Learning Prior Bias in Classifier

Takumi Kobayashi(✉) and Kenji Nishida

National Institute of Advanced Industrial Science and Technology, 1-1-1 Umezono, Tsukuba, Japan
{takumi.kobayashi,kenji.nishida}@aist.go.jp

Abstract. In pattern classification, a classifier is generally composed both of feature (vector) mapping and bias. While the mapping function for features is formulated in either a linear or a non-linear (kernel-based) form, the bias is simply represented by a constant scalar value, rendering prior information on class probabilities. In this paper, by focusing on the *prior bias* embedded in the classifier, we propose a novel method to discriminatively learn not only the feature mapping function but also the prior bias based on the extra prior information assigned to samples other than the class category, *e.g.*, the 2-D position where the local image feature is extracted. Without imposing specific probabilistic models, the proposed method is formulated in the framework of maximum margin to adaptively optimize the biases, improving the classification performance. We present a computationally efficient optimization approach for making the method applicable even to large-scale data. The experimental results on patch labeling in the on-board camera images demonstrate the favorable performance of the proposed method in terms of both classification accuracy and computation time.

Keywords: Pattern classification · Bias · Discriminative learning · SVM

1 Introduction

Prior information has been effectively exploited in the fields of computer vision and machine learning, such as for shape matching [1], image segmentation [2], graph inference [3], transfer learning [4] and multi-task learning [5]. Learning prior has so far been addressed mainly in the probabilistic framework on the assumption that the prior is defined by a certain type of generative probabilistic model [6,7]; especially, non-parametric Bayesian approach further considers the hyper priors of the probabilistic models [8].

In this paper, we focus on the (linear) classifier, $y = \boldsymbol{w}^\top \boldsymbol{x} + b$, and especially on the bias term, so called 'b' term [9]. The bias is also regarded as rendering the prior information on the class probabilities [10,11] and we aim to learn the *unstructured* prior bias b without assuming any specific models, while some transfer learning methods are differently built upon the prior of the weight $\boldsymbol{w}$ for effectively transferring the knowledge into the novel class categories [4,12] and the prior of $\boldsymbol{w}$ also induces a regularization on $\boldsymbol{w}$. While the bias b is generally

A. Fred et al. (Eds.): ICPRAM 2014, LNCS 9443, pp. 62–76, 2015.
DOI: 10.1007/978-3-319-25530-9_5

set as a constant across samples depending only on the class category, in this study we define it adaptively based on the extra prior information other than the class category. Note that, in the case of non-linear classification, the above classifier can be similarity formulated by $y = \boldsymbol{w}_\phi^\top \boldsymbol{\phi}_{\boldsymbol{x}} + b$ where the feature vector $\boldsymbol{x}$ is simply replaced with the kernel-based feature vector $\boldsymbol{\phi}_{\boldsymbol{x}}$ in the reproducing kernel Hilbert space defined by the kernel function $\mathsf{k}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}_{\boldsymbol{x}_i}^\top \boldsymbol{\phi}_{\boldsymbol{x}_j}$. Thereby, our focus also includes such kernel-based non-linear classifiers.

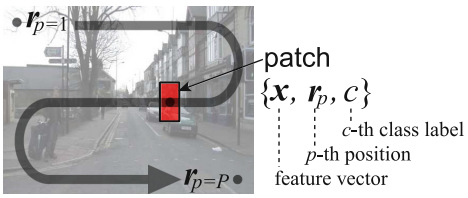


Fig. 1. The task of patch labeling is to predict the class labels c of the patches, each of which is represented by the appearance feature vector $\boldsymbol{x}$ and the prior position p. Note that there are P positions in total, $p \in \{1, \cdots, P\}$.

Suppose samples are associated with the extra prior information $p \in \{1, \cdots, P\}$ as well as the class category $c \in \{1, \cdots, C\}$, where P and C indicate the total number of the prior types and the class categories, respectively. For instance, in the task of labeling patches on the on-board camera images, each patch (sample) is assigned with the appearance feature vector $\boldsymbol{x}$, the class category c and the position (extra prior information) p, as shown in Fig. 1. The class category of the patch is effectively predicted by using not only the feature $\boldsymbol{x}$ but also the prior position p where the feature is extracted; the patches on an upper region probably belong to *sky* and the lower region would be *road*, even though the patches extracted from those two regions are both less textured, resulting in similar features.

The probabilistic structure that we assume in this study is shown in Fig. 2b with comparison to the `simple` model in Fig. 2a. By using generalized linear model [13], the standard classifier (Fig. 2a) is formulated to estimate the posterior on the class category c as[1]

$$\log \mathsf{p}(c|\boldsymbol{x}) \sim \log \mathsf{p}(\boldsymbol{x}|c) + \log \mathsf{p}(c) = \boldsymbol{w}_c^\top \boldsymbol{x} + b_c, \tag{1}$$

where $b_c = \log \mathsf{p}(c)$ indicates the class-dependent bias. On the other hand, the `proposed` model (Fig. 2b) using the extra prior p induces the following classifier;

$$\log \mathsf{p}(c|\boldsymbol{x}, p) \sim \log \mathsf{p}(\boldsymbol{x}|c) + \log \mathsf{p}(p|c) + \log \mathsf{p}(c) = \boldsymbol{w}_c^\top \boldsymbol{x} + b_c^{[p]}, \tag{2}$$

where the bias $b_c^{[p]} = \log \mathsf{p}(p|c) + \log \mathsf{p}(c)$ is dependent on both the class category c and the prior information p. Thus, if the bias could be properly determined,

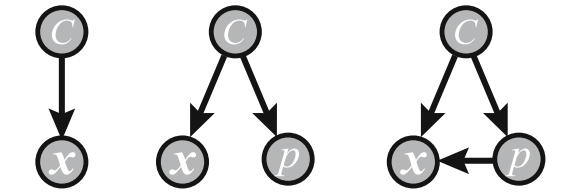
[1] '$\sim$' in (1) means the equality in disregard of the irrelevant constant term $\log \mathsf{p}(\boldsymbol{x})$ or $\log \mathsf{p}(\boldsymbol{x}, p)$ in (2) and (3).

the classification performance would be improved compared to the standard classification model (1). One might also consider the `full-connected` model shown in Fig. 2c whose classifier is formulated by

$$\log p(c|\boldsymbol{x}, p) \sim \log p(\boldsymbol{x}|c, p) + \log p(p|c) + \log p(c) = \boldsymbol{w}_c^{[p]\top} \boldsymbol{x} + b_c^{[p]}, \quad (3)$$

where the classifier weight $\boldsymbol{w}_c^{[p]}$ relies on the prior p as the bias $b_c^{[p]}$ does. This model is more complicated and consumes large memory storage since the classifier model $\{\boldsymbol{w}_c^{[p]}, b_c^{[p]}\}$ is prepared for respective priors $p = 1, \cdots, P$ and classes $c = 1, \cdots, C$. And, due to the high degree of freedom (D.O.F) of this model, it would be vulnerable to over-learning. These models are summarized in Table 1 and will be again discussed later.

In this paper, we propose a novel method for discriminatively learning the prior biases $b_c^{[p]}$ in (2) to improve the classification performance. The proposed method is formulated in the optimization problem based on the maximum margin criterion [14]. We also propose a computationally efficient approach for the optimization which contains large amount of samples drawn from all the priors $p \in \{1, \cdots, P\}$. Thereby, the proposed method is fast and applicable to large-scale data, while providing the high-performance classifier that exploits the extra prior information.



(a) `simple` (b) `proposed` (c) `full-connected`

Fig. 2. Graphical models to depict the probabilistic dependencies. The notations c, $\boldsymbol{x}$ and p denote the class category, the (appearance) feature vector and the extra prior information, respectively. The arrows show the probabilistic dependencies. (a) The feature $\boldsymbol{x}$ is simply drawn from the class category c in the `simple` model. (b) The `proposed` model incorporates the extra prior information p which is connected to $\boldsymbol{x}$ via c. (c) Those three variables are fully connected in the `full-connected` model.

2 Classifier Bias Learning

We detail the proposed method by first defining the formulation for learning the biases and then presenting the computationally efficient approach to optimize them. As we proceed to describe a general form regarding the prior biases p, for better understanding, it might be helpful to refer to the task of labeling patches shown in Fig. 1; the sample is represented by the appearance feature vector $\boldsymbol{x}$ and the prior position $p \in \{1, \cdots, P\}$.

Table 1. Classification methods for c-th class category. The dimensionality of the feature vector is denoted by D, $\boldsymbol{x} \in \mathbb{R}^D$, and the number of extra prior types is P.

Method	Model	D.O.F
`simple`	$y_c = \boldsymbol{w}_c^\top \boldsymbol{x} + b_c$	$D+1$
`proposed`	$y_c = \boldsymbol{w}_c^\top \boldsymbol{x} + b_c^{[p]}$	$D+P$
`full-connected`	$y_c = \boldsymbol{w}_c^{[p]\top} \boldsymbol{x} + b_c^{[p]}$	$PD+P$

2.1 Formulation

We consider a binary classification problem for simplicity and take a one-vs-rest approach for multi-class tasks. Suppose we have P types of extra prior information, and let $\boldsymbol{x}_i^{[p]} \in \mathbb{R}^D$ denote the D-dimensional feature vector of the i-th sample ($i = 1, \cdots, n^{[p]}$) drawn from the p-th type of prior. As described in Sect. 1, we deal with the classification defined by

$$y = \boldsymbol{w}^\top \boldsymbol{x}^{[p]} + b^{[p]}, \tag{4}$$

where y denotes the classifier output which is subsequently thresholded by zero for performing binary classification, and $\boldsymbol{w}$ and $b^{[p]}$ are the classifier weight vector and the bias, respectively. Note again that the bias $b^{[p]}$ depends on the p-th type of prior, $p \in \{1, \cdots, P\}$. The classifier (4) can be optimized via the following formulation in the framework of maximum margin [14];

$$\min_{\boldsymbol{w}, \{b^{[p]}\}_p} \frac{1}{2}\|\boldsymbol{w}\|^2 + C \sum_p^P \sum_i^{n^{[p]}} \xi_i^{[p]} \tag{5}$$

$$s.t.\ \forall p \in \{1, \cdots, P\},\ \forall i \in \{1, \cdots, n^{[p]}\},\ y_i^{[p]}(\boldsymbol{w}^\top \boldsymbol{x}_i^{[p]} + b^{[p]}) \geq 1 - \xi_i^{[p]},\ \xi_i^{[p]} \geq 0,$$

where C is the cost parameter. This is obviously convex and its Lagrangian is written by

$$L = \frac{1}{2}\|\boldsymbol{w}\|^2 + C \sum_p^P \sum_i^{n^{[p]}} \xi_i^{[p]} - \sum_p^P \sum_i^{n^{[p]}} \beta_i^{[p]} \xi_i^{[p]} - \sum_p^P \sum_i^{n^{[p]}} \alpha_i^{[p]} \{ y_i^{[p]} (\boldsymbol{w}^\top \boldsymbol{x}_i^{[p]} + b^{[p]}) - 1 + \xi_i^{[p]} \}, \tag{6}$$

where we introduce the Lagrange multipliers $\alpha_i^{[p]} \geq 0$, $\beta_i^{[p]} \geq 0$. The derivatives of the Lagrangian are

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_p^P \sum_i^{n^{[p]}} \alpha_i^{[p]} y_i^{[p]} \boldsymbol{x}_i^{[p]} = \boldsymbol{0} \Rightarrow \boldsymbol{w} = \sum_p^P \sum_i^{n^{[p]}} \alpha_i^{[p]} y_i^{[p]} \boldsymbol{x}_i^{[p]} \tag{7}$$

$$\frac{\partial L}{\partial \xi_i^{[p]}} = C - \alpha_i^{[p]} - \beta_i^{[p]} = 0 \Rightarrow 0 \leq \alpha_i^{[p]} \leq C \tag{8}$$

$$\frac{\partial L}{\partial b^{[p]}} = \sum_i^{n^{[p]}} \alpha_i^{[p]} y_i^{[p]} = 0. \tag{9}$$

Thereby, the dual is finally obtained as

$$\min_{\{\alpha_i^{[p]}\}_{i,p}} \frac{1}{2}\sum_{p,q}^{P}\sum_{i}^{n^{[p]}}\sum_{j}^{n^{[q]}} \alpha_i^{[p]}\alpha_j^{[q]}y_i^{[p]}y_j^{[q]}{\boldsymbol{x}_i^{[p]}}^\top \boldsymbol{x}_j^{[q]} - \sum_{p}^{P}\sum_{i}^{n^{[p]}} \alpha_i^{[p]} \tag{10}$$

$$s.t.\ \forall p,\ \sum_{i}^{n^{[p]}} \alpha_i^{[p]} y_i^{[p]} = 0,\ \forall i, \forall p,\ 0 \leq \alpha_i^{[p]} \leq C.$$

This is a quadratic programming (QP) analogous to the dual of SVM [15] except that there exist P linear equality constraints with respect to $\boldsymbol{\alpha}^{[p]}$. The standard QP solver is applicable to optimize (10), though requiring substantial computation cost. For solving QP of the SVM dual, the method of sequential minimal optimization (SMO) [16] is successfully applied, but in this case, we can not employ it directly due to the multiple equality constraints. In what follows, we present a computationally efficient approach to optimize (10).

2.2 Optimization

A large number of variables $\{\alpha_i^{[p]}\}_{i,p}$ in the QP (10) are inherently partitioned into block-wise variables regarding the prior p; we obtain P blocks of $\boldsymbol{\alpha}^{[p]} = \{\alpha_i^{[p]}\}_{i=1,\cdots,n^{[p]}} \in \mathbb{R}^{n^{[p]}}$, $p = 1, \cdots, P$. According to those block-wise variables, (10) is decomposed into the following sub-problem as well:

$$\min_{\boldsymbol{\alpha}_i^{[p]}} \frac{1}{2}\sum_{i,j}^{n^{[p]}} \alpha_i^{[p]}\alpha_j^{[p]}y_i^{[p]}y_j^{[p]}{\boldsymbol{x}_i^{[p]}}^\top \boldsymbol{x}_j^{[p]} - \sum_{i}^{n^{[p]}} \alpha_i^{[p]} \left\{1 - y_i^{[p]}\sum_{q\neq p}^{P}\sum_{j}^{n^{[q]}} \alpha_j^{[q]}y_j^{[q]}{\boldsymbol{x}_i^{[p]}}^\top \boldsymbol{x}_j^{[q]}\right\} \tag{11}$$

$$s.t.\ \sum_{i}^{n^{[p]}} \alpha_i^{[p]} y_i^{[p]} = 0,\ \forall i,\ 0 \leq \alpha_i^{[p]} \leq C. \tag{12}$$

This is again a quadratic programming which resembles the SVM dual except for the linear term with respect to $\boldsymbol{\alpha}^{[p]}$ and thus is effectively optimized by using the SMO [16]. Therefore, the whole procedure for optimizing (10) consists of iteratively optimizing the sub-problem (11) with respect to every prior $p \in \{1, \cdots, P\}$ by means of SMO. According to [17], the order of the priors p to be optimized is randomly permuted. The detailed procedures are shown in Algorithm 1.

In order to discuss the convergence of the iterative optimization, we mention the KKT condition of (10) [18]. The optimizer $\alpha_i^{[p]}$ satisfies the following condition:

$$G_{i,p}(\boldsymbol{\alpha}) + b_i^{[p]}y_i^{[p]} = \lambda_i^{[p]} - \mu_i^{[p]}, \tag{13}$$

$$\lambda_i^{[p]}\alpha_i^{[p]} = 0,\ \mu_i^{[p]}(C - \alpha_i^{[p]}) = 0, \tag{14}$$

$$\lambda_i^{[p]} \geq 0,\ \mu_i^{[p]} \geq 0, \tag{15}$$

where $G_{i,p}(\boldsymbol{\alpha}) = y_i^{[p]} \boldsymbol{x}_i^{[p]\top} \sum_q^P \sum_j^{n^{[q]}} \alpha_j^{[q]} y_j^{[q]} \boldsymbol{x}_j^{[q]} - 1$ is the derivative of the objective function in (10) with respect to $\alpha_i^{[p]}$. This condition is rewritten into

$$\alpha_i^{[p]} < C : \; G_{i,p}(\boldsymbol{\alpha}) + b_i^{[p]} y_i^{[p]} \geq 0, \tag{16}$$

$$\alpha_i^{[p]} > 0 : \; G_{i,p}(\boldsymbol{\alpha}) + b_i^{[p]} y_i^{[p]} \leq 0, \tag{17}$$

and since $y_i^{[p]} \in \{+1, -1\}$, it results in

$$-y_i^{[p]} G_{i,p}(\boldsymbol{\alpha}) \begin{cases} \leq b_i^{[p]} \; i \in \mathbb{I}_+^{[p]} \\ \geq b_i^{[p]} \; i \in \mathbb{I}_-^{[p]} \end{cases}, \tag{18}$$

where

$$\mathbb{I}_+^{[p]} = \{i | (\alpha_i^{[p]} < C \wedge y_i^{[p]} = 1) \vee (\alpha_i^{[p]} > 0 \wedge y_i^{[p]} = -1)\}, \tag{19}$$

$$\mathbb{I}_-^{[p]} = \{i | (\alpha_i^{[p]} < C \wedge y_i^{[p]} = -1) \vee (\alpha_i^{[p]} > 0 \wedge y_i^{[p]} = 1)\}. \tag{20}$$

Therefore, we can conclude that $\alpha_i^{[p]}$ is a stationary point if and only if

$$\delta^{[p]} \triangleq \left[\max_{i \in \mathbb{I}_+^{[p]}} -y_i^{[p]} G_{i,p}(\boldsymbol{\alpha})\right] - \left[\min_{i \in \mathbb{I}_-^{[p]}} -y_i^{[p]} G_{i,p}(\boldsymbol{\alpha})\right] \leq 0. \tag{21}$$

On the basis of this measure, we can stop the iteration when $\max_p \delta^{[p]} < \epsilon$ with a small tolerance $\epsilon > 0$. At the optimum, the bias $b^{[p]}$ is retrieved by

$$b^{[p]} = \frac{1}{|\mathbb{I}^{[p]}|} \sum_{i \in \mathbb{I}^{[p]}} -y_i^{[p]} G_{i,p}(\boldsymbol{\alpha}), \text{ where } \mathbb{I}^{[p]} = \{i | 0 < \alpha_i^{[p]} < C\}, \tag{22}$$

since the right hand side in (13) equals zero for $i \in \mathbb{I}^{[p]}$.

2.3 Trivial Biases

Finally, we mention the trivial sub-problem for further reducing the computational cost in the optimization. From a practical viewpoint, the samples of the two class categories are not equally distributed across the priors $p = 1, .., P$ but are localized in limited number of priors. For instance, in the case of on-board camera images, the *road* never appears in upper regions where the *sky* usually dominates. That is, we occasionally encounter the following sub-problem;

$$\min_{\boldsymbol{\alpha}_i^{[p]}} \frac{1}{2} \sum_{i,j}^{n^{[p]}} \alpha_i^{[p]} \alpha_j^{[p]} y_i^{[p]} y_j^{[p]} \boldsymbol{x}_i^{[p]\top} \boldsymbol{x}_j^{[p]} - \sum_i^{n^{[p]}} \alpha_i^{[p]} \left\{ 1 - y_i^{[p]} \sum_{q \neq p}^{P} \sum_j^{n^{[q]}} \alpha_j^{[q]} y_j^{[q]} \boldsymbol{x}_i^{[p]\top} \boldsymbol{x}_j^{[q]} \right\} \tag{23}$$

$$s.t. \sum_i^{n^{[p]}} \alpha_i^{[p]} y_i^{[p]} = 0, \; \forall i, \; 0 \leq \alpha_i^{[p]} \leq C, \; \underline{\forall i, \; y_i^{[p]} = 1} \text{ (or } \underline{\forall i, \; y_i^{[p]} = -1}). \tag{24}$$

The above QP is trivially optimized by $\boldsymbol{\alpha}^{[p]} = \mathbf{0}$ without exhaustive computation due to the constraint (24). Thus, by eliminating such priors that result in trivial optimization, we can reduce the computational burden of the whole procedure to optimize (10); see line 1 in Algorithm 1.

The only issue in this trivial case is how to determine the bias $b^{[p]}$. In this case, the bias is not uniquely determined but accepts any value satisfying the following;

$$y^{[p]} = +1 : b^{[p]} \geq \max_i -\boldsymbol{w}^\top \boldsymbol{x}_i^{[p]} + 1, \tag{25}$$

$$y^{[p]} = -1 : b^{[p]} \leq \min_i -\boldsymbol{w}^\top \boldsymbol{x}_i^{[p]} - 1. \tag{26}$$

Thus, we can provide three alternative ways for computing the bias on the trivial prior.

1. Tight bias:

$$b^{[p]} = \begin{cases} \max_i -\boldsymbol{w}^\top \boldsymbol{x}_i^{[p]} + 1 & (for\ y^{[p]} = +1) \\ \min_i -\boldsymbol{w}^\top \boldsymbol{x}_i^{[p]} - 1 & (for\ y^{[p]} = -1) \end{cases}. \tag{27}$$

Algorithm 1. Bias Learning.

Input: $\{\boldsymbol{x}_i^{[p]}, y_i^{[p]}\}$: feature vector and its class label of the i-th training sample from the p-th type of prior, $p = 1, .., P$, $i = 1, .., n^{[p]}$.
$\epsilon > 0$: small tolerance for terminating the iteration.

1: $\mathbb{P} = \{p | \exists i, y_i^{[p]} = 1 \wedge \exists i, y_i^{[p]} = -1\}$
2: Initialization: $\forall p \in \{1, .., P\}$, $\boldsymbol{\alpha}^{[p]} = \mathbf{0}$
3: **repeat**
4: Random permutation of $\{1, \cdots, P\}$: $\{\pi(1), \cdots, \pi(P)\}$
5: **for** $i = 1$ to P **do**
6: $p \leftarrow \pi(i)$
7: Set $\boldsymbol{\alpha}^{[p]}$ as the optimizer of (11)
8: **end for**
9: **until** $\max_{p \in \mathbb{P}} \delta^{[p]} < \epsilon$

Output: $\boldsymbol{w}$ computed by (7) and $\{b^{[p]}\}_{p=1,..,P}$ computed by (22) for $p \in \mathbb{P}$ and (27∼29) for $p \notin \mathbb{P}$, using the optimizers $\{\boldsymbol{\alpha}^{[p]}\}_p$.

This gives the tight bias based on the above conditions (25, 26), which is computed by using only the samples $\boldsymbol{x}_i^{[p]}$ belonging to the prior p.

2. Mild bias:

$$b^{[p]} = \begin{cases} \max_{i,p} -\boldsymbol{w}^\top \boldsymbol{x}_i^{[p]} + 1 & (for\ y^{[p]} = +1) \\ \min_{i,p} -\boldsymbol{w}^\top \boldsymbol{x}_i^{[p]} - 1 & (for\ y^{[p]} = -1) \end{cases}, \tag{28}$$

By considering whole samples $\{\boldsymbol{x}_i^{[p]}\}_{i,p}$ across the priors, the bias is determined with a margin from the tight one, which might improve the generalization performance to some extent.

3. Extreme bias:

$$b^{[p]} = \begin{cases} +\infty & (for\ y^{[p]} = +1) \\ -\infty & (for\ y^{[p]} = -1) \end{cases}, \tag{29}$$

By this bias, the samples from such a prior are definitely classified as positive (or negative) no matter how the appearance features of the samples are. In this case, the class category is solely dependent on the extra prior information via the bias $b^{[p]} \in \{+\infty, -\infty\}$.

These three ways are empirically compared in the experiments (Sect. 3.3).

2.4 Discussion

In the proposed method, all samples across all types of priors are leveraged to train the classifier, improving the generalization performance. In contrast, the `full-connected` method (Table 1) treats the samples separately regarding the priors, and thus the p-th classifier is learnt by using only a small amount of samples belonging to the p-th type of prior, which might degrade the performance. On the other hand, the `simple` method learning the classifier from the whole set of samples is less discriminative without utilizing the extra prior information associated with the samples. The proposed method effectively incorporate the prior information into the classifiers via the biases which are discriminatively optimized.

The proposed method is slightly close to the cross-modal learning [19,20]. The samples belonging to different priors are separated as if they are in different modalities, though the feature representations are the same in this case. The proposed method deals with them in a unified manner via the adaptive prior biases. Actually, the proposed method is applicable to the samples that are distributed differently across the priors; the sample distribution is shifted (translated) as $\boldsymbol{x}^{[q]} = \boldsymbol{x}^{[p]} + \boldsymbol{e}$ and the prior bias can adapt to it by $b^{[q]} = b^{[p]} - \boldsymbol{w}^\top \boldsymbol{e}$ since $y^{[p]} = \boldsymbol{w}^\top \boldsymbol{x}^{[p]} + b^{[p]}$, $y^{[q]} = \boldsymbol{w}^\top \boldsymbol{x}^{[q]} + b^{[q]} = \boldsymbol{w}^\top \boldsymbol{x}^{[p]} + (b^{[q]} + \boldsymbol{w}^\top \boldsymbol{e}) = y^{[p]}$. Therefore, the samples of the different priors are effectively transferred into the optimization to improve the classification performance.

3 Experimental Results

We evaluated the proposed method on a task of patch labeling in on-board camera images by using CamVid dataset [21]. This patch labeling contributes to understand the scene surrounding the car.

3.1 Setting

The CamVid dataset [21] contains several sequences composed of fully labeled image frames as shown in Fig. 3: each pixel is assigned with one of 32 class labels including 'void'. Those labeled images are captured at 10 Hz. In this experiment, we employ the major 11 labels frequently seen in the image frames, *road, building,*

On-board image Label image

Fig. 3. CamVid dataset [21].

sky, tree, sidewalk, car, column pole, sign symbol, fence, pedestrian and *bicyclist*, to form the 11-class classification task.

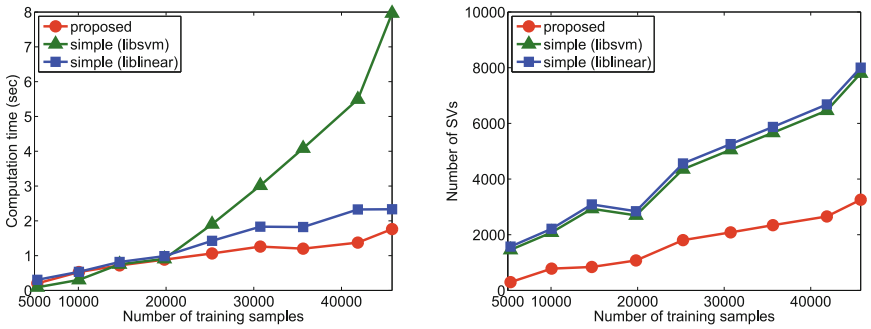
We extracted the GLAC image feature [22] from a local image patch of 20×40 pixels which slides at every 10 pixels over the resized image of 480×360. In this case, the feature vector $\boldsymbol{x} \in \mathbb{R}^{2112}$ is associated with the 2D position of the patch as the extra prior information; the total number of prior types (grid points) is $P = 1551$. Thus, the task is to categorize the patch feature vectors extracted at 1511 positions into the above-mentioned 11 classes.

We used three sequences in the CamVid dataset, and partitioned each sequence into three sub-sequences along the time, one of which was used for training and the others were for test. This cross validation was repeated three times and the averaged classification accuracy is reported.

For comparison, we applied the methods mentioned in Sect. 1; `simple` and `full-connected` methods as listed in Table 1. The `simple` method is a standard classification using the weight $\boldsymbol{w}$ with the constant bias b in disregard of the prior information p. The `full-connected` method applies classifiers comprising $\boldsymbol{w}^{[p]}$ and $b^{[p]}$ at respective priors $p = 1, \cdots, P$. This method requires tremendous memory storage for those P classifiers; in this experiment, 2112-dimensional weight vectors $\boldsymbol{w}^{[p]}$ and scalar bias $b^{[p]}$ in 11 class categories are stored at each of 1511 positions. On the other hand, in the `proposed` method, the feature vectors are classified by using the identical weight $\boldsymbol{w}$ across the priors together with the adaptively optimized bias $b^{[p]}$ depending on the prior p. We consider the linear classification form $y = \boldsymbol{w}^\top \boldsymbol{x} + b$ in all these methods for fast computation time.

3.2 Computation Cost

We first evaluated the proposed method in terms of computation cost. The method trains the classifier by using all the samples across the priors, scale of which is as large as in the `simple` method. These methods are implemented by MATLAB on Xeon 3.4 GHz PC. In the proposed method, we apply libsvm [23]



(a) Computation time (b) Number of support vectors (SVs)

Fig. 4. Comparison of the `simple` and `proposed` methods in terms of (a) computation time as well as (b) number of support vectors (SVs).

to solve QP and efficiently compute the derivatives $G_{i,p}(\boldsymbol{\alpha})$ required for the linear term in the objective function (11) and $\delta^{[p]}$ in (21) by exploiting the linear classification form as in [24]. On the other hand, two types of solvers, libsvm and liblinear [24], are applied to the `simple` method.

Figure 4a shows the computation time on various sizes of training samples. The proposed method is significantly faster than the `simple` method using libsvm and competitive with that using liblinear. The time complexity of `simple` method which solves the standard SVM dual has been empirically shown to be $O(n^{2.1})$ [25]. The proposed optimization approach iteratively works on the block-wise subset into which the whole training set is decomposed (Sect. 2.2). The subset is regarded as the working set whose size is an important factor for fast computing QP [18]. In the `proposed` method, it is advantageous to inherently define the subset, *i.e.*, the working set, of adequate size according to the prior. Thus, roughly speaking, the time complexity of the `proposed` method results in $O(M\frac{n^{2.1}}{M^{2.1}}) = O(\frac{n^{2.1}}{M^{1.1}})$. Besides, the technique to skip the trivial subset (Sect. 2.3) empirically contributes to further reduce the computational cost. The computation time essentially depends on the (resultant) number of support vectors (SVs); Fig. 4b shows the number of support vectors produced by those two methods. The `proposed` method provides a smaller number of support vectors, which significantly contributes to reduce the computation time. As a result, the proposed optimization approach works quite well from the viewpoint of computation time. This result shows the favorable scalability of the `proposed` method, especially compared to the standard `simple` method.

3.3 Trivial Biases

As described in Sect. 2.3, in the proposed method, it is necessary to compute the biases on the trivial priors rather heuristically, since they are not theoretically determined. We presented three ways, *tight*, *mild* and *extreme* ones, for computing those biases (Sect. 2.3), and the classification performances are compared

in Table 2. We can not find significant differences in performance across those approaches; the *extreme* way provides slightly better performance and in the following experiments we apply this method for computing biases on the trivial priors.

Table 2. Performance comparison (%) on the ways of computing biases on trivial priors.

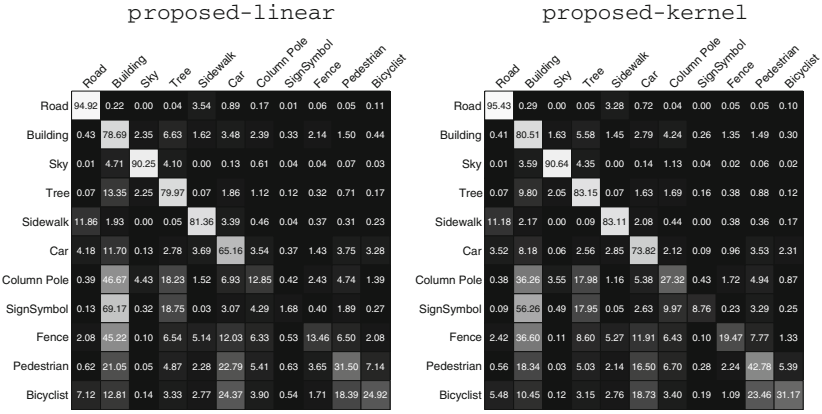
Tight	Mild	Extreme
52.22	52.19	52.25

Table 3. Performance comparison.

(a) Classification accuracy (%)

	simple	full-connected	proposed-linear	proposed-kernel
road	93.10	93.80	**94.92**	95.43
building	75.90	72.96	**78.70**	80.51
sky	**90.52**	82.21	90.25	90.64
tree	70.49	77.59	**79.95**	83.15
sidewalk	77.06	78.43	**81.36**	83.11
car	53.84	58.64	**65.16**	73.82
column pole	9.53	**16.15**	12.85	27.32
sign symbol	**1.73**	1.62	1.70	8.76
fence	5.23	11.09	**13.48**	19.47
pedestrian	17.26	30.69	**31.52**	42.78
bicyclist	17.09	18.49	**24.88**	31.17
avg.	46.52	49.24	**52.25**	57.83

(b) Confusion matrix (%)



3.4 Classification Performance

We finally compared the classification performance of the three methods, `simple`, `full-connected` and `proposed` (listed in Table 1); for reference, we also apply the kernel-based extension of the proposed method by using Gaussian kernel $\mathsf{k}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\gamma})$ where γ is mean of the pair-wise distances. Table 3 shows the overall performance, demonstrating that the `proposed` method outperforms the others. It should be noted that the `full-connected` method individually applies the classifier specific to the prior $p \in \{1, \cdots, P\}$, requiring a plenty of memory storage and consequently taking large classification time due to loading the enormous memory. The `proposed` method renders as fast classification as the `simple` method since it enlarges only the bias. By discriminatively optimizing the biases for respective priors, the performance is significantly improved

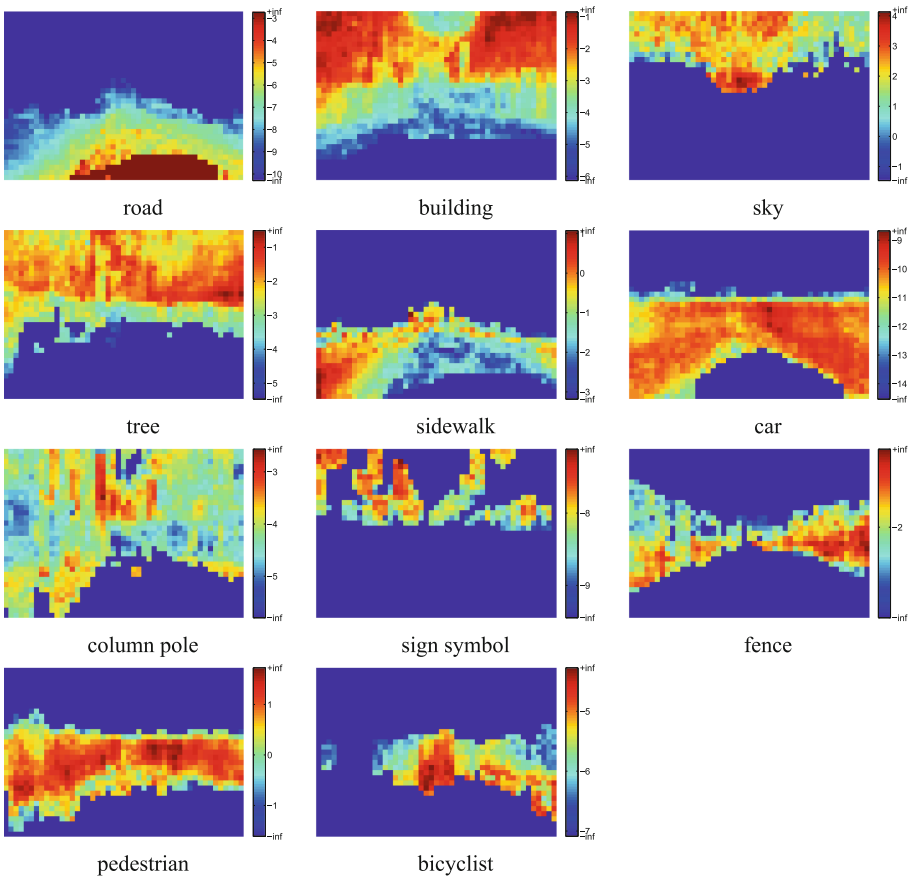


Fig. 5. Maps of the biases learnt by the `proposed-linear` method. The significance of the biases are shown by using pseudo colors from (dark) blue to (dark) red. This figure is best viewed in color (Color figure online).

in comparison to the `simple` method; the improvement is especially found at the categories of *car, pedestrian* and *bicyclist* that are composed of patch parts similar to other categories but are associated with the distinct prior positions.

The kernel-based method (`proposed-kernel`) further improves the performance on the foreground object categories, such as *column pole* and *pedestrian*. Those foreground objects exhibit large variations in appearance due to viewpoint changes and within-class variations themselves, and the kernel-based method produces more discriminative feature mapping function compared to the linear method.

Finally, we show in Fig. 5 the biases learnt by the `proposed` method; the biases $\{b^{[p]}\}_p$ are folded into the form of image frame according to the x-y positions. These maps of the biases reflect the *prior* probability over the locations where the target category appears. These seem quite reasonable from the viewpoint of the traffic rules that cars obeys; since the CamVid dataset is collected at the Cambridge city [21], in this case, the traffic rules are of the United Kingdom. The *pedestrian* probably walks on the *sidewalk* mainly shown in the left side. The oncoming *car* runs on the right-hand road, and the row of the *building* is found on the roadside. These biases are adaptively learnt from the CamVid dataset and they would be different if we use other datasets collected under different traffic rules.

4 Conclusions

We have proposed a method to discriminatively learn the prior biases in the classification. In the proposed method, for improving the classification performance, all samples are utilized to train the classifier and the input sample is adequately classified based on the prior information via the learnt biases. The proposed method is formulated in the maximum-margin framework, resulting in the optimization problem of the quadratic programming form similarly to SVM. We also presented a computationally efficient approach to optimize the resultant quadratic programming along the line of sequential minimal optimization. The experimental results on the patch labeling in the on-board camera images demonstrated that the proposed method is superior in terms of classification accuracy and the computation cost. In particular, the proposed classifier operates as fast as the standard (linear) classifier, and besides the computation time for training the classifier is even faster than the SVM of the same size.

References

1. Jiang, T., Jurie, F., Schmid, C.: Learning shape prior models for object matching. In: CVPR 2009, the 22nd IEEE Conference on Computer Vision and Pattern Recognition, pp. 848–855 (2009)
2. El-Baz, A., Gimel'farb, G.: Robust image segmentation using learned priors. In: ICCV 2009, the 12nd International Conference on Computer Vision, pp. 857–864 (2009)

3. Cremers, D., Grady, L.: Statistical priors for efficient combinatorial optimization via graph cuts. In: Pinz, A., Leonardis, A. (eds.) ECCV 2006. LNCS, vol. 3953, pp. 263–274. Springer, Heidelberg (2006)
4. Jie, L., Tommasi, T., Caputo, B.: Multiclass transfer learning from unconstrained priors. In: ICCV 2011, the 13th International Conference on Computer Vision, pp. 1863–1870 (2011)
5. Yuan, C., Hu, W., Tian, G., Yang, S., Wang, H.: Multi-task sparse learning with beta process prior for action recognition. In: CVPR 2013, the 26th IEEE Conference on Computer Vision and Pattern Recognition, pp. 423–430 (2013)
6. Wang, C., Liao, X., Carin, L., Dunson, D.: Classification with incomplete data using dirichlet process priors. J. Mach. Learn. Res. **11**, 3269–3311 (2010)
7. Kapoor, A., Hua, G., Akbarzadeh, A., Baker, S.: Which faces to tag: adding prior constraints into active learning. In: ICCV 2009, the 12th International Conference on Computer Vision, pp. 1058–1065 (2009)
8. Ghosh, J., Ramamoorthi, R.: Bayesian Nonparametrics. Springer, Berlin (2003)
9. Poggio, T., Mukherjee, S., Rifkin, R., Rakhlin, A., Verri, A.: b. Technical report CBCL Paper#198/AI Memo #2001-011, Massachusetts Institute of Technology, Cambridge (2001)
10. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, New York (1995)
11. Van Gestel, T., Suykens, J., Lanckriet, G., Lambrechts, A., De Moor, B., Vandewalle, J.: Bayesian framework for least squares support vector machine classifiers, gaussian processes and kernel fisher discriminant analysis. Neural Comput. **15**, 1115–1148 (2002)
12. Gao, T., Stark, M., Koller, D.: What makes a good detector? – structured priors for learning from few examples. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part V. LNCS, vol. 7576, pp. 354–367. Springer, Heidelberg (2012)
13. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Berlin (2006)
14. Smola, A.J., Bartlett, P., Schölkopf, B., Schuurmans, D.: Advances in Large-Margin Classifiers. MIT Press, Cambridge (2000)
15. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
16. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning, pp. 185–208. MIT Press, Cambridge (1999)
17. Hsieh, C.J., Chang, K.W., Lin, C.J., Keerthi, S.S., Sundararajan, S.: A dual coordinate descent method for large-scale linear svm. In: ICML 2008, the 25th International Conference on Machine Learning, pp. 408–415 (2008)
18. Fan, R.E., Chen, P.H., Lin, C.J.: Working set selection using second order information for training support vector machines. J. Mach. Learn. Res. **6**, 1889–1918 (2005)
19. Kan, M., Shan, S., Zhang, H., Lao, S.: Multi-view discriminant analysis. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part I. LNCS, vol. 7572, pp. 808–821. Springer, Heidelberg (2012)
20. Sharma, A., Jacobs, D.: Bypassing synthesis: pls for face recognition with pose, low-resolution and sketch. In: CVPR 2011, the 24th IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 593–600 (2011)
21. Fauqueur, J., Brostow, G.J., Shotton, J., Cipolla, R.: Segmentation and recognition using structure from motion point clouds. In: Torr, P., Forsyth, D., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 44–57. Springer, Heidelberg (2008)

22. Kobayashi, T., Otsu, N.: Image feature extraction using gradient local auto-correlations. In: Torr, P., Forsyth, D., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 346–358. Springer, Heidelberg (2008)
23. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001). Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
24. Fan, R., Chang, K., Hsieh, C., Wang, X., Lin, C.: Liblinear: a library for large linear classification. J. Mach. Learn. Res. **9**, 1871–1874 (2008). Software available at http://www.csie.ntu.edu.tw/~cjlin/liblinear
25. Joachims, T.: Making large-scale svm learning practical. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning, pp. 169–184. MIT Press, Cambridge (1999)