Adaptation

---

**Input:** Old detector with parameter vector  $\mathbf{w}^{(o)}$ , adaptation dataset  $\mathcal{S}^{(n)}$ , and threshold  $\xi$ .

**Output:**  $\mathbf{w}^{(n)}$  for adapted detector.

Based on Eq.(2.10), divide  $\mathcal{S}^{(n)}$  into  $M$  small subsets:

$\mathcal{S}^{(n)} = \mathcal{S}^{(n)_1} \cup \mathcal{S}^{(n)_2} \cup \dots \cup \mathcal{S}^{(n)_M}$ ;

form  $N$  data covers  $G_1, G_2, \dots, G_N$  refer to Eq.(2.11, 2.12), and get the cost functions  $C_1(\mathbf{w}), C_2(\mathbf{w}), \dots, C_N(\mathbf{w})$ ;

define cost function  $E(\mathbf{w})$  on  $\mathcal{S}^{(n)}$  using Eq.(2.13);

approximate  $E(\mathbf{w})$  as shown in Eq.(2.15);

set  $\mathbf{w}^{[0]} = \mathbf{w}^{(o)}, T = \infty, i = 0$ ;

**while**  $T \geq \xi$  **do**

    compute  $\mathbf{H}_E(\mathbf{w}^{[i]})$ ,  $\nabla E(\mathbf{w}^{[i]})$  as Eq.(2.18, 2.19);

    compute  $\mathbf{w}^{[i+1]}$  refer to Eq.(2.17);

$T = \sqrt{\nabla E(\mathbf{w}^{[i]})^T \cdot \mathbf{H}_E^{-1}(\mathbf{w}^{[i]}) \cdot \nabla E(\mathbf{w}^{[i]})}$ ;

$i = i + 1$ ;

$\mathbf{w}^{(n)} = \mathbf{w}^{[i]}$ ;

**return**  $\mathbf{w}^{(n)}$ ;

---

## 2.4 Experiments

We now present experiments to evaluate the proposed model adaptation algorithm. The model adaptation algorithm is test for two group of adaptation scenario: the first three experiments are designed to adapt an original human detector trained on the same old dataset to three different adaptation datasets; the last experiment emphasizes on the adapting a general object detector. The good performance of the proposed adaptation algorithm is validated by these experiments.



Figure 2.2: From left to right, the first box gives the examples we use to train our generic human detector, all coming from INRIA dataset; the second to the fourth boxes give positive examples sampled from three adaptation dataset: the INRIA dataset, the TUD dataset, and the PASCAL dataset.

The rest of this section consists of seven subsections: the first subsection describes the features we use, and details the parameters used; the second subsection presents the comparison of the adaptation methods; the third subsection introduces the generic human detector we use and evaluates our method on part of the INRIA dataset [2]; the fourth subsection tests our algorithm on the TUD-pedestrian dataset [31]; the fifth subsection adopts our method on adapting a “motorcyclist” detector, with data sampled from the PASCAL dataset [32], and with the analysis on the effects of the related parameters; the sixth subsection focuses on regular object detector: adapting a Parrot detector from a generic bird detector, in which training and test images come from ImageNet dataset [33]; the last subsection gives the computational complexity and a short experimental discussion.

### 2.4.1 Image representation and experimental setup

**Shape and Texture descriptors.** In all of the following experiments, we use integrated HOG-LBP features [34]. HOG has been widely accepted as one of the best features to capture the edge or local shape information, whereas LBP is an exceptional texture descriptor.

**Parameters setting.** For all of the positive and negative examples, we fix their sizes as  $64 \times 128$  pixels. Thus, the HOG-LBP feature is a 5668 dimensional vector, which makes the Hessian matrix in Eq.(2.17, 2.19) a  $5668 \times 5668$  matrix. Considering the huge computational cost in computing the Hessian matrix, we use a small positive number  $\alpha =$

$0.1 \cdot \mathbf{I}$  to replace the Hessian matrix in Eq.(2.20). To construct the random cover, we firstly split the adaptation set into 10 small subsets, and then form 5 data covers. In the third experiment, we analyze the effect of data cover number  $N$  using PASCAL dataset. As mentioned before, the  $q$  in Eq.(2.15) affects the approximation of our proposed cost function. Theoretically, bigger  $q$  leads to better approximation. In our experiments, we assign 3 to  $q$  and achieve quite good performance. The iteration threshold  $\xi$  is set 0.04.

## 2.4.2 Performance comparison between different adaptation methods

**Taylor expansion adaptation with optimizing adaptation rate.** Taylor expansion adaptation method proposed in [21] gives us a good way to adapt a generic classifier in a new environment. The method uses Taylor expansion of the cost function on the old data as an approximation, and then combines it with the cost function on the adaptation dataset:

$$\begin{aligned}
J(\mathbf{w}) \approx & C_{(o)}(\mathbf{w}^{(o)}) + \nabla C_{(o)}(\mathbf{w}^{(o)}) \cdot (\mathbf{w} - \mathbf{w}^{(o)}) + \\
& \frac{1}{2}(\mathbf{w} - \mathbf{w}^{(o)})^T \cdot \mathbf{H}_{C_{(o)}}(\mathbf{w}^{(o)}) \cdot (\mathbf{w} - \mathbf{w}^{(o)}) \\
& \lambda \cdot C_{(n)}(\mathbf{w})
\end{aligned} \tag{2.21}$$

where  $J(\mathbf{w})$  is the overall cost function, and  $\lambda$  is the parameter controlling the adaptation rate.

The adaptation rate  $\lambda$  is a very important parameter and requires careful tuning in [21]. The poor setting of  $\lambda$  will make the adapted detector perform worse than the original detector. So we improve this algorithm using *cross-validation* to find the optimal  $\lambda$ . Shown in Algorithm.(2), the improved version of [21] is used as a baseline to compare with our conservative adaptation algorithm based on min-max.



---

**Algorithm 2:** Taylor Expansion Adaptation With Optimizing Adaptation Rate

---

**Input:** Old detector with parameter vector  $\mathbf{w}^{(o)}$ , and adaptation dataset  $\mathcal{S}^{(n)}$

**Output:**  $\mathbf{w}^{(n)}$  for adapted detector

divide  $\mathcal{S}^{(n)}$  into K same size partitions  $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_K$ ;

$i = 1$ ;

**while**  $i \leq K$  **do**

    take  $\mathcal{S}_i$  to validate, the rest parts to adapt;

**for**  $\lambda = 0 \rightarrow \infty$  **do**

        use Eq.(2.21) to adapt. validate on  $\mathcal{S}_i$ , find the optimal  $\lambda_i$

$i = i + 1$ ;

$\bar{\lambda} = \frac{1}{K} \sum_{i=1}^K \lambda_i$ ;

set  $\lambda = \bar{\lambda}$  in Eq.(2.21), do adaptation and get  $\mathbf{w}^{(n)}$ ;

return  $\mathbf{w}^{(n)}$ ;

---

### 2.4.3 INRIA dataset

The INRIA person dataset [2] now has become a standard to evaluate human detector on, which provides 2416 positive samples and 1218 negative images for training, 1132 positive samples and 453 negative images for testing. All the positive samples are set into  $64 \times 128$  pixels.

Looking at INRIA dataset, we find there exists one interesting point that the positive samples from training and test set can be divided into two parts: pedestrian images and cyclist images. So they could be separately used to train a generic pedestrian detector and to adapt a cyclist detector. We pick up 246 cyclist positive examples from the whole dataset, and take all the 453 negative images from test set, to build up the cyclist dataset. Then using 116 positive examples and 79 negative images in it for adaptation, the rest positive and negative for evaluation. In the INRIA dataset, the rest 3302 positive pedestrian samples together with 1218 negative images from training set are used to train a generic pedestrian detector. This detector is treated as the old classifier in experiment 1-3.

Fig. 2.5(a) shows that the old detector already works very well in the cyclist dataset. When false alarm rate is 0.02, detection rate is already 99.3%. In certain point, it proves our generic pedestrian detector performs well on norm person detection task, suitable to be used

as old detector in experiment 1-3. Also, we could see our proposed adaptation algorithm could further improve the detector rate, even when the old one already performs well. However, experiment 1 is not quite able to show the good performance of our algorithm, when comparing with the rest experiments.

#### **2.4.4 TUD-pedestrian dataset**

TUD-pedestrian dataset [31] is a new benchmark for pedestrian detection. In this dataset, images are sampled from a long video sequence, which is collected by an on-board camera from a driving car in urban environment. So this dataset is a realistic and challenging pedestrian dataset for human detector. It contains 3552 positive samples with mirroring and 192 negative image pairs for training and 508 test images with 1269 evaluation positive examples.

As we mentioned before, the INRIA pedestrian images train an old detector. From TUD dataset, we use 1000 positive samples with mirroring from training dataset and 1620 negative examples extracted from 81 frames(20 negative examples per image), to adapt the old detector. Then we test our adaptation algorithm on the whole TUD testing dataset(1269 positive examples, 10160 negative examples from 508 images). From Fig. 2.5(b), it is easy to observe that the detector adapted by our algorithm outperforms the old detector by 15% – 20% detection rate in FPPW, and 7% – 10% higher comparing with the Taylor expansion algorithm result with optimal parameter  $\lambda$ .

#### **2.4.5 PASCAL dataset**

PASCAL VOC [32] is probably one of the most difficult and widely used reference datasets in computer vision. The PASCAL VOC 2007 is the latest version having labels for both training and test datasets. For the human detection specific category, there are 2000+ images for training and 2000+ images for testing. Using our generic human detector on these,

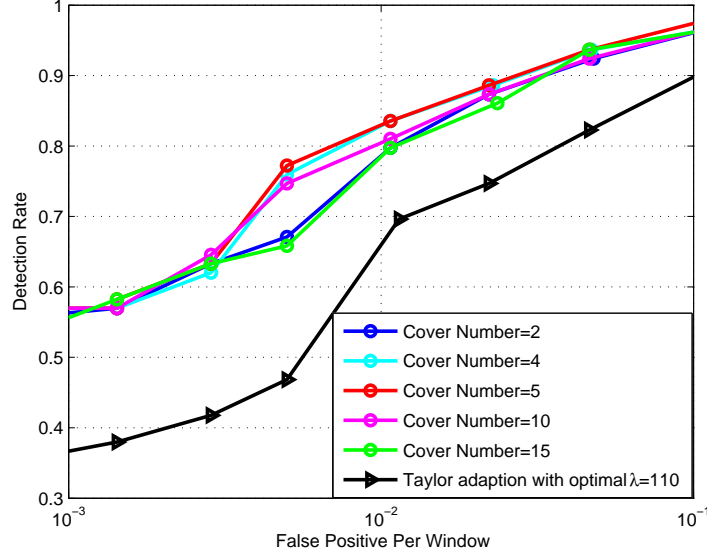


Figure 2.3: Evaluation of the effect of data cover number  $N$  in our algorithm, experiment is on PASCAL dataset. We randomly set  $N$ , and draw the curve when  $N = 2, 4, 5, 10, 15$ , all comparing with Taylor adaptation method with optimal  $\lambda$ .

we find the performance in “motorcyclist” images is bad. So we extract these images out, constructing a motorcyclist dataset, containing 600 positive examples, and 200 negative images randomly selected from images which not contain person in PASCAL dataset. From this small dataset, we take 92 positive samples, and 1440 negative samples extracted from 72 images (20 negative examples per image) as our adaptation dataset, the rest positive and negative are all considered as test dataset.

Fig. 2.5(c) illustrates how badly the old classifier works on this motorcyclist dataset. If using Taylor expansion method, we can improve it a lot, about 30% detection rate in FPPW. While using our adaptation algorithm, the performance can be further improved by 10% – 15%, and now the adapted detector turns out to be a good detector on this new dataset.

**Effect of data cover number  $N$**  In Fig. 2.5(c), there exists a *STD bar* (standard deviation bar) in the curve of our proposed adaptation algorithm. It represents the effect of different adaptation data cover number  $N$  to our algorithm. It is obvious that this effect is

small and tolerable. Tuning  $N$ , the performance of our adaptation algorithm is always good with slight fluctuation. This means, our algorithm is stable to parameters. So it doesn't have much troubles in tuning parameters. Fig. (2.3) presents more details.



Figure 2.4: First two rows show positive and negative examples to train our bird detector, all extracted from ImageNet dataset; The third row gives some positive examples we labeled for our parrot dataset

## 2.4.6 ImageNet dataset

ImageNet [33] is a huge image dataset designed according to the WordNet [35] hierarchy (recently only with the 8000+ nouns), in which each node of the hierarchy is described with hundreds and thousands of images.

This dataset follows tree-like structure, while images of each concept in it own good quality and provide some annotations. These characteristics make it able to train tons of object detectors. Considering the time and labor consumption, training a generic detector for node in low level and adapting this for node in high level (root node is treated as level 0) might be an optimal way to get detectors for different categories.

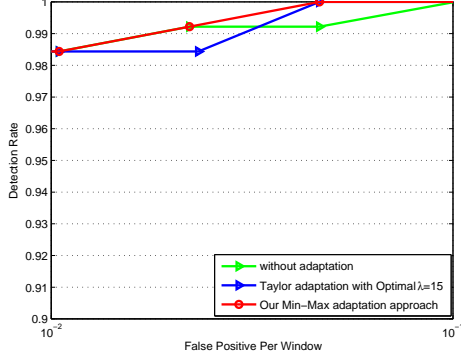
Our proposed adaptation algorithm provides an efficient, good and non-expensive method to complete this job. We demonstrate this in bird category. From the ImageNet dataset, we extract 2762 bird positive examples and 749 negative images from different other categories to train a generic bird detector, which is regarded as old detector in this experiment. Then we labeled a small *parrot* dataset, including 400 parrot positive examples, and 100 negative images selected from other categories. 20% of the parrot dataset are used for adaptation, the rest for test. In Fig. 2.5(d), we could see the adapted detector obtained by our algorithm outperforms the old bird detector by 10% – 20%, the Taylor’s adapted detector by 5% – 10% in FPPW. So it proves our algorithm could make a norm generic detector adapt into a good detector for certain specific category.

#### 2.4.7 Experiments discussion

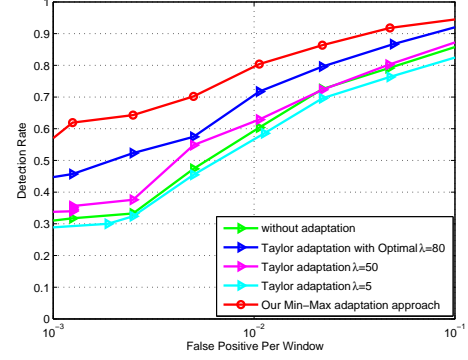
**Computational Complexity** On a Core 2 Duo 2.8 GHz computer, when we adapt with 2000 examples which generate 5 data covers, the average adaptation time of our proposed algorithm is 3 hours for 10000 adaptation iterations, which usually returns near-optimal adapted detector.

These experiments indicate, firstly the adapted detector generated by our algorithm works well in different detector adaptation tasks and outperforms the old detector and the Taylor expansion adaptation detector. Even when the old detector already performs very well in certain environment, as shown in experiment 1, our adapted detector also further improves the performance. Secondly, Comparing with the Taylor expansion adaptation method, our adaptation method has much less “tuning parameter” troubles, which means it’s easy and efficient to apply for detector adaptation tasks. Finally, from the PASCAL experiment, we find the adaptation data cover number  $N$  during division of adaptation dataset affects less to our algorithm’s performance. In Figure(2.6), we sample some detection results from our experiments. Comparing with old detector and Taylor’s adapted detector, the results from our adapted detector are mostly better. The positive and negative examples

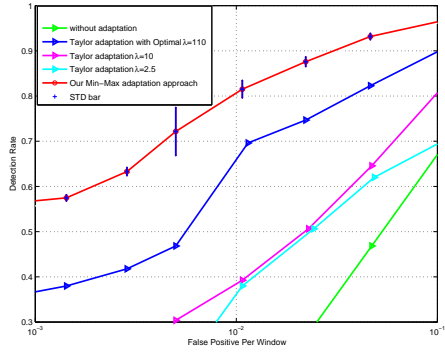
are separated further away.



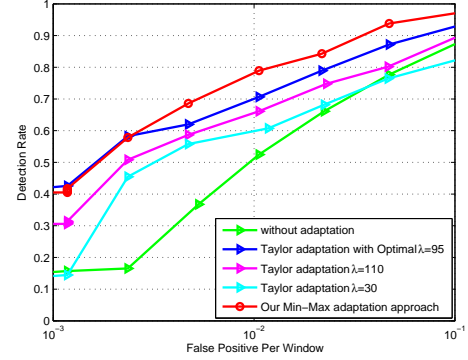
(a) INRIA dataset



(b) TUD dataset



(c) PASCAL dataset



(d) ImageNet dataset

Figure 2.5: Evaluation of our method on multiple datasets, compared with other adaptation method. In the curve of PASCAL dataset, the standard deviation bar represents the effect of data cover number  $N$  on the performance of our algorithm.

## 2.5 Conclusions

We have proposed an object detector adaptation algorithm which has little worry about tuning many parameters. This idea is to generate several adaptation data covers from the adaptation dataset according to certain rules, and during each adaptation iteration, always consider the worse data cover. The adaptation problem then changes into a Min-Max problem which could be solved by approximation of infinite norm method. This algorithm

has the ability to train performance-guaranteed detectors for different categories in various environments. This characteristic of our algorithm is also demonstrated in experiments.




0.426	0.440	0.028	-4.280	-3.307	-4.033
0.434	0.437	-0.201	-5.264	-3.917	-3.925
1.060	0.927	0.301	-9.897	-5.321	-8.376

(a) INRIA dataset




-2.351	-0.422	0.414	-2.561	-1.338	-2.686
1.767	2.057	2.119	-1.510	-0.992	-3.992
6.646	3.682	6.047	-2.695	-5.29	-10.495

(b) TUD dataset



-2.718	-1.728	-2.101	-3.392	-3.505	-2.719
-0.714	-0.172	1.649	-2.908	-2.837	-1.890
0.142	0.975	3.018	-4.121	-3.082	-1.712

(c) PASCAL dataset



1.031	1.520	1.222	-2.191	-1.523	-0.808
4.362	4.517	1.432	-2.643	-1.743	-1.380
7.505	7.434	2.673	-2.717	-2.175	-3.566

(d) ImageNet dataset

Figure 2.6: Performance comparison between detectors, all the number are detection scores. The first row is from old detector; the second row is from adapted detector generated by Taylor expansion adaptation method with the optimal adaptation rate; the third row is from detector adapted by our proposed algorithm. In each dataset, we samples 3 positive and 3 negative examples. The score difference between positive and negative examples is larger, the performance is better.

## Chapter 3

# Divide with Global Classifier, Conquer with Local Adaptation

### 3.1 Introduction

Among various approaches to object detection, the sliding window approach [3, 2] dominates due to its good performance [4, 5, 6, 7, 8, 9], efficiency [3, 10], parallelizability, and easy implementation. The sliding-window-based detectors treat the object detection as a classification problem: The whole image is densely scanned from the top left to the bottom right with rectangular scanning windows of different sizes. For each possible scanned rectangle, certain features such as edge histogram, texture histogram, or wavelet coefficients, are extracted and fed to an offline trained classifier using labeled training data. The classifier is trained to classify any rectangle bounding an object of interest as a positive sample and to classify all other rectangles as negative samples.

The performances of object detectors of this kind are mainly determined by two factors: features and underlying classification algorithms. In this work, we aim at improving the performance of object detectors from the aspect of classification algorithm. Observing the fact that classifiers used for object detection are task dependent and data driven, we de-



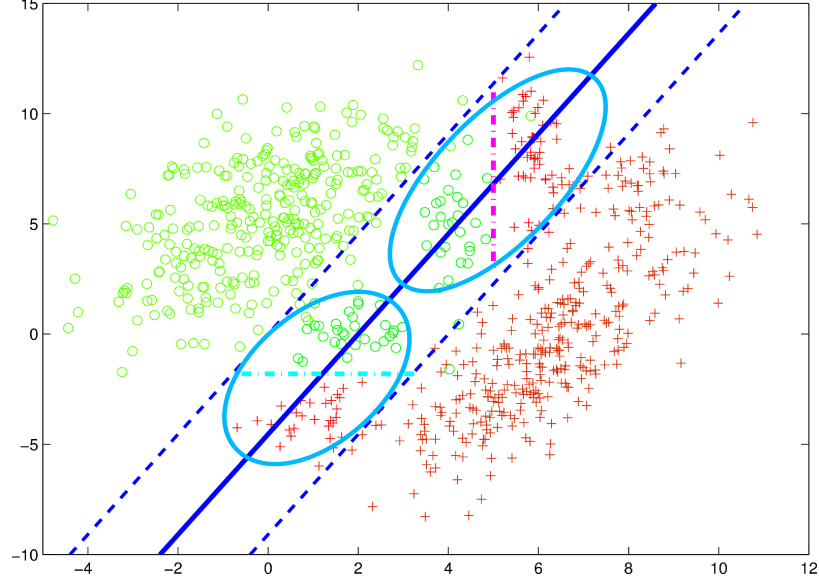


Figure 3.1: A Toy Example of Two-class (“o” vs “x”) Classification Using Our Approach: A global classifier (blue solid line) and its boundaries (blue dotted lines) divide the data space into easy regions and hard regions. The ambiguous data in hard regions are clustered according to the data distribution, which automatically adjusts the model complexity. Each cluster of samples are classified using locally adapted classifier that avoids under training. The hybrid learning algorithm autonomously strikes a balance between model complexity and learning capacity.

veloped a hybrid learning algorithm combining global classification and local adaptations, which automatically adjusts model complexity according to data distribution. We divide data samples into two groups, easy samples and ambiguous samples, using a learned global classifier. A local adaptation approach based on spectral clustering and Min-Max model adaptation is then applied to further process the ambiguous samples. The proposed algorithm automatically determines model complexity of the local learning algorithm according to the distribution of ambiguous samples. By autonomously striking a balance between model complexity and learning capacity, the proposed hybrid learning algorithm incarnates a human detector outperforming the state-of-the-art algorithms on a couple of benchmark datasets [2, 1] and a self-collected pedestrian dataset. Compared with 11 state-of-the-art algorithms [1] on the Caltech dataset, the proposed approaches achieves the highest detection rate, outperforming the seminal and successful deformable model approach [4] by

17% at FPPI=1.

Our contributions are in three-folds: 1) we develop a hybrid learning algorithm that enables the application of local learning algorithm in large-scale tasks; 2) the proposed hybrid learning method can automatically adjust the model complexity according to the distribution of the training data; 3) the proposed scheme of local adaptation from global classifier avoids the common under-training problem for local classifier: we gain significant performance enhancement in human detection over traditional algorithms, with very little increment in computational cost.

The rest of the paper is organized as follows. Sec. 3.2 describes our global classification process for dividing the candidates into easy and ambiguous cases. Sec. 3.3 details our clustering method for balancing model complexity and learning capacity. Sec. 3.4 presents our local adaptation algorithm to further enhance learning capacity. Experimental results are shown in Sec. 3.5 followed by conclusion and future work in Sec. 3.6.

## 3.2 Divide by Global Classification

Our approach starts with a global classifier learned using all of the training data. The classified training data are then divided into two groups: easy samples and ambiguous samples. The ambiguous samples are further processed using a local adaptation algorithm.

The learned global classifier partitions the input space into easy regions and hard regions. Only the *ambiguous* data in hard regions will be passed into the next stage and handled by more discriminative local classifiers. Various general global learning algorithms [36, 37, 4, 38] are suitable to this task. Since one role of global classifier is a filter to select hard regions for local learning/adaptation, we require the global classifier to be efficient and highly generalizable with a relative satisfactory performance. Linear SVM meets our requirements. In order to locate the hard regions of *ambiguous* data, we set up the upper bound  $\Theta_1$  and lower bound  $\Theta_2$  based on the classification scores of the global



Figure 3.2: Sample results of our automatic clustering on ambiguous sample data. Each row corresponds to a particular cluster, showing similar shape, background, and appearance.

classifier. The data bounded inside are ambiguous data, requiring local learning.

### 3.3 Clustering - Adjusting Model Complexity

After filtering by the global classification, the remaining data (ambiguous data) are then processed using an automatic clustering algorithm. This provides an efficient and effective way to probing the local data distribution for each sample. The number of clusters and the population of each cluster are automatically adjusted. Together with a follow-up local adaptation, it strikes a balance between model complexity and learning capacity. Specifically we use a tailored spectral clustering algorithm to automatically divide the ambiguous data into local regions in feature space. This essentially adjusts the model complexity autonomously according to the data distribution.

### 3.3.1 Distance Metrics

In order to effectively cluster the ambiguous data, we first define the distance metric between a pair of samples in the feature space. Different distance metrics may lead to different clustering on the data. Here for popular shape descriptor and texture descriptor, we introduce several frequently used distance measures.

**Crude distance** Several simple and yet good measures are frequently used for computing distance in feature space, such as  $L_1$ -*sqr*t distance,  $L_\infty$ -*Norm* distance, and *Euclidean* distance. These crude distance measures have been widely adopted as meaningful solutions to distance computation.

**Accurate distance** Alternatively, more costly “accurate” distance measures were developed. [39] proposes *shape context distance* that matches the point sets of two shapes and scores how different the shapes are;  $\chi^2$  *distance* [40] maps the texture of each example to a histogram of “textons”, then defines distance as the Pearson’s  $\chi^2$  test statistic between the two histograms; Adapted from [40], *marginal distance* [41] sums up the distances between response histograms to measure the texture distance.

All these metrics may be used for clustering. In our experiments, crude distance already yields reasonable results with low computational complexity. Specifically we adopt the Euclidean-like distance. For each sample, the features are normalized according to their  $L_2$  norm, then the Euclidean distances with others are computed. The normalization is critical for finding and setting proper clustering parameters.

### 3.3.2 Clustering Algorithm

Many clustering algorithms can be adopted for clustering the ambiguous data. One straightforward method is *k-means* with a given number of clusters  $k$ . However inappropriate  $k$  may drastically deteriorate the performance of the system: if  $k$  is too small, certain local clusters would contain too many samples resulting in over-complicated models; On

the contrary, if  $k$  is too big, most local clusters may be sparsely populated and inevitably suffer from under-training. Thus care must be taken to choose an appropriate  $k$ , which is usually unknown beforehand. To find the appropriate value of  $k$ , one may need to exhaustively search all possible  $k$  and check the associated performance, demanding formidable computations.

---

**Algorithm 3:** Spectral Clustering with Eigen-Selection

---

**Input:** ambiguous data points  $\{\mathbf{x}_i | \mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$ ; eigen-selection parameters  $\alpha, \beta$  ( $0 < \alpha, \beta < 1$ ).

**Output:**  $k$  partitions of the input data.

1. Form the affinity matrix  $A$  with elements:

$$a_{ij} = \exp\left(-\frac{\|Dis(x_i, x_j)\|^2}{2\sigma_{ij}^2}\right), \quad i, j = 1, \dots, n;$$

2. Compute diagonal matrix  $D$  with elements  $d_i = \sum_{j=1}^n a_{ij}$ ;

3. Compute Normalized Laplacian matrix  $L$ :

$$L = D^{-\frac{1}{2}} \cdot (D - A) \cdot D^{-\frac{1}{2}};$$

4. Compute eigenvalues of  $L$  and sort in descending order:

$$\lambda_i \geq \lambda_{i+1}, i = 1, 2, \dots, n-1$$

5. Get  $k$  by eigenvalue selection:

**for**  $i = 2 \rightarrow n$  **do**

**if**  $\lambda_i \leq \alpha \cdot \lambda_{i-1}$  **or**  $\lambda_i \leq \beta$  **then**  
         **break**;

$k = i - 1$ ;

6. Form normalized matrix  $S$  using  $k$  largest eigenvectors;

7. Treating  $S$ 's row as point, cluster points by  $k$ -means;

8. Assign original data  $x_i$  to cluster  $j$  if and only if row  $i$  of the matrix  $S$  was assigned to cluster  $j$ ;

---

To solve this problem, we adopt spectral clustering to effectively find appropriate value of  $k$  and compute the clustering. Inspired by [42, 43] we search for certain drop in the magnitude of the eigenvalues to decide the number of clusters. The clustering algorithm is summarized in Algorithm (3).

In Algorithm (3), parameters  $\alpha$  and  $\beta$  define the criterion for selecting number of clusters, and  $Dis(x_i, x_j)$  is the distance between data points  $x_i$  and  $x_j$ . The scaling parameter

$\sigma_{ij}$  is a measure to decide whether two examples are similar, which can be specified by self-tuning [44]:

$$\sigma_{ij} = \sqrt{\sigma_i \cdot \sigma_j} \quad (3.1)$$

where,

$$\sigma_i = Dis(x_i, x_{k_{th}}) \quad (3.2)$$

In Eq.(3.2),  $x_{k_{th}}$  represents the  $k$ 'th nearest neighbor of point  $x_i$ , typically  $k=7$ . Although Algorithm (3) produces high-quality clustering result, the computational complexity of  $O(n^3)$  limits its application to large-scale data. Note that usually the number of ambiguous data is huge, so a fast approximation of spectral clustering should be applied, such as KASP [45], explained in step 3–6 of Algorithm (4).

### 3.3.3 Parameter Selection & Fast Approximation

The spectral clustering algorithm helps effectively avoid exhaustive search for optimal model complexity. To balance learning capacity, we determine the parameters of spectral clustering,  $\alpha$  and  $\beta$ , based on the accuracy of corresponding locally learned classifiers. Specifically, we randomly partition the ambiguous training data into  $M$  (typically  $M=10$ ) subsets for cross validation to find the optimal parameters. For each fold, we first apply step 1–4 of Algorithm (3) on training set and sort the eigenvalues in descending order. We search for drops between consecutive eigenvalues that are bigger than half of the prior eigenvalue. The corresponding  $\{\alpha, \beta\}$  are marked as candidate parameter sets. For each candidate set, we then apply step 6–8 of Algorithm (3) to construct clusters. Local learning algorithm (Section 3.4) is applied for each cluster and the detection rates are evaluated. Among all candidate parameter sets, we use the one with the best detection rate as the optimal parameters.

Given the clustering parameters  $\alpha$  and  $\beta$ , we can apply Algorithm (3) to do clustering. However as explained in Sec. 3.3.2, when the data set is large, directly applying Algo-

---

**Algorithm 4:** Accelerated Automatic Clustering
 

---

**Input:** ambiguous data points  $\{\mathbf{x}_i | \mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$ .

**Output:** same as Algorithm (3)

1. Randomly split  $\{\mathbf{x}_i\}_{i=1}^n$  into partitions  $\{\mathbf{P}_i\}_{i=1}^M$  for  $M$ -fold cross validation to find optimal parameters  $\alpha, \beta$ :

**for**  $m = 1 \rightarrow M$  **do**

- 1) Define training set  $\mathbf{P}_{tr}$  and validation set  $\mathbf{P}_{va}$   
 $\mathbf{P}_{tr} = \{\mathbf{x}_i\}_{i=1}^n - \mathbf{P}_m, \quad \mathbf{P}_{va} = \mathbf{P}_m;$
- 2) Apply **step. 1-4** of Algorithm (3) on  $\mathbf{P}_{tr}$  and get all eigenvalues  $\{\lambda'_i\}_{i=1}^{n_1}$ , where  $n_1 = |\mathbf{P}_{tr}|$ ;
- 3) Search candidate eigenvalue drops. Initialize  $T = 0$ :  
**for**  $i = 2 \rightarrow n_1$  **do**  
     **if**  $\lambda'_i \leq 0.5 \cdot \lambda'_{i-1}$  **then**  
          $T = T + 1;$   
          $k'_T = i - 1, \quad \alpha'_T = \lambda'_{i-1}/\lambda'_i, \quad \beta'_T = \lambda'_i;$   
     **end if**  
**end for**
- 4) Cluster and check performance for each candidate:  
**for**  $t = 1 \rightarrow T$  **do**  
     Apply **step. 6-8** of Algorithm (3) with  $k = k'_t$  on  $\mathbf{P}_{tr}$  to get clusters  $\mathbf{C}'_t$ ;  
     Learn local classifiers  $\mathbf{F}'_t$  on each cluster of  $\mathbf{C}'_t$ ;  
     Evaluate  $\mathbf{F}'_t$  on  $\mathbf{P}_{va}$  and get detection rate  $\varepsilon'_t$ ;  
**end for**
- 5)  $\{\alpha_m, \beta_m, k_m\} = \arg \min_{\{\alpha'_t, \beta'_t, k'_t\}} \{\varepsilon'_t\}_{t=1}^T$

2. Set optimal parameters:

$$\alpha = \frac{1}{M} \cdot (\sum_{m=1}^M \alpha_m), \quad \beta = \frac{1}{M} \cdot (\sum_{m=1}^M \beta_m)$$

3. Fix a  $k_0$ , where  $k_0 \geq 20 \cdot \frac{1}{M} \cdot (\sum_{m=1}^M k_m)$  and  $k_0 \ll n$ , then perform  $k_0$ -means on  $\{\mathbf{x}_i\}_{i=1}^n$  to get the cluster centroids  $\{\mathbf{y}_j\}_{j=1}^{k_0}$ ;
  4. Build a correspondence table to associate each  $x_i$  with the nearest cluster centroid  $y_j$ ;
  5. Run Algorithm (3) on  $\{\mathbf{y}_j\}_{j=1}^{k_0}$  to obtain the  $k$  cluster membership for each of  $y_i$ ;
  6. Recover the cluster membership for each  $x_i$  using that of the associated  $y_j$  by looking up the correspondence table.
- 

Algorithm (3) is computationally prohibitive. We use a fast approximation KASP to speed up this process. Specifically, for the desired  $k$  clusters, we first apply  $k_0$ -means and compute the centroid  $\{\mathbf{y}_j\}_{j=1}^{k_0}$  of each cluster ( $k_0$  defined in step 1–3 of Algorithm (4)). We then run Algorithm (3) on  $\{\mathbf{y}_j\}_{j=1}^{k_0}$ . The cluster membership of each sample  $x_i$  is recovered using a table of correspondences with  $y_j$ .

The complete proposed clustering algorithm with parameter selection and fast approximation is summarized in Algorithm (4). It not only speeds up the original clustering method in Algorithm (3), but also simplifies the query. During test, for each sample we simply compute its nearest neighbor in the  $k_0$  centers, then use the correspondence table to find its cluster membership. A sample of clustering results by our algorithm is shown in Figure (3.2), where different rows correspond to different clusters. As we can see, the proposed method works well: images from each row shares similar shape, background, and appearance.

### 3.4 Local Adaptation - Enhancing Learning Capacity

After the ambiguous data being appropriately clustered, local learning is used to enhance the learning capacity.

A straightforward local learning approach is to train a general classifier [38, 4] directly using the data from each local cluster. Considering speed and performance, linear SVM seems to be a good choice. However the disadvantage of direct learning is obvious: it only uses limited samples in a local cluster and discards the information from the whole dataset that are usually beneficial. Hence, the performance of generated local classifier heavily relies on the population and data distribution of the cluster, and often suffers from under-training.

To address this issue, we propose a model adaptation strategy that leverages on global classifier for effective local learning. Though the global classifier  $F_0$  trained in the first stage may not behave perfectly on each local cluster, it should not be too far from the optimum classification boundary. Furthermore, it contains non-negligible information about global data distribution. Therefore, we treat the local learning problem as utilizing a *coarse* global classifier  $F_0$  to adapt into different *fine* local classifiers. This effectively enhances the learning capacity of our classification algorithm in each local cluster.



Here we adopt Min-Max model adaptation [46] in our algorithm. Comparing with other state of the art adaptation methods, such as [16, 20, 21, 19], the Min-Max is free of tuning parameters (e.g., the adaptation rate) and able to adapt from general parametric classifier. Thus  $F_0$  could be trained by various complicated methods such as [38, 4]. We summarize our local adaptation approach in Algorithm (5).

---

**Algorithm 5:** Local Learning by Min-Max Adaptation

---

**Input:** Pre-learned global classifier  $F_0$  with parameters  $\mathbf{w}_0$ ;  $K$  clusters  $\{S_k\}_{k=1}^K$  obtained using Algorithm (4); Min-Max adaptation parameters.

**Output:** Adapted local classifiers  $\{F'_k\}_{k=1}^K$

**for**  $k = 1 \rightarrow K$  **do**

1. Form  $N$  data covers  $\{G_j\}_{j=1}^N$  from data in  $S_k$ ;
2. Build the cost functions  $\{C_j\}_{j=1}^N$  based on logistic regression;
3. Define cost function  $E(\mathbf{w})$  on  $S_k$ :

$$E(\mathbf{w}) = \left( \max_j [C_j(\mathbf{w})] \right)$$

4. Approximate  $E(\mathbf{w})$  as  $\infty$ -Norm:

$$E(\mathbf{w}) = \|\mathbf{C}(\mathbf{w})\|_\infty \approx \left( \sum_{j=1}^N [C_j(\mathbf{w})]^q \right)^{\frac{1}{q}}$$

5. Set  $\mathbf{w}^{[0]} = \mathbf{w}_0$ ,  $T = \infty$ ,  $i = 0$ ;

**while**  $T \geq \xi$  **do**

compute the gradient and Hessian matrix  $\nabla E(\mathbf{w}^{[i]})$ ,  $\mathbf{H}_E(\mathbf{w}^{[i]})$ , and update  $\mathbf{w}$ :  
 $\mathbf{w}^{[i+1]} = \mathbf{w}^{[i]} - \mathbf{H}_E^{-1}(\mathbf{w}^{[i]}) \cdot \nabla E(\mathbf{w}^{[i]})$ ;  
 $T = \sqrt{\nabla E(\mathbf{w}^{[i]})^T \cdot \mathbf{H}_E^{-1}(\mathbf{w}^{[i]}) \cdot \nabla E(\mathbf{w}^{[i]})}$ ;  
 $i = i + 1$ ;

6. Set parameter for classifier  $F'_k$ :  $\mathbf{w}'_k = \mathbf{w}^{[i]}$ ;
- 

## 3.5 Experiments

We evaluate our algorithm on pedestrian detection from images that is important and yet challenging in practice. We compare our algorithm with state of the art single layer (non-cascaded) detectors and demonstrate that our algorithm greatly improves the detection

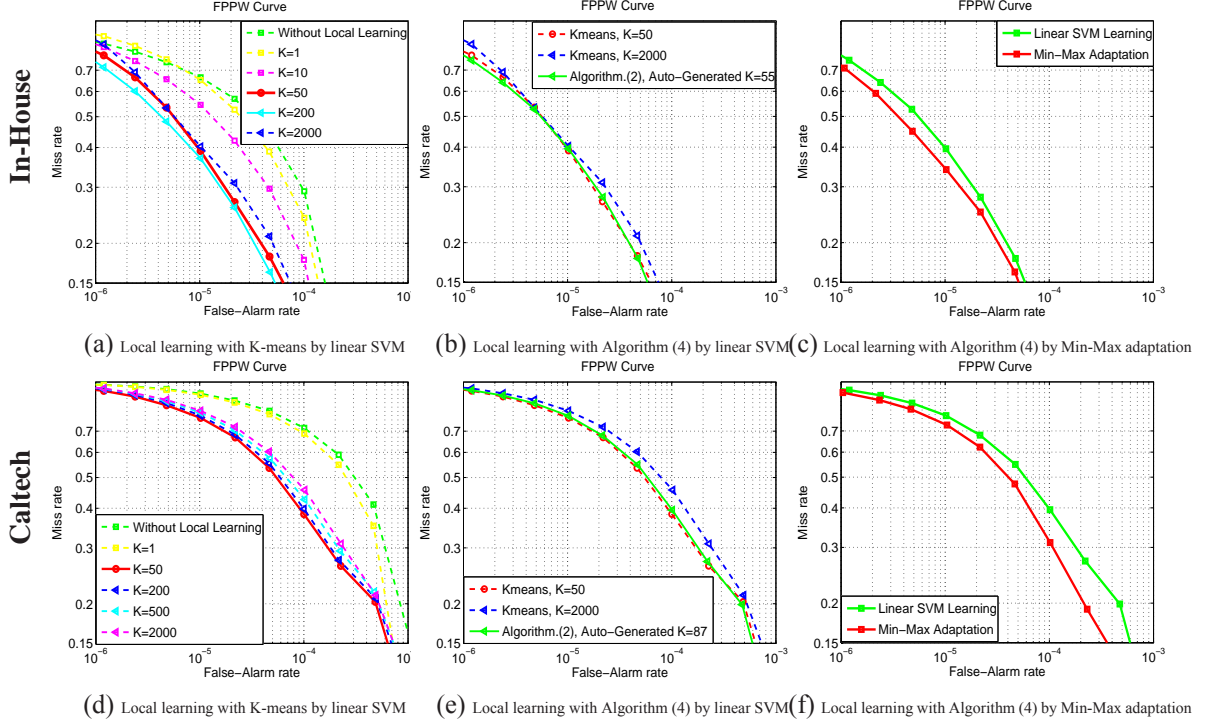


Figure 3.3: Evaluation of the proposed algorithm on In-House (1<sup>st</sup> row ) and Caltech (2<sup>nd</sup> row) datasets. All results are plotted as miss-rate w.r.t. false-alarm-rate in FPPW. The first column details performances of local linear SVM learning with  $k$ -means clustering, where  $k$  varies from 1 to 2000; The second column shows performance achieved by proposed clustering methods, compared with the best results in the first column; The final column compares performances achieved by 2 different local learning methods.

rates. We also compare our approach with state of art using the same evaluation methodology used in [1]. We show both qualitative and quantitative results on several challenging datasets to confirm the advantages of our approach.

The rest of this section consists of five parts: the first one explains the experiment design and implementation details; the second part tests our algorithm on a large challenging pedestrian dataset “In-House” collected by ourselves; the third and fourth parts respectively evaluate our algorithm on two popular benchmarks: Caltech [1] and INRIA [2] datasets; and finally the algorithm efficiency is discussed.

### 3.5.1 Experimental Setup

**Parameter setting.** In Algorithm (3) the eigenvalue-selection parameters  $\alpha, \beta$  are chosen as discussed in Sec. 3.3.3. In our experiments,  $\alpha=0.2$  and  $\beta=0.01$  yields the best performance. This holds true over cross-validation experiments with the In-House dataset. Later experiments confirm that the same setting also applies for Caltech and INRIA datasets. As stated before, the scaling parameter  $\sigma_{ij}$  could be automatically decided by self-tuning, which requires extra computation on neighbor searching to ensure good performance. In practice we found the local-learning performance is similar to self-tuning when  $\sigma_{ij}$  is between 2 and 5 and the performance is insensitive to different values of  $\sigma_{ij}$  in that range. Thus to speed up,  $\sigma_{ij}$  can be simply set as a constant, e.g. 3. The Min-Max adaptation parameters in Algorithms (5) are the same as [46].

**Image representation.** We test both shape (HOG [2]) and texture (LBP [34]) descriptors with our learning framework, since HOG has been widely accepted as one of the best features to capture the edge or local shape information and the LBP is an exceptional texture descriptor. We use only the HOG descriptor for In-House and Caltech datasets, and HOG together with LBP for INRIA dataset. Experimental results confirm that our algorithm can robustly find the clusters that yield good detection rates.

**Experiments design and evaluation measure.** First, to show the effect of clustering on detection accuracy, we cluster every dataset by simple  $k$ -means with different values of  $k$ . We then construct local classifiers for individual clusters by linear SVM and compare the overall performance with respect to different values of  $k$ . Second, to show the effectiveness of our clustering algorithm, we compare the performance of our clustering method with the best result achieved by  $k$ -means approach. Finally, we compare the performances improvement gained by two different local learning methods: directly learning by linear SVM and local adaption with Min-Max, not only on detection rate but on speed. For all three types of experiments, we plot detection curves in terms of FPPW instead of FPPI to evaluate the performance of classifiers, since FPPW allows a better assessment of the learn-

ing algorithm and isolates the impact of post-processing such as non-maximal suppression. Additionally, in order to compare with 11 algorithms [1] presented in Caltech dataset [1], we also present the accuracy of our detection system in FPPI for Caltech dataset.

### 3.5.2 In-House dataset

The In-House dataset is collected and labeled by ourselves, containing 5691 images with fixed resolution at  $320 \times 240$ . Performing detection on this dataset is challenging due to the fact that many of the pedestrians are small, partly occluded, and hardly identifiable from background even by human eyes. We randomly divide the dataset into 3 partitions for 3-fold cross-validation.

Firstly, we directly apply  $k$ -means clustering followed by local learning using linear SVM with  $k$  varying from 1 to 2000. The results with different  $k$  are shown in Figure 3.3(a).  $k = 1$  means considering all ambiguous data as a single cluster and training for one classifier, similar to a re-training on the whole dataset. As we can see, when  $k$  increases from 1 to 200, the performance improves logarithmically. However, if  $k$  exceeds 200, the performance starts to drop. The results at  $k=50$  or 200 are similarly the best. Since smaller number of clusters leads to better efficiency in the testing stage, we take  $k=50$  as the optimal clustering. The results confirm that an appropriate local learning would greatly improve the detection performance. Compared with the traditional approach without local adaptation, the miss rate at  $k=50$  is significantly reduced by 25% at  $10^{-5}$  and 16% at  $10^{-4}$  False-Alarm rate in FPPW.

Secondly, we evaluate our local learning algorithm with the proposed clustering and compare with the best results in the first experiment. With a proper  $k_0=2000$ , the number of clusters computed by Algorithm (4) is 55. We then perform local learning with linear SVM on individual clusters. As shown in Figure 3.3(b), our method achieves the results as good as the best one in the first experiment. It confirms that our method autonomously strikes a good balance between model complexity and learning capacity.

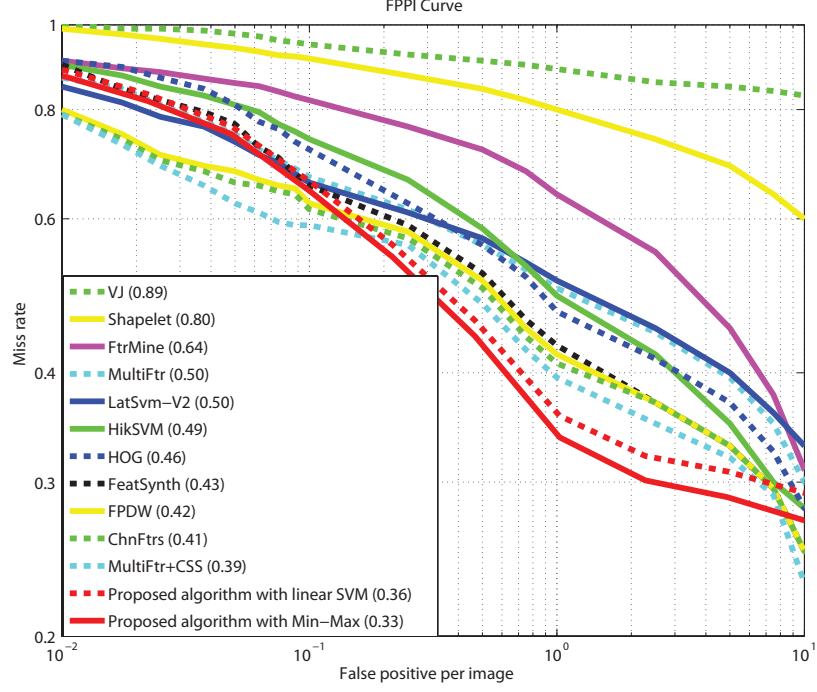


Figure 3.4: Comparison between proposed algorithm and state of the art on Caltech *training* dataset, adopting the same experimental setting and evaluation methodology as [1], plotted as miss-rate w.r.t. FPPI. The precision values at FPPI=1 are shown for easy comparison.

Finally, we compare the two local classification methods. For fair comparison, we learn classifiers based on the proposed clustering in the second experiment. The detection performance of the two types of local learning algorithms are shown in Figure 3.3(c). As we can see, the Min-Max adaptation method performs better, further reducing the miss rate by 5% at  $10^{-4}$  FPPW. Training the linear SVMs is much faster, 55 local classifiers taking only half an hour to train, while Min-Max adaptation takes 1 day. Overall, the proposed local adaptation algorithm achieves the best detection rates, reducing the miss rate by 30% at  $10^{-5}$  and 21% at  $10^{-4}$  FPPW compared with the single global classifier.

### 3.5.3 Caltech dataset

Caltech dataset is currently one of the most challenging pedestrian datasets, since pedestrians appear from multiple viewpoints and with a wide range of scales. Additionally, lots

of cars and buildings make the background very cluttered. The labeled Caltech *training* dataset contains six sessions (S0-S5), each with multiple videos taken from a moving vehicle. We follow the same 6-folder-cross-validation evaluation methodology as [1] and only consider the “Reasonable” [1] pedestrians.

Following the same step as In-House dataset, we first evaluate the performance with  $k$ -means where  $k$  varies from 1 to 2000. Figure 3.3(d) show our experiment results. We can see that clusters with  $k=50$  achieve the best result, outperforming the traditional methods by 15% at  $10^{-5}$  and 35% at  $10^{-4}$  in FPPW. Then we test our method with the proposed clustering, which automatically clusters data in 87 classes. The detection rate is similar to the best case in the previous experiment, detailed in Figure 3.3(e). Finally, we compare our local adaptation with direct local learning, as shown in Figure 3.3(f). Again Min-Max model adaptation achieves about 10% higher detection rate at  $10^{-4}$  FPPW while taking longer time to train. Figure (3.6) show some example images.

**Comparison with state-of-the-art** Taking the same evaluation procedures as [1], we show the comparison with 11 algorithms [1] on Caltech *training* dataset. From the FPPI curves in Figure (3.4), our algorithm achieves the lowest miss rate at 1 FPPI, 6% lower than the best in [1]. Our method only uses the HOG feature, while many of the 11 algorithms combine several different descriptors. It tells more about the advantage of our method when only comparing with methods using similar features, such as HOG [2], LatSVM-V2 [4], HikSVM [7]. As shown in Figure (3.4), our method outperforms them by 13%, 17%, and 16% at 1 FPPI respectively.

### 3.5.4 INRIA dataset

INRIA [2] dataset is also popular for researchers to test human detectors. We evaluate on it to show the advantage of our method and its robustness on handling complex backgrounds and using different kinds of features.

Again, we start with testing the performance of  $k$ -means clustering. From Figure (3.5),

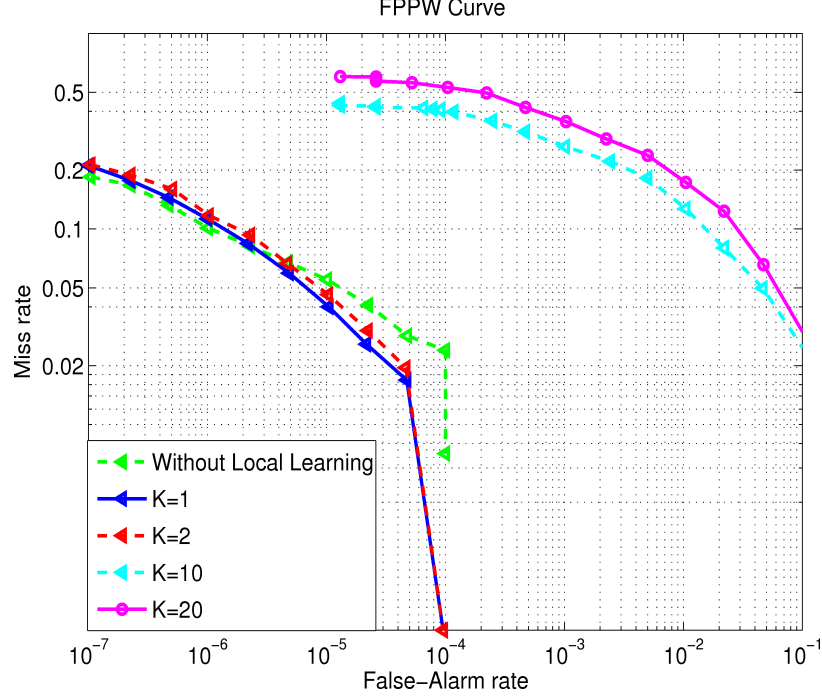


Figure 3.5: Evaluation of local SVM learning after k-means clustering on INRIA dataset. When  $k = 1$  or 2 the performance increases slightly. However, further increasing  $k$  deteriorates the performance quickly.

we can see that when  $k=1$  or 2, the performance has a little improvement than that without local adaptation. However, when the cluster number increases, the performance drops quickly. When  $k=10$ , the performance drastically decreases by 50% at  $10^{-4}$  FPPW. Inappropriately divide them into smaller clusters will lead to under-training. It also confirms that carefully balancing between model complexity and learning capacity is critical for local learning algorithm to achieve desired performance improvement. The reason is that, for INRIA dataset, the traditional method (HOG+LBP) has already achieved an impressively high detection rate, and the few remaining ambiguous data are difficult to be clustered in feature space (only 2% miss detection in  $10^{-4}$  FPPW). Therefore, instead of gaining improvement in performance, forcing clustering and local learning would make it worse. In such a case, the proposed clustering method is able to automatically identify the proper number of cluster is 1, warning us over-increasing model complexity would break the balance with learning capacity and deteriorate the detection performance. It thus ensures

reliable local learning.

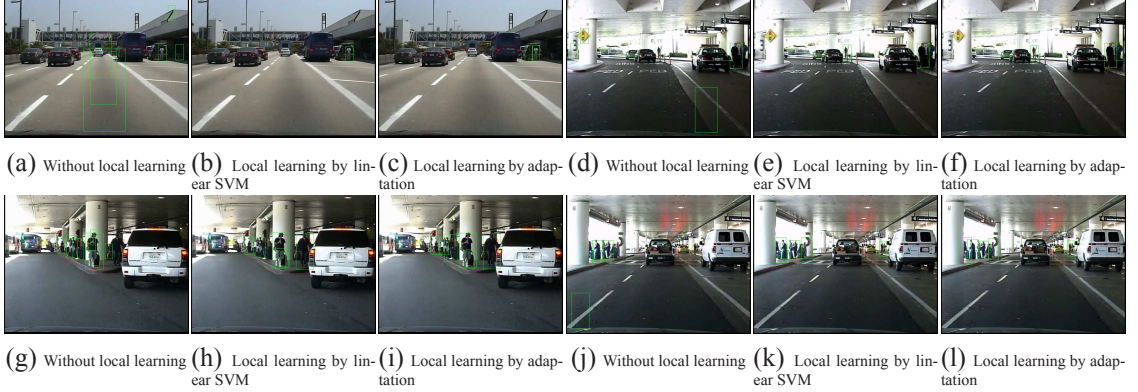


Figure 3.6: Comparison of detection results on Caltech dataset, between one-stage global classifier without local learning, local learning by linear SVM, and local learning by Min-Max adaptation. The last approach achieves both the best detection rate and the lowest false alarm rate.

### 3.5.5 Computational Complexity in Testing

Compared with one-stage learning methods, the only extra computation for our algorithm is cluster query and classification by corresponding local classifier. Since we use hierarchical  $k$ -means with cluster number  $k_0$  before fast spectral clustering, the cluster query computes  $\log(k_0)$  times of distance, while SVM-KNN [24] needs at least  $\log(n_{train})$  times ( $n_{train}$  is the number of samples in training set, where  $n_{train} \gg k_0$ ), and more cost on training kernel SVM during test. From our experiments, adding the local-adaptive stage only takes less than 10% extra time during test.

## 3.6 Conclusion and Future Work

We developed a hybrid learning algorithm combining global classification and local adaptations, which automatically adjusts model complexity according to data distribution. The



proposed adaptation algorithm automatically determines model complexity of the local learning algorithm according to the distribution of the training samples.

In term of classification algorithm, we have shown that our method effectively improves the performance compared with the state-of-art methods, especially when using similar features. On the other hand, the recently proposed features such as Integral Channel Features [47] and Multi-Resolution Features [48] have been successfully used for pedestrian detection and achieved highly competitive results. We plan to incorporate such features into our hybrid learning algorithm and believe this can further improve the detection performance.

## Chapter 4

### Conclusion

In the thesis, we have proposed two algorithms: 1) an object detector adaptation algorithm which has little worry about tuning many parameters. This idea is to generate several adaptation data covers from the adaptation dataset according to certain rules, and during each adaptation iteration, always consider the worse data cover. The adaptation problem then changes into a Min-Max problem which could be solved by approximation of infinite norm method. This algorithm has the ability to train performance-guaranteed detectors for different categories in various environments. This characteristic of our algorithm is also demonstrated in experiments. 2) We developed a hybrid learning algorithm combining global classification and local adaptations, which automatically adjusts model complexity according to data distribution. The proposed adaptation algorithm automatically determines model complexity of the local learning algorithm according to the distribution of the training samples. In term of classification algorithm, we have shown that our method effectively improves the performance compared with the state-of-art methods, especially when using similar features.

# Bibliography

- [1] B. Schiele P. Dollar, C. Wojek and P. Perona. Pedestrian detection: A benchmark. In *CVPR*, 2009.
- [2] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.
- [3] P. Viola and M.J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, May 2004.
- [4] Pedro F. Felzenszwalb, David A. McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.
- [5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.
- [6] Long Zhu, Yuanhao Chen, Alan L. Yuille, and William T. Freeman. Latent hierarchical structural learning for object detection. In *CVPR*, 2010.
- [7] Subhransu Maji, Alexander C. Berg, and Jitendra Malik. Classification using intersection kernel support vector machines is efficient . In *ICCV*, 2009.
- [8] Chaitanya Desai, Deva Ramanan, and Charles Fowlkes. Discriminative models for multi-class object layout. In *ICCV*, 2009.

- [9] M. Varma A. Vedaldi, V. Gulshan and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.
- [10] Qiang Zhu, Shai Avidan, Mei chen Yeh, and Kwang ting Cheng. Fast human detection using a cascade of histograms of oriented gradients. In *CVPR*, 2006.
- [11] Yuan Li, Haizhou Ai, Takayoshi Yamashita, Shihong Lao, and Masato Kawade. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2008.
- [12] Junge Zhang, Kaiqi Huang, Yinan Yu, and Tieniu Tan. Boosted local structured hog-lbp for object localization. In *CVPR*, 2010.
- [13] Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.
- [14] Shai Avidan. Ensemble tracking. In *CVPR*, pages 494–501, 2005.
- [15] Robert T. Collins, Yanxi Liu, and Marius Leordeanu. On-line selection of discriminative tracking features. In *ICCV*, 2003.
- [16] Helmut Grabner and Horst Bischof. On-line boosting and vision. In *CVPR*, 2006.
- [17] Yuan Li and Haizhou Ai. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different lifespans. In *CVPR*, 2007.
- [18] Xiaoming Liu and Ting Yu. Gradient feature selection for online boosting. In *ICCV*, 2007.
- [19] Minh-Tri Pham and Tat-Jen Cham. Online learning asymmetric boosted classifiers for object detection. In *CVPR*, 2007.

- [20] Gert Cauwenberghs and Tomaso Poggio. Incremental and Decremental Support Vector Machine Learning. In *NIPS*, pages 409–415, 2000.
- [21] Cha Zhang, Raffay Hamid, and Zhengyou Zhang. Taylor expansion based classifier adaptation: Application to person detection. In *CVPR*, 2008.
- [22] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995.
- [23] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.
- [24] Hao Zhang, Alexander C. Berg, Michael Maire, and Jitendra Malik. SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. 2006.
- [25] Eon Bottou and Vladimir Vapnik. Local learning algorithms. *Neural Computation*, 1992.
- [26] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee S. Lee. Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.
- [27] E. Robert. A Brief Introduction to Boosting. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, 1999.
- [28] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28, 2000.
- [29] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning internal representations by error propagation*, pages 673–695. MIT Press, Cambridge, MA, USA, 1988.
- [30] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer Science++ Business Media, LLC, 2006. 3.

- [31] C. Wojek, S. Walk, and B. Schiele. Multi-cue onboard pedestrian detection. *CVPR*, 2009.
- [32] *The PASCAL Visual Object Classes Challenge 2007*.  
<http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/>.
- [33] *The ImageNet dataset*. <http://www.image-net.org/index>.
- [34] Xiaoyu Wang, Xu Han, and Shuicheng Yan. An hog-lbp human detector with partial occlusion handling. In *ICCV*, 2009.
- [35] *The WordNet dataset*. <http://wordnet.princeton.edu/>.
- [36] Zhuowen Tu. Probabilistic Boosting-Tree: Learning Discriminative Models for Classification, Recognition, and Clustering. *ICCV*, 2005.
- [37] Boris Babenko, P. Dollar, Zhuowen Tu, and Serge Belongie. Simultaneous learning and alignment: Multi-instance and multi-pose learning. In *ECCV Faces in Real-Life Images*, 2008.
- [38] Hedi Harzallah, Frédéric Jurie, and Cordelia Schmid. Combining efficient object localization and image classification. In *ICCV*, sep 2009.
- [39] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on PAMI*, 2001.
- [40] Thomas Leung and Jitendra Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Comput. Vision*, 2001.
- [41] L.Levina. *Statistical Issues in Texture Analysis*. PhD thesis, Department of Statistic, University of California, Berkeley, 2002.
- [42] Marzia Polito and Pietro Perona. Grouping and dimensionality reduction by locally linear embedding. In *NIPS*, 2001.

- [43] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, 2001.
- [44] Lihi Zelnik-manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems 17*, 2004.
- [45] Donghui Yan, Ling Huang, and Michael I. Jordan. Fast approximate spectral clustering. In *SIGKDD '09*, 2009.
- [46] Guang Chen, Tony X. Han, and Shihong Lao. Adapting an object detector by considering the worst case: a conservative approach. In *CVPR*, 2011.
- [47] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral Channel Features. In *British Machine Vision Conference*, 2009.
- [48] Dennis Park, Deva Ramanan, and Charless Fowlkes. Multiresolution models for object detection. In *ECCV*, 2010.