17th *Computer Vision Winter Workshop*
Matej Kristan, Rok Mandeljc, Luka Čehovin (eds.)
Mala Nedelja, Slovenia, February 1-3, 2012

Efficient Algorithm for Regularized Risk Minimization

Michal Uřičář and Vojtěch Franc
Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
[uricamic, xfrancv]@cmp.felk.cvut.cz

Abstract. *Many machine learning algorithms lead to solving a convex regularized risk minimization problem. Despite its convexity the problem is often very demanding in practice due to a high number of variables or a complex objective function. The Bundle Method for Risk Minimization (BMRM) is a recently proposed method for minimizing a generic regularized risk. Unlike the approximative methods, the BMRM algorithm comes with convergence guarantees but it is often too slow in practice. We propose a modified variant of the BMRM algorithm which decomposes the objective function into several parts and approximates each part by a separate cutting plane model instead of a single cutting plane model used in the original BMRM. The finer approximation of the objective function can significantly decrease the number of iterations at the expense of higher memory requirements. A preliminary experimental comparison shows promising results.*

1. Introduction

Learning predictors from data is a standard machine learning task. A large number of such learning tasks are translated into a convex regularized risk minimization problem

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}\in\mathbb{R}^n} F(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + R(\boldsymbol{w})\,. \quad (1)$$

The objective $F\colon \mathbb{R}^n \to \mathbb{R}$, called regularized risk, is a sum of the quadratic regularization term [1] and a convex empirical risk $R\colon \mathbb{R}^n \to \mathbb{R}$. The scalar $\lambda > 0$ is pre-defined regularization constant and $\boldsymbol{w} \in \mathbb{R}^n$ is a parameters vector to be learned. The quadratic regularization terms serves as a mean to constraint the space of solutions in order to improve generalization. The empirical risk R evaluates a match between the parameters $\boldsymbol{w}$ and given set of training examples. The empirical risk R is most often defined as a sum of loss functions evaluated for all training examples.

A list of learning algorithms which are in their core solvers of (1) is long, e.g. SVM classification and regression, logistic regression, maximal margin structured output classification, SVM for multi-variate performance measure, novelty detection, learning of Gaussian processes to name the most popular.

Using off-the-shelf solvers to tackle practical instances of the minimization problem (1) is not feasible. The difficulty stems from the risk term R which is often complex non-differentiable function whose evaluation is expensive. A huge afford has been put to development of specialized algorithms tailored to particular instances of (1). Recently a generic solver, named Bundle Method for Risk Minimization (BMRM), has been proposed in [3]. The BMRM algorithm replaces the risk R by its cutting plane approximation which is build in an iterative fashion. The BMRM algorithm is highly modular and was proved to converge in $\mathcal{O}(\frac{1}{\epsilon})$ to an ϵ-optimal solution [3].

In this paper, we propose a modified variant of the BMRM, called P-BMRM, which uses $P > 1$ cutting plane models to approximate the risk R instead of a single one as the original BMRM. Higher number of cutting plane models allows to obtained finer approximation of the risk which in turn can decrease the number of iterations. Importantly, updating P cutting plane models, which needs to be done in each iteration, has exactly the same computational complexity as updating a single cutting plane

[1] We assume the most frequently used quadratic regularizer for simplicity but the proposed method can be used for any convex regularizer, e.g. L_1-norm.

model of the original BMRM. On the other hand, the P-BMRM requires P times more memory for storing the cutting planes as compared to the original BMRM. The P-BMRM algorithm can be efficiently parallelized by computing and storing each of the P cutting plane models on a separate computer.

We experimentally evaluate the P-BMRM on two maximal margin classifier learning problems using real-life data. We show that P-BMRM algorithm significantly reduces the number of iterations if compared to the standard BMRM algorithm [2].

The paper is organized as follows. Section 2 gives an overview of the original BMRM. Section 3 provides a brief description of the online SGD algorithm. The proposed P-BMRM algorithm is given in Section 4. Section 5 reports empirical results and Section 6 concludes the paper.

2. Bundle Method for Regularized Risk Minimization

BMRM [3] is a generic optimization method for solving the regularized risk minimization problem (1) What makes the optimization difficult is the risk $R(\boldsymbol{w})$ being the part of the objective of the problem (1). The risk $R(\boldsymbol{w})$ is typically non-differentiable function whose evaluation is expensive. The core idea behind the BMRM is to replace the complex risk $R(\boldsymbol{w})$ by its simpler cutting plane model $R_t(\boldsymbol{w})$. The cutting plane model

$$R_t(\boldsymbol{w}) = \max_{i=0,\ldots,t-1} \left[R(\boldsymbol{w_i}) + \langle R'(\boldsymbol{w_i}), \boldsymbol{w} - \boldsymbol{w_i}\rangle\right] \tag{2}$$

is defined as a point wise maximum over t linear under-estimators, so called *cutting planes*. $R'(\boldsymbol{w}_i) \in \partial_{\boldsymbol{w}_i} R(\boldsymbol{w}_i)$ is an arbitrary sub-gradient of R at point $\boldsymbol{w}_i$. The i-th cutting plane $R(\boldsymbol{w_i})+\langle R'(\boldsymbol{w_i}), \boldsymbol{w}-\boldsymbol{w_i}\rangle$ is a linear lower bound of the risk function $R(\boldsymbol{w})$ which is tight at the point $\boldsymbol{w}_i$. The cutting plane approximation is illustrated in Figure 1. With this approximation, one can replace the original problem (1) by a simpler *reduced problem*

$$\boldsymbol{w}_t = \arg\min_{\boldsymbol{w}\in\mathbb{R}^n} F_t(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + R_t(\boldsymbol{w})\,. \tag{3}$$

The reduce problem objective $F_t(\boldsymbol{w})$ is obtained from the original problem objective $F(\boldsymbol{w})$ by replacing the risk $R(\boldsymbol{w})$ with its cutting plane approximation $R_t(\boldsymbol{w})$ while the regularizer is kept unchanged.

[2]Note, that the StructSVM, which is the state-of-the-art solver for structured output SVM solver, is just a special instance of the BMRM.

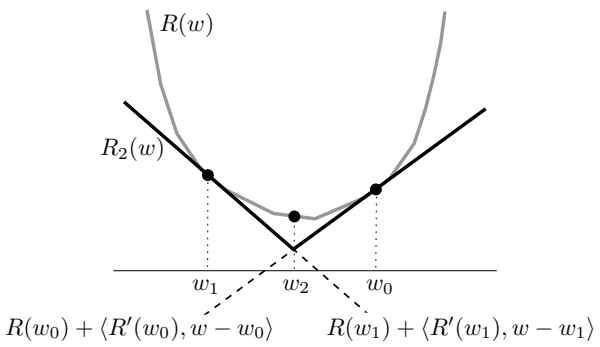


Figure 1. A convex function $R(\boldsymbol{w})$ can be approximated by a collection of linear under-estimators (cutting planes).

Algorithm 1 BMRM

Require: ϵ, a function pointer $R(\boldsymbol{w})$, a function pointer $R'(\boldsymbol{w})$

1: **Initialization:** $\boldsymbol{w} \leftarrow \boldsymbol{0}, t \leftarrow 0$
2: **repeat**
3: $t \leftarrow t + 1$
4: Compute $R(\boldsymbol{w}_t), R'(\boldsymbol{w}_t)$
5: Update $R_t(\boldsymbol{w}_t)$
6: Solve $\boldsymbol{w}_t \leftarrow \arg\min F_t(\boldsymbol{w})$
7: $\epsilon_t \leftarrow F(\boldsymbol{w}_t) - F_t(\boldsymbol{w}_t)$
8: **until** $\epsilon_t \leq \epsilon$

Algorithm 1 shows a pseudo-code of the standard BMRM. It can be proved [3], that for arbitrary $\epsilon > 0$, the BMRM returns a solution vector $\boldsymbol{w}_t$ satisfying $F(\boldsymbol{w}_t) \leq F(\boldsymbol{w}^*) + \epsilon$ after at most $\mathcal{O}(\frac{1}{\epsilon})$ iterations.

The BMRM algorithm typically requires few hundred iterations to converge. The evaluation of the risk $R(\boldsymbol{w})$ and its sub-gradient $R'(\boldsymbol{w})$ required in step 4 is very often the most expensive part of the algorithm. If the risk $R(\boldsymbol{w})$ is additively decomposable (which is the case in most situations) the step 4 can be efficiently parallelized. The parallel BMRM [3] distributes the computations over P threads (or computers) but the number of iterations remains the same. In this paper we propose a modified BMRM algorithm which can be not only efficiently parallelized but which also decreases the number of iterations.

3. Stochastic Gradient Descent

In this section we describe the Stochastic Gradient Descent (SGD) algorithm which is a popular approximate solver of the problem (1). We use the SGD algorithm as a competing method for the proposed solvers. The SGD is a simple on-line algorithm which often reaches sufficiently good solution if its learning rate (i.e. the free parameter of the

SGD) is chosen properly. Unlike the precise methods, the SGD does not provide any certificate of optimality for the computed solution and hence it lacks a good stopping condition.

The SGD assumes that the risk R is additively decomposable, i.e. $R(\boldsymbol{w}) = \sum_{i=1}^{m} r_i(\boldsymbol{w})$. For example, $r_i(\boldsymbol{w})$ can be a loss for a predictor with parameters $\boldsymbol{w}$ incurred on i-th example coming from a training set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$. Starting from an initial guess $\boldsymbol{w}_0 \in \mathbb{R}^n$, the SGD applies the following update rule

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha_t \boldsymbol{g}_t\,, \quad \text{where} \tag{4}$$

$$\boldsymbol{g}_t = \lambda \boldsymbol{w}_t + \boldsymbol{r}'_t(\boldsymbol{w_t})\,, \tag{5}$$

t is the iteration counter, α_t is a prescribed learning rate and $\boldsymbol{r}'_t(\boldsymbol{w_t})$ is a sub-gradient of $r_t(\boldsymbol{w})$ at point $\boldsymbol{w}_t$. During one pass through the training examples the SGD update rule (4) is applied m times which is approximately as expensive as one iteration of the BMRM algorithm. In practice, a fixed number of passes through the training examples is used as a "stopping condition" of the SGD algorithm. The convergence speed crucially depends on setting the learning rate α_t. Following [1], we set the learning rate to

$$\alpha_t = \frac{\lambda^{-1}}{t_0 + t} \tag{6}$$

where t_0 is a positive constant its value is tuned on a small portion (e.g. 10%) of training examples.

4. Proposed P-BMRM

In this section we propose a modified BMRM which uses $P > 1$ cutting plane models to approximate the risk $R(\boldsymbol{w})$ instead of the single cutting plane model as in the standard BMRM. We call the proposed method P-BMRM.

We assume that the risk can be decomposed into $P > 1$ terms

$$R(\boldsymbol{w}) = \sum_{p=1}^{P} R(\boldsymbol{w}, p). \tag{7}$$

We propose to approximate each of the P terms by its own cutting plane model

$$R_t(\boldsymbol{w}, p) = \max_{i=0,\ldots,t-1} \left[\langle a_{i,p}, \boldsymbol{w}\rangle + b_{i,p}\right], \ \forall p, \tag{8}$$

where

$$a_{i,p} = R'(\boldsymbol{w}_i, p), \tag{9}$$

$$b_{i,p} = R(\boldsymbol{w}_i, p) - \langle \boldsymbol{w}_i, R'(\boldsymbol{w}_i, p)\rangle. \tag{10}$$

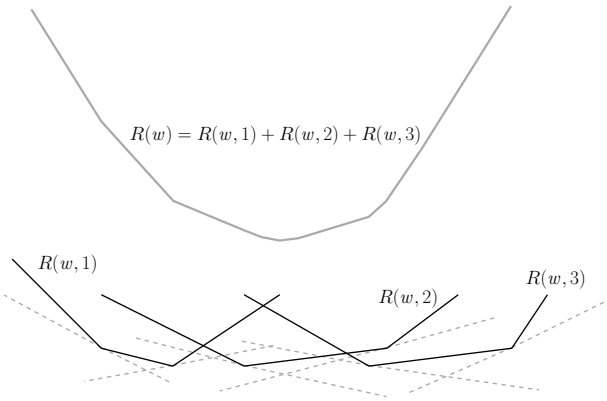


Figure 2. The figure illustrates how is the risk $R(\boldsymbol{w})$ (upper figure) decomposed into a sum of functions $R(\boldsymbol{w}, p)$, $p = 1, \ldots, P$, (lower figure) each of which is approximated by its own cutting planes (dashed lines).

Algorithm 2 P-BMRM

Require: ϵ, routines computing $R(\boldsymbol{w}, p)$ and $R'(\boldsymbol{w}, p)$

1: **Initialization:** $\boldsymbol{w} \leftarrow \boldsymbol{0}, t \leftarrow 0$
2: **repeat**
3: $\quad t \leftarrow t + 1$
4: $\quad$ Compute $R(\boldsymbol{w}_t, p), R'(\boldsymbol{w}_t, p), p = 1, \ldots, P$
5: $\quad$ Update $R_t(\boldsymbol{w}_t, p), \quad p = 1, \ldots, P$
6: $\quad$ Solve $\boldsymbol{w}_t \leftarrow \arg\min F_t(\boldsymbol{w})$
7: $\quad \epsilon_t \leftarrow F(\boldsymbol{w}_t) - F_t(\boldsymbol{w}_t)$
8: **until** $\epsilon_t \leq \epsilon$

The function $R(\boldsymbol{w}, p)$ represents the value of p-th term evaluated at point $\boldsymbol{w}$ and $R'(\boldsymbol{w}, p)$ denotes a sub-gradient of $R(\boldsymbol{w}, p)$. Figure 2 illustrates the idea.

The cutting plane model of the risk $R(\boldsymbol{w})$ is then

$$R_t(\boldsymbol{w}) = \sum_{p=1}^{P} R_t(\boldsymbol{w}, p). \tag{11}$$

We denote (11) as P-cutting plane model. Note that for $P = 1$, the P-cutting plane model (11) reduces to the original model (2). It is seen that the P-cutting plane model (11) preserves the crucial properties of the original model, i.e., $R_t(\boldsymbol{w})$ is a lower bound of $R(\boldsymbol{w})$ which is tight at the points $\boldsymbol{w}_i$, $i = 0, \ldots, t-1$.

Now, we can derive the P-BMRM algorithm just by substituting the P-cutting plane model for the original cutting plane model (2) in the definition of the reduced problem (3). Algorithm 2 shows a pseudo code of the proposed P-BMRM.

The reduced problem (3) of the P-BMRM algorithm required in step 6 can be equivalently ex-

pressed as a convex quadratic program (QP)

$$\boldsymbol{w}_t = \arg\min_{\boldsymbol{w}\in\mathbb{R}^N} \left[\frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \sum_{p=1}^{P} \xi_p \right] \tag{12}$$

subject to

$$\langle a_{i,p}, \boldsymbol{w}\rangle + b_{i,p} \leq \xi_p, \quad i = 0, \ldots, t-1,\ p = 1, \ldots, P.$$

If the dimensionality n of the parameter vector $\boldsymbol{w} \in \mathbb{R}^n$ is large it is better to solve (12) in its dual form

$$\alpha_t = \arg\max_{\alpha\in\mathbb{R}^t} \left[\sum_{i=0}^{t-1}\sum_{p=1}^{P} \alpha_{i,p} b_{i,p} - \frac{1}{2\lambda}\|\sum_{i=0}^{t-1}\sum_{p=1}^{P} \alpha_{i,p} a_{i,p}\|^2 \right] \tag{13}$$

subject to

$$\sum_{i=0}^{t-1} \alpha_{i,p} = 1, \quad p = 1, \ldots, P \tag{14}$$

$$\alpha_{i,p} \geq 0, \quad i = 0, \ldots, t-1, \quad \forall p \tag{15}$$

The transformation of the dual solution to the primal one is given by the following formula

$$\boldsymbol{w}_t^* = -\frac{1}{\lambda}\sum_{i=0}^{t-1}\sum_{p=1}^{P} \alpha_{i,p} a_{i,p}. \tag{16}$$

The number of the dual variables equals to $P \cdot t$ which is typically small as the algorithm converges after a few iterations.

Let us briefly compare the standard BMRM and the proposed P-BMRM algorithm. The standard BMRM adds a single linear term to the cutting plane model in each iteration, requiring one pass over the whole training set. In contrast the proposed P-BMRM algorithm adds a single linear term to each P partial cutting plane models still requiring only one pass over the training set and the same number of computations. As a result, the P-BMRM algorithm produces much finer approximation than the standard BMRM algorithm which leads to less iterations, however, at the price of requiring P times more memory. The proposed P-BMRM requires $\mathcal{O}(T \cdot P \cdot n)$ memory where T is the overall number of iterations and n is the dimensionality of $\boldsymbol{w}$.

Similarly to the original BMRM, the P-BMRM remains highly modular. In order to be applied to a particular instance of (1) it only requires two subroutines: one for evaluation of the risk $R(\boldsymbol{w}, p)$ and one computing its sub-gradient $R'(\boldsymbol{w}, p)$.

5. Experiments

In this section, we compare the proposed P-BMRM algorithm to the standard BMRM. We also compare to the SGD algorithm being a popular representative of the approximative methods. As benchmark problems we use two different instances of the maximum margin classifier learning which both lead to solving (1). In particular, we consider problem learning a face landmark detector and an optical character recognition classifier described in section 5.2 and 5.3, respectively.

5.1. Maximal margin classification

Assume we are given a set of training examples $\{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})^m$ assumed to be i.i.d. from some unknown p.d.f. $p(x, y)$. The goal is to learn a classifier $h\colon \mathcal{X} \to \mathcal{Y}$ which minimizes the expected risk $E_{p(x,y)}[\ell(y, h(x))]$ for some given loss function $\ell\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$.

Let us assume that the optimal classifier can be approximated well by a linear classifier

$$\hat{y} = h(x; \boldsymbol{w}) = \arg\max_{y\in\mathcal{Y}} \langle \boldsymbol{w}, \boldsymbol{\Psi}(x, y)\rangle \tag{17}$$

where $\boldsymbol{w} \in \mathbb{R}^n$ is a parameter vector and $\Psi\colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^n$ is a known map from the input-output space onto the parameter space. The maximum margin approach to learning the parameters $\boldsymbol{w}$ of the classifier (17) from the training examples leads to the risk minimization problem (1) with the risk set to

$$R(\boldsymbol{w}) = \sum_{i=1}^{m} r_i(\boldsymbol{w}) \ , \tag{18}$$

where

$$r_i(\boldsymbol{w}) = \max_{y\in\mathcal{Y}} \left[\ell(y_i, y) + \langle \boldsymbol{w}, \boldsymbol{\Psi}(x_i, y) - \boldsymbol{\Psi}(x_i, y_i)\rangle\right]. \tag{19}$$

It can be shown [4] that for a broad class of loss functions $r_i(\boldsymbol{w})$ is a convex upper bound of the loss $\ell(y_i, h(x_i; \boldsymbol{w}))$. In particular, the statement holds if $\ell(y, y') \geq 0, \forall y, y'$ and $\ell(y, y') = 0$ iff $y = y'$.

This general formulation has many incarnations which differ in the definition of the input space $\mathcal{X}$, the output space $\mathcal{Y}$, the loss function $\ell(y, y')$ and the map $\boldsymbol{\Psi}$. In order to use BMRM algorithm (as well as P-BMRM and SGD) for learning of the particular instances of the classifier (17) one needs to evaluate $r_i(\boldsymbol{w})$ and its sub-gradient $r_i'(\boldsymbol{w})$. By Danskin's theorem, the sub-gradient of $r_i(\boldsymbol{w})$ reads

$$r_i'(\boldsymbol{w}) = \boldsymbol{\Psi}(x_i, \hat{y}_i) - \boldsymbol{\Psi}(x_i, y_i) \tag{20}$$

where

$$\hat{y}_i = \arg\max_{y \in \mathcal{Y}} \left[\ell(y_i, y) + \left\langle \boldsymbol{w}, \boldsymbol{\Psi}(x_i, y) \right\rangle \right]. \quad (21)$$

In the next sections we described two particular instances of the maximum margin classifier we use as the benchmarks to evaluate the proposed solver.

5.2. Benchmark 1: Facial landmark detection

As the first benchmark problem we consider learning of a face landmark detector from examples. We follow the approach of [5] where the landmark detection is posed as an instance of the structured output SVM classification, i.e. the detector is an instance of the linear classifier (17). In this case the input $x \in \mathcal{X}$ is an image 40×40 pixels which contains a face. The classifier outputs $y = (y^1, \ldots, y^L) \in \mathcal{Y} = \mathcal{N}^{2 \times L}$ which is a set of 2D coordinates of L landmarks like corners of the eyes, corners of the mouth, tip of the nose and the center of the face. The scoring function $\langle \boldsymbol{w}, \boldsymbol{\Psi}(x, y) \rangle$ of the classifier (17) is composed of the appearance model and the deformation cost. The appearance model evaluates a match between the input image x and the landmark templates put at positions y. The deformation cost evaluates likeliness of the particular landmark configuration y and this cost decomposes to a set of pair wise terms defined over edges of an acyclic graph. The map $\boldsymbol{\Psi}(x, y)$ is as a column-wise concatenation of local feature descriptors of individual landmarks and parameters of the deformation cost. Evaluation of the classifier (17) leads to solving an instance of the dynamic programing. As the loss function we use the normalized mean deviation between the ground truth landmark positions y and the estimated positions y' defined as

$$\ell(y, y') = \kappa(y) \frac{1}{L} \sum_{j=0}^{L-1} \| y^j - y'^j \| , \quad (22)$$

where $\kappa(s)$ is a normalization constant ensuring that the loss ℓ is scale invariant.

We trained the landmark detector from a set of $m = 7,000$ images with manually annotated landmark positions. In our experiment the number of landmarks was $L = 8$ and the number of parameters $\boldsymbol{w} \in \mathbb{R}^n$ to learn was $n = 232,476$.

5.3. Benchmark 2: Optical character recognition

As the second benchmark we consider the optical character recognition (OCR) problem. We use the standard MNIST database [2] composed of labeled examples of handwritten numerals. In this case, the input x is gray scale image 28×28 pixels. The output y is the digit name, i.e. $y \in \mathcal{Y} = \{0, \ldots, 9\}$. We model each class by a single template image $w_y \in \mathbb{R}^{28 \times 28}$, $y \in \mathcal{Y}$. As the scoring function of the classifier (17) we use

$$\langle \boldsymbol{w}, \boldsymbol{\Psi}(x, y) \rangle = \langle x, w_y \rangle . \quad (23)$$

The parameter vector $\boldsymbol{w} \in \mathbb{R}^n$ has dimension $n = 7,840$ resulting from a column-wise concatenation of 10 templates w_y, $y \in \mathcal{Y}$, represented themselves as columns vectors. We use the standard classification 0/1-loss defined to be $\ell(y, y') = 1$ for $y \neq y'$ and $\ell(y, y') = 0$ otherwise. With these definitions the classifier (17) becomes an instance of a linear multi-class SVM classifier.

The linear classifier is certainly not the best model for handwritten OCR. We use this model because the risk minimization leads to a relatively low dimensional optimization problem ($n = 7,840$) allowing us to test the proposed algorithm with a high P even on a computer with small memory. We train on all $m = 60,000$ training examples and we use the test part of the database for validation and testing.

5.4. Results

We compare the original BMRM, the proposed P-BMRM and the SGD algorithms in terms of the number of iterations needed to achieve the ϵ-precise solution of the problem (1). There are two reasons justifying the use of the number of iterations instead of the wall clock time. First, the per-iteration computational complexity of all the tested algorithms is similar (in the case of the SGD algorithm we consider one pass through the example as one iteration) if the computations are dominated by evaluation of the risk and its sub-gradient. This is often the case in the structured output classification or in learning from a huge number of examples. Second, the number of iterations does not depend on the used computer and the particular implementation of the algorithms.

We run the BMRM and the P-BMRM algorithm on the two instances of the problem (1) for different values of the regularization constants λ. This simulates the real situation when the optimal λ is tuned in the model selection stage.

We stop the algorithms when the relative distance to the optimal solution drops below 0.01, i.e. the

stopping condition reads

$$\frac{F(\boldsymbol{w}_t) - F_t(\boldsymbol{w}_t)}{F(\boldsymbol{w}_t)} \leq 0.01 \,. \tag{24}$$

The stopping condition (24) cannot be used in the SGD algorithm because it does not proved the lower bound on the optimal value. To allow fair comparison, we first ran the BMRM algorithm and then we used the value of the objective function in the last iteration as a stopping criterion for SGD. I.e. we iterate SGD until the value of the objective function $F(\boldsymbol{w})$ gets below to the value returned from the BMRM algorithm.

Tables 1 and 2 show the number of iterations required to achieve the ϵ-optimal solution for the two benchmark problems. Each row shows results for a particular algorithm. The first row, P=1 BMRM, corresponds to the standard BRMR algorithm which is our baseline. The columns show the required number of iterations and the speedup of the parallelized P-BMRM algorithm for different values of λ. The parallelized P-BMRM means that each of the P cutting plane models is computed on a separate computer. Note that the speedup of the standard parallel BMRM algorithm [3], which evenly decomposes computation of the risk and the sub-gradient on P computers, equals to P. I.e. the speedup of the parallel P-BMRM algorithm is higher than the speedup of the standard parallel BMRM due to the lower number of iterations. The last row of the table 1 shows the number of iterations and the speedup obtained for the SGD algorithm. The symbol "–" means that the SGD algorithm has not converged after two days.

The results show two pleasant observations. First, the number of iterations decreases as the number of cutting plane models P grows. This is an expected result because higher P implies finer approximation of the objective. Second, the P-BMRM algorithm is more effective for low values of λ, i.e. for the most time consuming instances of the problem (1). For example, the P=64-BMRM requires 11 times less iterations compared to the standard BMRM when used to learn the OCR classifier with $\lambda = 10$. Using the parallel P-BMRM implies speedup of factor of 700.

Figures 3 and 5 show the convergence curves of the tested algorithms for different values of λ. The x-axis corresponds to the number of iterations and the y-axis is the relative distance to the optimal solution. The effect of using P-BMRM with higher number of P is clearly demonstrated.

In Figure 4(a) we also show value of the objective function $F(\boldsymbol{w})$ and the value of the validation risk for the BMRM algorithm and the SGD algorithm. The graphs correspond to the landmark detection problem and the optimal setting of λ which was 100. It is seen that the SGD reaches relatively good precision of objective function after a few iterations but then it stalls unlike the BMRM which reaches the ϵ-precise solution. It is seen that more precise solution of the objective $F(\boldsymbol{w})$ is translated to lower validation risk. In particular, the precise BMRM algorithm outperformed the approximative SGD by significant 1.5% in terms of the detection accuracy.

6. Conclusions

We have proposed an efficient algorithm, called P-BMRM, for solving the regularized risk minimization problem (1). The P-BMRM improves the standard BMRM algorithm by using $P > 1$ cutting plane models to approximate the risk R instead of a single cutting plane model as in the original BMRM. Higher number of cutting plane models allows to obtain finer approximation of the risk which in turn decreases the number of iterations needed to achieve the ϵ-optimal solution. Importantly, updating P cutting plane models, which needs to be done in each iteration, has exactly the same computational complexity as updating a single cutting plane model of the original BMRM. On the other hand, the P-BMRM requires P times more memory for storing the cutting planes as compared to the original BMRM. The P-BMRM algorithm can be efficiently parallelized by computing and storing each of the P cutting plane models on a separate computer.

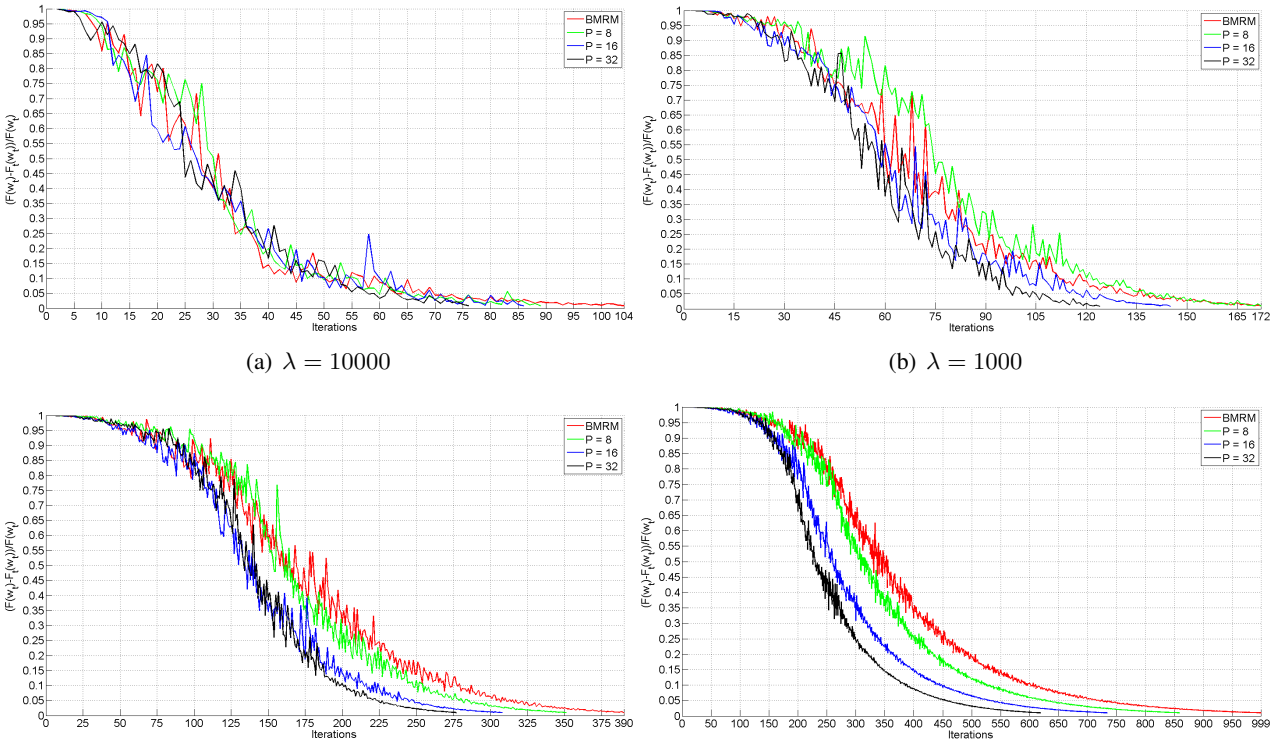
The experimental evaluation on two real-life problems show that the proposed P-BMRM algorithm requires significantly less iterations compared to the standard BMRM. The P-BMRM algorithm with P=64 decrease the number of iterations by more than an order of magnitude compared to the standard BMRM. The speedup obtained by P-BMRM algorithm is more significant for the problems with low values of the regularization constant which are the most time consuming ones.

Our current implementation of the P-BMRM algorithm runs on a single computer only. In the future work we will focus on an efficient implementation of the proposed algorithm using distributed computers as well as we will conduct more thorough comparison.

Facial landmark detection

	$\lambda = 10000$		$\lambda = 1000$		$\lambda = 100$		$\lambda = 10$	
	#iter	speedup	#iter	speedup	#iter	speedup	#iter	speedup
P = 1 BMRM	104	1	172	1	390	1	999	1
P = 8 BMRM	88	9.5	171	8.0	350	8.9	858	9.3
P = 16 BMRM	85	19.6	144	19.1	307	20.3	733	21.8
P = 32 BMRM	75	44.4	123	44.7	276	45.2	618	51.7
SGD	1	104	1	171	—	—	—	—

Table 1. The table shows the number of iterations needed to reach the ϵ-optimal solution and the speed up of the parallelized version of the algorithm. Each row corresponds to different setting of the number of cutting plane models. The last row shows results for the SGD algorithm. The symbol "—" means that the SGD has not converged.



(a) $\lambda = 10000$ (b) $\lambda = 1000$

(c) $\lambda = 100$ (d) $\lambda = 10$

Figure 3. Convergence curves of the standard BMRM (i.e.P=1-BMRM) and P-BMRM algorithm applied with different setting of the regularization constant λ to the face landmark detection problem. The curves show the relative distance to the optimal solution as the function of the number of iterations.

Acknowledgements

The authors were supported by EC projects FP7-ICT-247525 HUMAVIPS and PERG04-GA-2008-239455 SEMISOL.

References

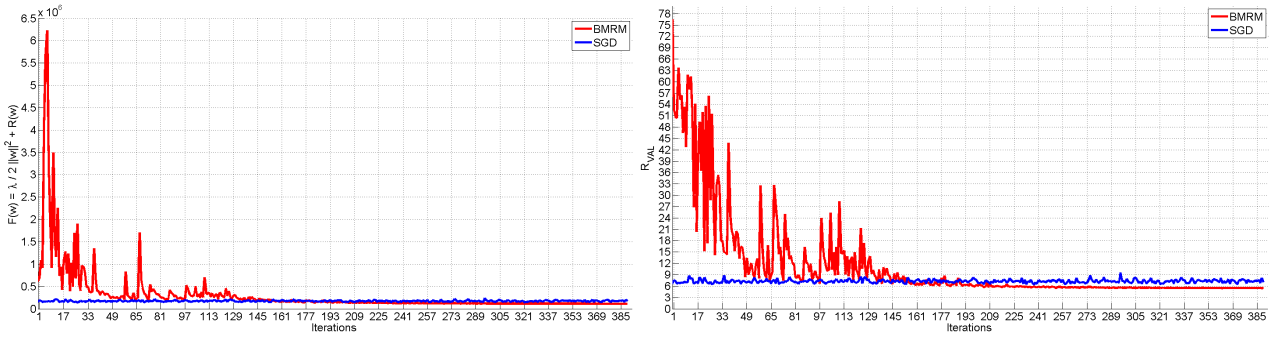
[1] A. Bordes, L. Bottou, and P. Gallinari. SGD-QN: Careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research*, 10:1737–1754, July 2009. 3

[2] Y. Lecun and C. Cortes. The MNIST database of handwritten digits. 5

[3] C. H. Teo, S. Vishwanthan, A. J. Smola, and Q. V. Le. Bundle methods for regularized risk minimization. *J. Mach. Learn. Res.*, 11:311–365, 2010. 1, 2, 6

[4] I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun, and Y. Singer. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005. 4

[5] M. Uřičář, V. Franc, and V. Hlaváč. Detector of facial landmarks learned by the structured output SVM. In *Proceedings of VISAPP*, 2012. to be published. 5



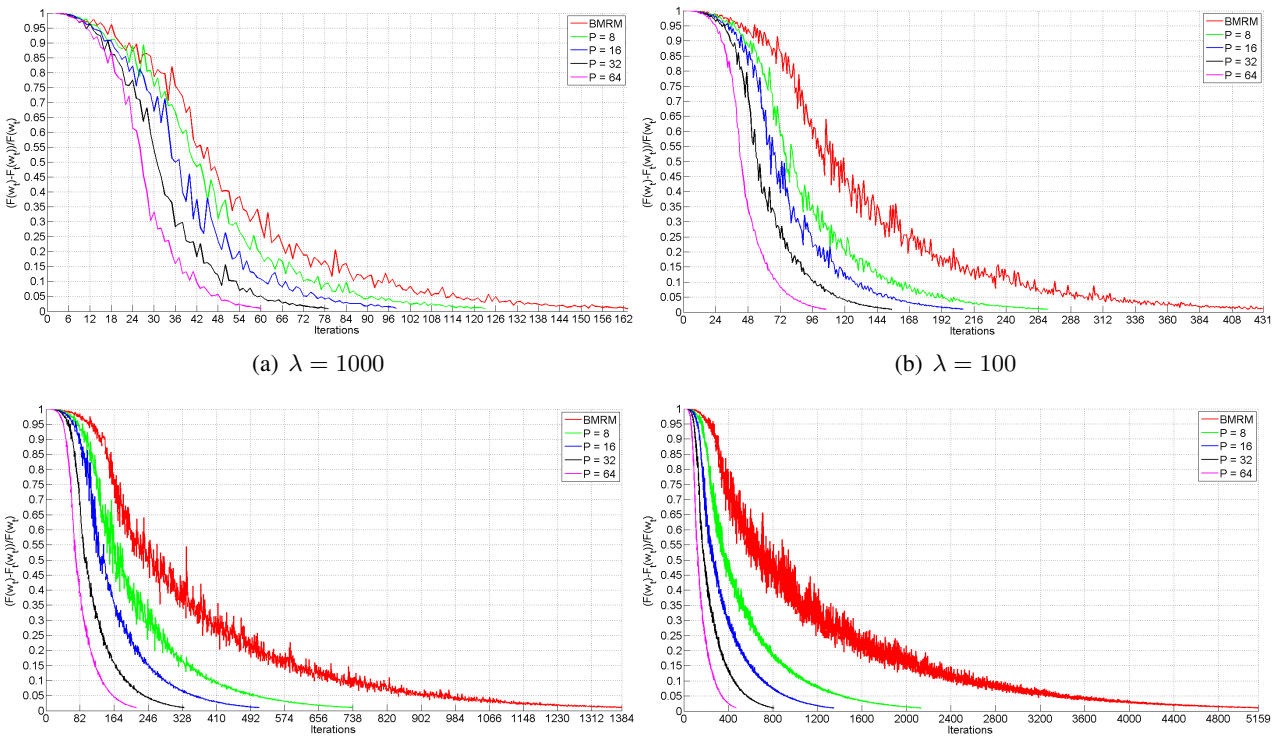
(a) Objective function. (b) Validation risk

Figure 4. The convergence of the objective value (a) and the validation risk (b) for the BMRM and the SGD algorithm on the face landmark detection problem using the optimal setting of the regularization $\lambda = 100$.

OCR — MNIST data

	$\lambda = 1000$		$\lambda = 100$		$\lambda = 10$		$\lambda = 1$	
	#iter	speedup	#iter	speedup	#iter	speedup	#iter	speedup
P = 1 BMRM	163	1	431	1	1384	1	5159	1
P = 8 BMRM	123	10.6	271	12.7	737	15.0	2132	19.4
P = 16 BMRM	98	26.6	208	33.2	512	43.3	1346	61.3
P = 32 BMRM	79	66.0	155	89.0	332	133.4	810	203.8
P = 64 BMRM	60	173.9	106	260.2	218	406.3	470	702.5

Table 2. The table shows the number of iterations needed to reach the ϵ-optimal solution and the speed up of the parallelized version of the algorithm. Each row corresponds to different setting of the number of cutting plane models.



(a) $\lambda = 1000$ (b) $\lambda = 100$

(c) $\lambda = 10$ (d) $\lambda = 1$

Figure 5. Convergence curves of the standard BMRM (i.e. P=1-BMRM) and P-BMRM algorithm applied with different setting of the regularization constant λ to the OCR benchmark problem. The curves show the relative distance to the optimal solution as the function of the number of iterations.