Transductive Minimax Probability Machine

Gao Huang[1,2], Shiji Song[1,2], Zhixiang (Eddie) Xu[3], and Kilian Weinberger[3]

[1] Tsinghua National Laboratory for Information Science and Technology (TNList)
[2] Department of Automation, Tsinghua University, Beijing, 100084 China
[3] Department of Computer Science & Engineering, Washington University, St. Louis, MO 63130 USA

huang-g09@mails.tsinghua.edu.cn, shijis@mail.tsinghua.edu.cn, xuzx@cse.wustl.edu, kilian@wustl.edu

Abstract. The Minimax Probability Machine (MPM) is an elegant machine learning algorithm for inductive learning. It learns a classifier that minimizes an upper bound on its own generalization error. In this paper, we extend its celebrated *inductive* formulation to an equally elegant *transductive* learning algorithm. In the transductive setting, the label assignment of a test set is already optimized during training. This optimization problem is an intractable mixed-integer programming. Thus, we provide an efficient label-switching approach to solve it approximately. The resulting method scales naturally to large data sets and is very efficient to run. In comparison with nine competitive algorithms on eleven data sets, we show that the proposed Transductive MPM (TMPM) almost outperforms all the other algorithms in both accuracy *and* speed.

Keywords: minimax probability machine, transductive learning, semi-supervised learning.

1 Introduction

The Minimax Probability Machine (MPM) was originally introduced by Lanckriet et al. and provides an elegant approach to *inductive* supervised learning. It trains a discriminant classifier that directly minimizes an upper bound on its own generalization error. In particular, it first estimates the first and second moments of the conditional class distributions empirically. Building upon the celebrated work in [8] and [2], it then trains a classifier to minimize the worst case (maximal) probability of a test point falling on "the wrong side" of the decision hyperplane.

In this paper we revisit the MPM and extend it to an equally elegant *transductive* formulation. In transductive learning (TL) [20], the *unlabeled* test data is available during training and the label assignment is optimized directly while the classifier is learned. In classification settings, this results in an integer assignment problem, which is inherently NP-hard [11]. Nevertheless, many approaches have been proposed, typically based on clever heuristics including spectral graph partitioning [9], support vector machines [10], and others [23].

The MPM framework can incorporate the transductive label assignment problem much more naturally and efficiently than other learning paradigms, *e.g.,* Support Vector Machines (SVM) [16]. First, MPM has the attractive property that its learning complexity is independent of the size of training set provided the first and second moments of the conditional class distributions are given, enabling it handle large amount of training samples effortlessly. Second, the two steps of MPM, first estimating the conditional data distribution and then optimizing the hyperplane, give rise to an EM-like transductive algorithm [4] that is both highly efficient and accurate. As a first step, the test-label assignments are optimized (by label switching) to give rise to conditional probability distributions that maximize the *worst-case separation probability* with the current hyperplane. As a second step, the hyperplane is retrained based on the updated label assignments.

We first formulate the *Transductive Minimax Probability Machine* (TMPM) as an exact mixed-integer prgramming and then formalize our approximate solution. We show that the proposed algorithm provably increases the problem objective with every update and converges in a finite number of iterations. As both steps of TMPM are highly efficient, the algorithm scales to large data sets effortlessly. Similar to Transductive SVM (TSVM) [9], TMPM is particularly well suited for data sets with inherent cluster structure. In the presence of underlying manifold structure, Laplacian regularization [1] is often used for semi-supervised learning. We show that TMPM can be further extended to also incorporate such manifold smoothing if it is supported by the data set.

Finally, we evaluate the efficacy of TMPM on an extensive set of real world classification tasks. We compare against nine state-of-the-art learning algorithms and show that TMPM clearly outperforms most of them in *speed and accuracy* with an impressive consistency across learning tasks.

2 Related Works

Several extensions to the MPM [13] have been explored before, in particular for handling uncertain or missing data [3,17]. These works can be seen as dealing with missing information in the input space, while our work is dealing with missing information in the label space. The recent work [12] adopted the minimax probability approach for multiple instance learning. Huang et al. [7] proposes a semi-supervised learning method by combining k-nearest neighbors with a robust extension of MPM. Prior work by Nigam el al. [14] utilizes similar structure with the EM algorithm. The low density separation (LDS) semi-supervised algorithm proposed in [6] builds a fully connected graph kernel and trains a transductive SVM [9] to learn a hyperplane that traverses a low density region between clusters.

Perhaps most similar to our work is the Transductive SVM (TSVM) [9], which also iterates between label switching and classifier re-training. In contrast to TSVM, our algorithm is based on MPM, which greatly reduces the computational cost of re-training. Moreover, we further improve the efficiency drastically by adopting the idea of switching multiple class assignments at a time as the large-scale extension of TSVM [18]. Therefore, the proposed algorithm only re-trains MPM very few times. Additionally, TMPM provably optimizes a well-defined global objective function with each iteration, without heuristically adjusting it (gradually up-weight unlabeled data) during training as in TSVM [9].

3 Minimax Probability Machine

Consider the binary classification case, where we are given labeled training inputs $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \in \mathcal{R}^d$ and their corresponding labels $\{y_1, \ldots, y_m\} \in \{-1, +1\}$. We are also provided with *unlabeled* test inputs $\{\mathbf{x}_{m+1}, \ldots, \mathbf{x}_n\} \in \mathcal{R}^d$.

Let us denote the two class-conditional data distributions as $\mathbb{P}(\mathbf{x}_+|y = +1)$ and $\mathbb{P}(\mathbf{x}_-|y = -1)$, respectively. MPM aims to learn a hyperplane $\{\mathbf{w}, b\}$ that separates positive and negative classes with maximum probability. Since the true class-conditional distributions $\mathbb{P}$ are usually unknown, Lanckriet et al. [13] propose to maximize the worst case probability p that the two classes are separated:

$$\begin{aligned} \max_{p,\mathbf{w}\neq 0,b} \quad & p \\ \text{s.t.} \quad & \inf_{\mathbb{P}\in\mathcal{S}_+} \mathbb{P}(\mathbf{w}^\top \mathbf{x}_+ + b \geq 0| + 1) \geq p, \\ & \inf_{\mathbb{P}\in\mathcal{S}_-} \mathbb{P}(\mathbf{w}^\top \mathbf{x}_- + b \leq 0| - 1) \geq p. \end{aligned} \tag{1}$$

To make this optimization tracktable, the infimums are constrained to sets of distributions $\mathcal{S}_+, \mathcal{S}_-$ that match the empirical first and second order moments of the training data. Let us denote these estimated moments as mean $\hat{\mu}_+$ and covariance $\hat{\Sigma}_+$ for the positive class and $\hat{\mu}_-, \hat{\Sigma}_-$ for the negative class respectively. Then $\mathcal{S}_+$ is defined as:

$$\mathcal{S}_+ = \left\{ \mathbb{P} \,:\, \mathbb{E}[\mathbf{x}] = \hat{\mu}_+ \wedge \mathbb{E}[(\mathbf{x} - \mu_+)(\mathbf{x} - \mu_+)^\top] = \hat{\Sigma}_+ \right\}.$$

Based on the prior work in [8] and [2], Lanckriet et al. [13] show that with this restriction, the separating probability constraints in (1) can be converted into tractable inequality constraints:

$$\mathbf{w}^\top \hat{\mu}_+ + b \geq \kappa\sqrt{\mathbf{w}\hat{\Sigma}_+\mathbf{w}}, \tag{2}$$

where $\kappa = \sqrt{p/(1-p)}$ and the inequality for the negative class is similarly defined. The above inequality can be proven with the multivariate Chebyshev inequality, and we refer readers to [13] for details.

With inequality (2), the optimization in (1) can be converted into the following unconstrained optimization problem [13], which accesses the data only through the empirical estimates of the first and second order moments,

$$\max_{\mathbf{w}} \; \kappa := \frac{\mathbf{w}^\top(\hat{\mu}_+ - \hat{\mu}_-)}{\sqrt{\mathbf{w}^\top(\hat{\Sigma}_+ + \Sigma_{\delta_+})\mathbf{w}} + \sqrt{\mathbf{w}^\top(\hat{\Sigma}_- + \Sigma_{\delta_-})\mathbf{w}}}. \tag{3}$$

Here, $\Sigma_{\delta_+}, \Sigma_{\delta_-}$ are regularization terms, often set to $\sigma^2\mathbf{I}$ for some small σ, or proportional to the diagonal elements of the covariance matrix of all training inputs.

Let $\mathbf{w}^*$ denote the optimal solution to (3), then the optimal bias term can be computed as $b^* = -\mathbf{w}^{*\top}\hat{\mu}_+ + \kappa^*\sqrt{\mathbf{w}^{*\top}(\hat{\Sigma}_+ + \Sigma_{\delta_+})\mathbf{w}^*}$. The optimal separation probability corresponds to $p^* = \kappa^{*2}/(1+\kappa^{*2})$. The optimization (3) can be solved by an iterative least-squares method [13] with a worst-case computational complexity of $O(d^3)$. If the cost of estimating μ_+, μ_-, Σ_+ and Σ_- is taken into account, then the total complexity of this approach is $O(d^3 + md^2)$.

4 Transductive Minimax Probability Machine

In this section, we introduce our transductive extension to MPM, which we refer to as TMPM. In transductive learning [20], the unlabeled test data is available during training and it is allowed to assign the labels directly as part of the learning procedure. We first formalize the TMPM optimization problem, which is NP-hard. We then introduce an efficient algorithm to find an approximate solution. Finally, we prove that our algorithm monotonically increases the objective function value and converges in a finite number of steps.

4.1 Setup

Let $\hat{\mathbf{y}} = [\hat{\mathbf{y}}^l; \hat{\mathbf{y}}^u] \in \{-1,+1\}^n$ denote the class assignment vector for both labeled ($\hat{\mathbf{y}}^l$) and unlabeled ($\hat{\mathbf{y}}^u$) inputs (the class assignments for labeled inputs are fixed, and thus $\hat{\mathbf{y}}^l = \mathbf{y}$). We also let $\mathcal{D}_+, \mathcal{D}_-$ denote the sets of positive and negative labeled *test* inputs respectively, given the current class assignment vector $\hat{\mathbf{y}}$.

Transductive estimation of μ and Σ. A key aspect of TMPM is to incorporate test inputs into the empirical estimation of the mean $(\hat{\mu}_+, \hat{\mu}_-)$ and covariance $\hat{\Sigma}_+, \hat{\Sigma}_-$ of the two class distributions. Since they depend on the class assignment, we write them as functions of $\hat{\mathbf{y}}$:

$$\hat{\mu}_+(\hat{\mathbf{y}}) = \frac{1}{|\mathcal{D}_+|}\sum_i \mathbf{x}_i, \forall \mathbf{x}_i \in \mathcal{D}_+$$

$$\hat{\Sigma}_+(\hat{\mathbf{y}}) = \frac{1}{|\mathcal{D}_+|}\sum_i (\mathbf{x}_i - \hat{\mu}_+)(\mathbf{x}_i - \hat{\mu}_+)^\top, \forall \mathbf{x}_i \in \mathcal{D}_+$$

The corresponding $\hat{\mu}_-(\hat{\mathbf{y}})$ and $\hat{\Sigma}_-(\hat{\mathbf{y}})$ can be computed in a similar fashion.

Mixed-Integer Optimization. Our goal is to find the best class assignment $\hat{\mathbf{y}}^u$ for the test inputs and the corresponding MPM classifier $\mathbf{w}$ simultaneously. The joint search over $\mathbf{w} \in \mathcal{R}^d$ and $\hat{\mathbf{y}}^u \in \{-1,+1\}^{m-n}$ leads to the following mixed-integer optimization problem:

$$\max_{\mathbf{w},\hat{\mathbf{y}}^u} \kappa := \frac{\mathbf{w}^\top\Big(\hat{\mu}_+(\hat{\mathbf{y}}) - \hat{\mu}_-(\hat{\mathbf{y}})\Big)}{\sqrt{\mathbf{w}^\top(\hat{\Sigma}_+(\hat{\mathbf{y}})+\Sigma_{\delta_+})\mathbf{w}}+\sqrt{\mathbf{w}^\top(\hat{\Sigma}_-(\hat{\mathbf{y}})+\Sigma_{\delta_-})\mathbf{w}}},$$
$$\text{s.t.} \quad \frac{1}{n}\sum_{i=1}^{n}\hat{y}_i = 2r-1 \qquad (4)$$

where $\hat{y}_i$ denotes the class assignment for input $\mathbf{x}_i$. The equally constraint enforces the fraction of positive test inputs to match $r(0 < r < 1)$, which can be set according to prior knowledge or estimated from training labels.

Note that in the above formulation, the class conditioned means and convariances are estimated from both *labeled and unlabeled* data. Therefore, the empirical moments are functions of $\hat{\mathbf{y}}^u$, which are also optimization variables.

4.2 The TMPM Algorithm

The optimization problem (4) is computationally intractable to solve globally when the number of input n data is large.

Inspired by Transductive SVM [9], we adopt the strategy of label-switching, and approximate (4) with a iterative greedy procedure. Specifically, we alternately optimize the class assignment $\hat{\mathbf{y}}^u$ and MPM classifier $\mathbf{w}$. First, we keep the MPM classifier $\mathbf{w}$ fixed and optimize the class assignment $\hat{\mathbf{y}}$ through label switching, and then we fix the class assignment, and re-optimize the MPM classifier $\mathbf{w}$.

Initialization. In order to initialize the labels $\hat{\mathbf{y}}^u$, we first train a regular MPM (3) on the labeled training data and then use the resulting classifier to obtain predictions on the test data. To ensure the label assignment is within the feasible set of (4), *i.e.,* its class ratio matches r, we assign the $r(n-m)$ test inputs with highest prediction values to class $+1$, and the rest to class -1.

Classifier Re-optimization. Once the test labels are assigned, we re-train the MPM parameters $\mathbf{w}, b$ on the full (train and test) data set with the actual training labels $\hat{\mathbf{y}}^\ell$ and the (temporarily) assigned labels $\hat{\mathbf{y}}^u$. Note that the re-optimization of $\{\mathbf{w}, b\}$ is actually optimizing κ with fixed $\hat{\mathbf{y}}$. We use the resulting classifier to generate new predictions $t_i = \mathbf{w}^\top\mathbf{x}_i + b$ for all test inputs $\mathbf{x}_i$. These predictions are not immediately used to update the tentative labels of inputs $\mathbf{x}_i$. Instead, it will be used to guide the label switching procedure in the subsequent paragraph.

Label Switching. After the MPM is retrained and the predictions t_i are computed, we re-optimize the assignments of $\hat{\mathbf{y}}^u$ through label switching as in TSVM [9]. However, unlike TSVM in which identifying a candidate pair of labels for switching is straightforward by checking the values of slack variables, TMPM has a more complicated objective function with respect to the label assignments. A naive implementation is to tentatively switch each pair of labels to see if it increases the objective value. But this leads to a worst case complexity of

Algorithm 1. The TMPM algorithm

```
 1: Input: Labeled training inputs and their corresponding labels $\{\mathbf{x}_i, y_i\}_{i=1}^m$; Unlabeled testing inputs $\{\mathbf{x}_i\}_{i=m+1}^n$.
 2: Parameters: $\lambda$ (for regularization)
 3: Compute class ratio $r$ on $\mathbf{y}$ or using prior knowledge.
 4: Initialize class assignment vector $\hat{\mathbf{y}} = [\mathbf{y}, \hat{\mathbf{y}}^u]$ by training a MPM using labeled inputs, and assign $\lceil r(n-m) \rceil$ unlabeled test inputs with highest predicting value to class $+1$, and the rest to class $-1$.
 5: Compute $\hat{\mu}_+$, $\hat{\mu}_-$, $\hat{\Sigma}_+$, $\hat{\Sigma}_-$.
 6: Set $\Sigma_{\delta_+} = \Sigma_{\delta_-} = \lambda diag(\mathbf{v})$, where $\mathbf{v}$ are the diagonal elements of the covariance matrix of all training inputs.
 7: while $true$ do
 8:    $(\mathbf{w}, b)$ = Train MPM $(\hat{\mu}_+, \hat{\mu}_-, \hat{\Sigma}_+, \hat{\Sigma}_-, \Sigma_{\delta_+}, \Sigma_{\delta_-})$
 9:    $(t, \bar{t}_+, \bar{t}_-)$ = Predict MPM $(\mathbf{w}, b, \mathbf{x})$
10:    if $\nexists(\mathbf{x}_i, \mathbf{x}_j)$ satisfying conditions in (5) break
11:    while $\exists(\mathbf{x}_i, \mathbf{x}_j)$ satisfying conditions in (5) do
12:       Switch the labels of $\mathbf{x}_i$ and $\mathbf{x}_j$ $(\hat{y}_i \Leftrightarrow \hat{y}_j)$.
13:       Update $\hat{\mu}_+$, $\hat{\mu}_-$, $\hat{\Sigma}_+$, $\hat{\Sigma}_-$.
14:       Update $\bar{t}_+ = \mathbf{w}^\top \hat{\mu}_+ + b$;   $\bar{t}_- = \mathbf{w}^\top \hat{\mu}_- + b$;
15:    end while
16: end while
17: Output: Class assignment vector on test inputs $\hat{\mathbf{y}}^u$, MPM classifier $(\mathbf{w}, b)$.
```

$O(n^4 d^2)$[1], which is computationally inefficient when test data set is large. Here we introduce a method to quickly identify candidate label pairs and update the means and covariances at minimum cost, reducing the complexity of label switching at each iteration to $O(n\log(n) + nd^2)$.

Let us define the average prediction of class $+1$ as $\bar{t}_+ = \mathbf{w}^\top \hat{\mu}_+ + b$ and similarly $\bar{t}_-$. We search for pairs of *test* inputs $(\mathbf{x}_i \in \mathcal{D}_+, \mathbf{x}_j \in \mathcal{D}_-)$ who, if their labels were switched, would improve the objective κ. As we derive in the subsequent section, we can identify such pairs as inputs $\mathbf{x}_i, \mathbf{x}_j$ whose prediction values (t_i, t_j) satisfy the following two conditions:

$$\begin{aligned} &1.\ t_i < t_j \\ &2.\ \bar{t}_- \le \frac{t_i + t_j}{2} \le \bar{t}_+ \end{aligned} \tag{5}$$

Intuitively these two conditions require to check for: 1. the input $\mathbf{x}_i$, which is currently considered *positive*, has a *lower* prediction value than input $\mathbf{x}_j$, which is assumed to be *negative*; 2. The average of the two predictions t_i and t_j lies between the two class averages. We will prove in Theorem 1 that by switching label pairs that meet these conditions, the objective strictly increases.

It is efficient to search for label pairs for switching based on the above conditions. At each iteration, we first find the n_+ positively labeled test data whose

[1] First, explicitly computing the objective value requires $O(nd^2)$, and there are $O(n^2)$ candidate pairs for each label switching; Second, the number of label-pairs to be switched at each iteration is proportional to n.

prediction values are smaller than the maximal prediction of $\mathcal{D}_-$, and similarly we find the n_- negatively labeled test inputs whose predictions are above the minimal prediction of $\mathcal{D}_+$. These inputs are the candidates that meet Condition 1. This step requires $O(n)$ time of computation. Second, we sort the prediction values of these candidates, which can be done in $O(n\log(n))$ time in a worst case. Finally, we iteratively match the candidate from $\mathcal{D}_+$ with the lowest prediction with the candidate from $\mathcal{D}_-$ with the highest prediction. If they meet Condition 2, we switch their labels, update the means and covariances, and eliminate both of them from the candidate set; otherwise, we remove the positively (negatively) candidate if the right (left) inequality of Condition 2 is violated. It can be verified that these eliminated instance will never meet the switching conditions in this iteration. This procedure has a worst case complexity of $O(nd^2)$, if we update the empirical moments using the rules in the following lemma.

Lemma 1. *Let $\{\hat{\mu}'_+, \hat{\Sigma}'_+\}$ denote the estimated mean and covariance of positive class after switching two instances $\mathbf{x}_i \in \mathcal{D}_+$ and $\mathbf{x}_j \in \mathcal{D}_-$. Then we have*

$$\hat{\mu}'_+ = \hat{\mu}_+ + \frac{1}{|\mathcal{D}_+|}(\mathbf{x}_j - \mathbf{x}_i), \tag{6}$$

$$\begin{aligned}\hat{\Sigma}'_+ =& \hat{\Sigma}_+ + \frac{|\mathcal{D}_+|-1}{|\mathcal{D}_+|^2}(\mathbf{x}_j - \mathbf{x}_i)(\mathbf{x}_j - \mathbf{x}_i)^\top + \\ & \frac{1}{|\mathcal{D}_+|}(\mathbf{x}_i - \mu_+)(\mathbf{x}_j - \mathbf{x}_i)^\top + \frac{1}{|\mathcal{D}_+|}(\mathbf{x}_j - \mathbf{x}_i)(\mathbf{x}_i - \mu_+)^\top,\end{aligned} \tag{7}$$

where $\{\hat{\mu}_+, \hat{\Sigma}_+\}$ are the estimated mean and covariance before label switching.

Remark 1. Naturally, we can update the mean and covariance of the negative class in a similar fashion. The above expressions enable us to re-estimate the means and covariances after each label switching at a minimum cost, with a complexity of $O(d^2)$.

Termination. We keep iterating between label switching and MPM re-training until no more pairs can be found that satisfy (5). The TMPM algorithm is summarized in pseudo-code in Algorithm 1.

Remark 2. The overall time complexity of Algorithm 1 is $O(L(d^3 + n\log(n) + nd^2))$, where L is the number of outer loop executions. Typically, we have $L \approx 10$. The term $O(d^3)$ results from re-training the MPM, following the method proposed in [13]. The terms $O(n\log(n) + nd^2)$ capture the complexity of finding eligible pairs for switching and updating empirical moments.

4.3 Algorithm Analysis

In this subsection, we show that Algorithm 1 terminates in a finite number of iterations.

Firstly, we prove that each MPM label switching strictly increases κ in (4). We formalize this guarantee as Theorem 1.

Theorem 1. *If two test inputs $\mathbf{x}_i \in \mathcal{D}_+$, $\mathbf{x}_j \in \mathcal{D}_-$ and their corresponding predictions t_i, t_j satisfy the switching conditions in (5), then by switching the assigned labels of $\mathbf{x}_i$ and $\mathbf{x}_j$ ($\hat{y}_i \Leftrightarrow \hat{y}_j$), the objective function value κ in (4) strictly increases.*

Proof: We first show that the numerator of (4) increases after the label switching. This follows from plugging in the mean and covariance updates stated above in Eqs. (6) and (7).

$$\begin{aligned}&\mathbf{w}^\top(\hat{\mu}'_+ - \hat{\mu}'_-)\\&= \mathbf{w}^\top(\hat{\mu}_+ - \hat{\mu}_-) + (\frac{1}{|\mathcal{D}_+|} + \frac{1}{|\mathcal{D}_-|})(t_j - t_i)\\&> \mathbf{w}^\top(\hat{\mu}_+ - \hat{\mu}_-).\end{aligned}$$

The last inequality holds because of the condition $t_i < t_j$.

As a second step, we show that the denominator of (4) decreases after switching the labels. Again, by using the update rules in (6) and (7), we have

$$\begin{aligned}\mathbf{w}^\top\hat{\Sigma}'_+\mathbf{w} =&\mathbf{w}^\top\hat{\Sigma}_+\mathbf{w} + \frac{|\mathcal{D}_+|-1}{|\mathcal{D}_+|^2}(\mathbf{w}^\top(\mathbf{x}_j - \mathbf{x}_i))^2\\&+ \frac{2}{|\mathcal{D}_+|}\mathbf{w}^\top(\mathbf{x}_i - \mu_+)(\mathbf{x}_j - \mathbf{x}_i)^\top\mathbf{w}\\=&\mathbf{w}^\top\hat{\Sigma}_+\mathbf{w} + \frac{1}{|\mathcal{D}_+|}\mathbf{w}^\top(\mathbf{x}_j - \mathbf{x}_i)\mathbf{w}^\top(\mathbf{x}_j + \mathbf{x}_i - 2\hat{\mu}_+)\\&- \frac{1}{|\mathcal{D}_+|^2}(\mathbf{w}^\top(\mathbf{x}_j - \mathbf{x}_i))^2\\=&\mathbf{w}^\top\hat{\Sigma}_+\mathbf{w} + \frac{(t_j - t_i)(t_i + t_j - 2\bar{t}_+)}{|\mathcal{D}_+|} - \frac{(t_j - t_i)^2}{|\mathcal{D}_+|^2}\\<&\mathbf{w}^\top\hat{\Sigma}_+\mathbf{w},\end{aligned}$$

where the last inequality holds because of the switching conditions $(t_i + t_j)/2 \leq \bar{t}_+$ and $t_i < t_j$ given in Theorem 1.

Following a similar line of reasoning, we can show that

$$\mathbf{w}^\top\hat{\Sigma}'_-\mathbf{w} < \mathbf{w}^\top\hat{\Sigma}_-\mathbf{w}. \tag{8}$$

Further, notice that Σ_{δ_+} and Σ_{δ_-} are independent of class assignment vector $\hat{\mathbf{y}}$, we have the denominator of (4) decreases after label switching.

As the enumerator of κ increases and its denominator decreases, and as the label switching preserves the class ratio of of $\hat{\mathbf{y}}^u$, it follows that the objective κ in (4) strictly increases with each label switching. ■

Theorem 2. *Algorithm 1 terminates after a finite number of iterations.*

Proof: Since the label switching strictly increases the objective according to Theorem 1, and the re-training of MPM never decrease the objective, the alternating optimization method in Algorithm 1 strictly improves the value of κ at each iteration. Therefore, the outer loop cannot repeat label assignments (Otherwise, we will solve a same MPM model in two differently iterations, which will yield a same value of κ. This means the objective is not strictly increasing over these two iterations, leading to a contradictory). Since the number of label assignments is finite, the algorithm must terminate after a finite number of iterations. $\blacksquare$

Note that Theorem 2 is based on a worst case scenario analysis. In practice, each iteration switches many labels and TMPM terminates after a small number(≈ 10) of iterations.

4.4 TMPM with Manifold Regularization

Transductive learning or semi-supervised learning algorithms can often assist classifiers by revealing underlying structure in the data distribution. A successful approach is manifold regularization, introduced by Belkin et al. [1]. Here, a proximity graph is created and the classifier is regularized to make similar predictions on similar inputs. This approach is typically successful if the data distribution obeys some intrinsically low dimensional manifold structure. TMPM can also incorporate manifold regularization naturally. We replace the regularization covariance matrices $\Sigma_{\delta_+}, \Sigma_{\delta_-}$ in the TMPM objective function (4) with a Laplacian regularization term [21]. More formally, we set $\Sigma_{\delta_+} = \Sigma_{\delta_-} = \lambda \mathbf{X}^\top \mathbf{L} \mathbf{X}$, where $\mathbf{X}^\top = [\mathbf{x}_1, \ldots \mathbf{x}_n]$ is a matrix containing both training and test inputs and $\mathbf{L} \in \mathbb{R}^{n \times n}$ denotes the *graph Laplacian* constructed from $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Finally, λ denotes a regularization tradeoff parameter. We refer to this manifold regularization extension as TMPM^{mr}.

5 Results

We evaluate TMPM on a wide variety of synthetic and real world data sets. Our implementation is implemented in MATLAB^{TM}, and is executed on an Intel i7 Quad Core CPU 3.20GHz machine with 32GB RAM.

5.1 Toy Example

We use a toy data set to visualize the transductive learning process of TMPM. The toy data set is a binary classification problem, where each class contains 200 inputs generated from a 2-dimensional Gaussian distribution. We randomly reveal one label from each class to create a training set with two instances. The remaining inputs are unlabeled and constitute the test data, see Figure 1 (upper left panel). On this data, a linear SVM achieves 0.78 test accuracy.

Figure 1 visualizes the decision boundary and label assignments of each iteration of TMPM until termination. Inputs that will be switched in this iteration are highlighted with circles. The inner ellipsoids represent the covariances of the data in $\mathcal{D}_+, \mathcal{D}_-$, and the outer ellipsoids are κ times larger, so that the decision plane is tangent to both of the outer ellipsoids. Since $p = \kappa^2/(1+\kappa^2)$, the larger κ, the higher the minimax probability p of separating the two subsets. The value

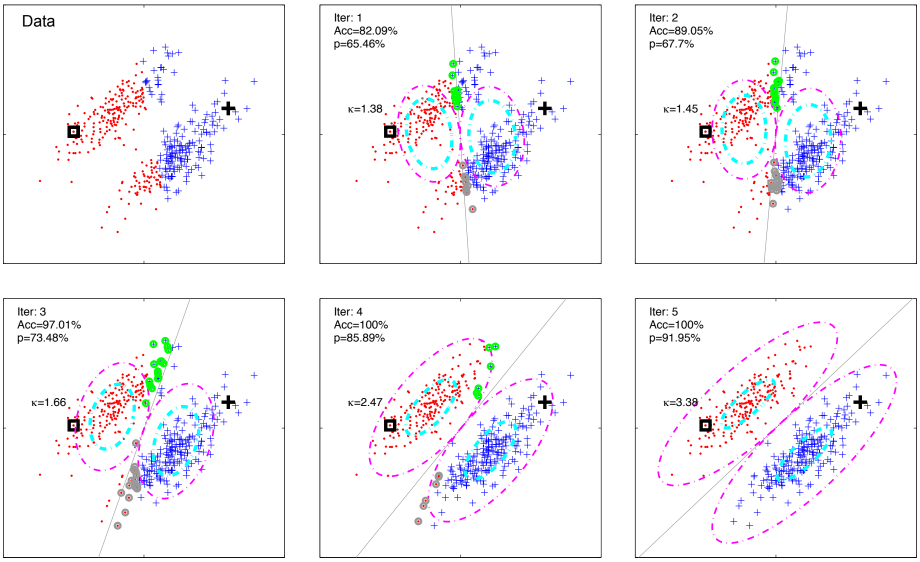

Fig. 1. The TMPM algorithm visualized on a toy data set. Only two inputs are initially labeled (big square and cross, *top left*). Small dots and crosses indicate the labels assigned by TMPM to the test data. Inputs highlighted with circles are those to be switched in a particular current iteration. The inner ellipsoids visualize the covariances of two classes, and the outer ellipsoids represent κ times of the covariances.

of p and the classification accuracy are indicated in the top left of each frame, both of which increase monotonically until the algorithm terminates after the 5^{th} iteration, when no more pairs of inputs satisfy the conditions for switching. As a byproduct, we obtain a lower bound $p = 91.95\%$ probability of separating these two classes for additional unseen data. We also obtain the estimated means and covariances for the two classes:

$$\hat{\mu}_+ = (1.01, -1.00), \quad \hat{\mu}_- = (-1.00, 0.94),$$
$$\hat{\Sigma}_+ = \begin{bmatrix} 0.94 & 0.81 \\ 0.81 & 1.04 \end{bmatrix}, \hat{\Sigma}_- = \begin{bmatrix} 0.89 & 0.72 \\ 0.72 & 0.87 \end{bmatrix},$$

which are very close to the true means and covariances of the two Gaussian distributions that generate the data:

$$\mu_+ = (1, -1), \quad \mu_- = (-1, 1), \quad \Sigma_+ = \Sigma_- = \begin{bmatrix} 1.0 & 0.8 \\ 0.8 & 1.0 \end{bmatrix}.$$

5.2 Transductive Learning Results

We evaluate TMPM on several real-world data sets, and compare against state-of-the-art transductive/semi-supervised learning algorithms. The characteristics of these data sets are summarized in the second and third rows of Table 1.

Experiment Setup. For each data set, we randomly select 10 samples as labeled set, another 10 samples as validation set, and the rest as unlabeled test set. The prediction error rate on the unlabeled test set is used as the evaluation criteria, and we report the average results over 20 runs with randomly selected labeled/validation/unlabeled data. The linear TMPM is used for all the experiments (for high dimensional data where $d > n$, we use *linear kernel* TMPM given in the Appendix). The only hyper-parameter in linear TMPM is the regularization coefficient λ, which is selected from the candidate set $[10^{-4}, 10^{-3}, \ldots, 10^{4}]$ based on the performance on the validation set.

Datasets. The *g50c*, *g241c*, *digit1*, *text* data sets are obtained form Olivier Chapelle's Semi-Supervised Learning benchmark data set collection[2] [5]. The data set *breast*, *australian* (Australian Credit Approval), *pcmac* (corresponding to two classes of the 20 newsgroups data set), *adults*, *kddcup*, are taken from the UCI Machine Learning Repository[3].

Baselines. First, the SVM and MPM are trained using only the labeled data, and we report the better results of a linear version and an *RBF* kernel version of these algorithms. Other baselines include the Transductive SVM (TSVM^{light}) [9], the TSVM with multiple switching strategy (TSVM^{ms}) [18], the semi-supervised EM with Gaussian distribution assumption (for all data sets except *pcmac* and *text*) and with multinomial distribution assumption (for *pcmac* and *text*), the low density separation algorithm (LDS) [6], and the squared-loss mutual information regularization (SMIR) [15]. For TSVM^{light} and TSVM^{ms}, we report both linear and RBF kernel version results. The tradeoff parameter C in these algorithms are selected from the set $[10^{-4}, 10^{-3}, \ldots, 10^{4}]$, and the kernel width is selected from the set $[2^{-5}, 2^{-4}, \ldots, 2^{1}]$ times the average pairwise distance of the training data. The kernel type is indicated in sub-scripts (*e.g.* TSVM$^{light}_{linear}$ and TSVM$^{light}_{rbf}$). For EM^{gauss}, a ridge $\lambda \boldsymbol{I}$ is added to the covariance matrix when computing posterior probability, where λ is selected from the set $[10^{-6}, 10^{-5}, \ldots, 10^{2}]$. For LDS and SMIR, we follow the suggestions in [6] and [15] to create a candidate hyperparameter set and select the best value based on the validation set.

Prediction Accuracy. The experimental results are summarized in Table 1. For each data set, the best performance up to statistical significance is highlighted in **bold**. Standard deviations are provided inside parenthesis. If an algorithm is not able to scale to a particular data set (or fails to converge), it is indicated with *N/A*.

A few trends can be observed: 1. TMPM obtains the best result (up to statistical significance) on almost all data sets; 2. TMPM's standard deviation of error is always the lowest among all TL/SSL algorithms over all data sets except *pcmac* and *text*, demonstrating that TMPM is insensitive to the initial predictions on unlabeled test data; 3. Generally, non-linear classifiers do not outperform

[2] `http://olivier.chapelle.cc/ssl-book/benchmarks.html`
[3] `http://archive.ics.uci.edu/ml/datasets.html`

Table 1. Data set statistics and test error rates (in %) on nine benchmark data sets, comparing TMPM against various state-of-the-art algorithms. *N/A* indicates that an algorithm fails to scale to that specific data set. Best results up to statistical significance are highlighted in **bold**.

Statistics	**g50c**	**g241c**	**breast**	**australian**	**digit1**	**pcmac**	**text**	**adults**	**kddcup**
# features	50	241	10	14	241	3289	11960	123	122
# inputs	550	1500	683	690	1500	1943	1500	32541	500000
Algorithm	**Test-error (%)**								
SVM	16.7 ± 4.2	36.9 ± 2.4	3.2 ± 0.3	19.3 ± 5.9	20.0 ± 5.0	38.4 ± 4.9	35.5 ± 4.4	26.8 ± 5.6	5.1 ± 1.7
MPM	18.1 ± 3.9	37.4 ± 2.5	3.3 ± 0.2	19.6 ± 3.2	25.1 ± 3.7	39.7 ± 5.6	38.9 ± 4.0	24.2 ± 4.8	1.6 ± 1.0
TSVM$^{light}_{linear}$	**4.8** ± 0.7	17.5 ± 3.3	**2.8** ± 0.5	16.5 ± 6.7	16.7 ± 2.4	40.9 ± 9.8	**27.0** ± 4.4	N / A	N / A
TSVM$^{light}_{rbf}$	**5.0** ± 0.5	14.3 ± 1.2	3.5 ± 0.7	16.7 ± 4.9	15.6 ± 2.0	41.6 ± 10.4	29.5 ± 6.2	N / A	N / A
TSVM$^{ms}_{linear}$	6.0 ± 1.3	17.1 ± 4.4	3.2 ± 0.1	16.3 ± 4.1	15.7 ± 4.0	38.8 ± 8.9	**26.3** ± 3.7	22.3 ± 4.0	N / A
TSVM$^{ms}_{rbf}$	**4.9** ± 0.6	15.1 ± 5.5	3.3 ± 0.2	17.0 ± 5.5	15.6 ± 3.3	37.8 ± 4.5	27.4 ± 3.8	N / A	N / A
EM	9.3 ± 4.8	37.7 ± 10.0	8.6 ± 4.3	19.9 ± 5.9	11.3 ± 9.1	**33.1** ± 9.8	33.7 ± 0.3	24.2 ± 8.5	1.0 ± 0.4
LDS	9.2 ± 4.9	15.7 ± 6.7	4.3 ± 0.7	16.8 ± 2.8	13.5 ± 6.9	38.4 ± 7.5	27.4 ± 3.8	N / A	N / A
SMIR	17.1 ± 4.4	36.8 ± 3.3	4.0 ± 1.2	19.9 ± 4.0	12.5 ± 5.0	44.7 ± 5.0	35.2 ± 4.6	N / A	N / A
TMPM	**4.8** ± 0.4	**13.1** ± 0.4	**2.8** ± 0.1	15.2 ± 0.8	9.0 ± 1.8	**30.5** ± 8.0	35.2 ± 5.2	**20.5** ± 2.8	**0.6** ± 0.1

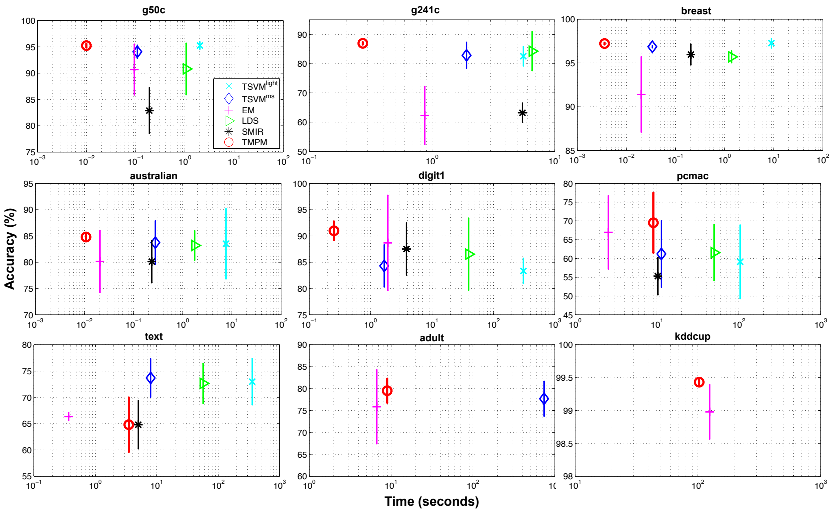

Fig. 2. Classification accuracy (in %) and computational time (in seconds) of different TL algorithms on data sets described in Table 1. Error bar indicates the standard deviation.

linear classifiers, which is not unusual in the TL/SSL setting due to the typically small training set sizes.

Analysis. We explain the strong performance of TMPM in parts on the underlying MPM framework. Compared to TSVM^{light} and TSVM^{ms}, TMPM only yields significantly worse performance on the *text* data, but outperforms or matches both on all other problems.

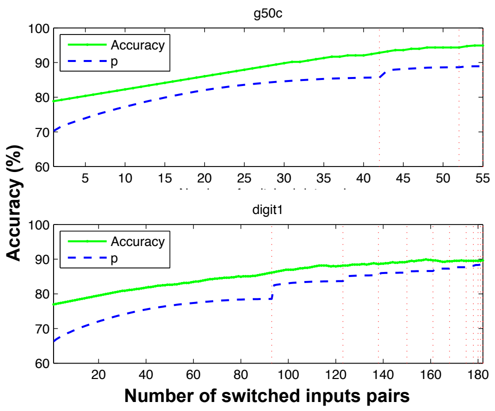

Fig. 3. Transductive learning curve on *g50c* and *digit1*. Each dotted vertical line represents a $(\mathbf{w}, b)$ re-optimization step.

Table 2. Test error rates on data sets with manifold structure. Best results (up to statistical significance) are highlighted in **bold**.

	d	n	SVM	LapSVM	LapMPM	TMPM^{mr}
coil20	1024	576	13.3 ± 4.2	**10.5** ± 3.9	12.9 ± 3.4	**11.7** ± 3.9
uspst	256	803	16.5 ± 2.3	**14.2** ± 2.4	16.6 ± 2.4	**13.6** ± 3.4

Figure 3 shows the transduction accuracy and the value of p with respect to the number of switched label pairs on two representative data sets. Each interval between two vertical lines corresponds to an execution of one inner loop of Algorithm 1. As predicted, p strictly increases after each pair of labels is switched or $\mathbf{w}, b$ are re-optimized. Not surprisingly, the transduction accuracy increases steadily as p increases. We can also observe that the number of switched labels per iteration decrease (roughly) exponentially, indicating TMPM converges quickly in practice.

Speed. In Figure 2 we compare the training time (plotted on a logarithmic scale) and classification accuracy of the above TL/SSL algorithms (we omit TSVM$_{rbf}^{light}$ and TSVM$_{rbf}^{ms}$ here since they are significantly slower than their linear version). The TMPM is the fastest among all the six algorithms on 6/9 of the data sets. It is only slower than the EM^{multi} algorithm on two high dimensional text data (*pcmac*, *text*) sets and *adult* (although the differences are sometimes no more than a few seconds). The TMPM is 1 to 4 orders of magnitude faster than the TSVM^{light}, and is also significantly faster than the TSVM^{ms}. The speed advantage of TMPM over other algorithms is even larger as the unlabeled data increases.

5.3 Performances on Manifold Data Sets

We also evaluate TMPM with manifold regularization (TMPM^{mr}) on two well known data sets considered to have manifold structures (from the UCI data set repository). For each data set, we select 50 inputs as labeled set, 50 as hold-out, and leave the rest as unlabeled test set.

The kernel TMPM with RBF kernel is adopted here, and the kernel width is selected from the set $[2^{-5}, 2^{-4}, \ldots, 2^{1}]$ times the average pairwise distance of the training data. For comparison, we evaluate two representative manifold-based SSL algorithms, LapSVM [1] and LapMPM [22]. For all algorithms, the same graph *Laplacian* is used, whose parameter setting can be found in [19]. The prediction error rates (in %) on unlabeled test set are summarized in Table 2. The results show that TMPM^{mr} is also competitive with LapSVM and LapMPM on these manifold data sets.

6 Conclusion

In this paper, we propose a novel transductive learning algorithm (TMPM) based on the minimax probability machine. Although TL learning is not new, the TMPM framework provides a fresh and exciting approach to transductive learning. The underlying assumption is that the optimal decision hyperplane should lead to a maximum worst-case separation probability between different data classes. We convert this search problem into a mixed-integer programming, and propose an efficient algorithm to approximate it greedily.

We show that TMPM converges in a finite number of iterations and has a low computational complexity in the number of unlabeled inputs. Experimental results demonstrate that TMPM is promising in generalization performance and scales naturally to large data sets.

Acknowledgments. G. Huang and S. Song were supported by the National Natural Science Foundation of China under Grant 61273233, the Research Fund for the Doctoral Program of Higher Education under Grant 20120002110035 and Grant 20130002130010, the National Key Technology Research and Development Program under Grant 2012BAF01B03, the Project of China Ocean Association under Grant DY125-25-02, and the Tsinghua University Initiative Scientific Research Program under Grant 2011THZ07132. K.Q. Weinberger and Z. Xu were supported by NSF IIS-1149882 and IIS-1137211.

References

1. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. The Journal of Machine Learning Research 7, 2399–2434 (2006)
2. Bertsimas, D., Sethuraman, J.: Moment problems and semidefinite optimization. In: Handbook of Semidefinite Programming, pp. 469–509. Springer (2000)

3. Bhattacharyya, C., Pannagadatta, K., Smola, A.J.: A second order cone programming formulation for classifying missing data. In: Neural Information Processing Systems (NIPS), pp. 153–160 (2005)
4. Bishop, C.M., Nasrabadi, N.M.: Pattern recognition and machine learning, vol. 1. Springer, New York (2006)
5. Chapelle, O., Schölkopf, B., Zien, A., et al.: Semi-supervised learning, vol. 2. MIT Press, Cambridge (2006)
6. Chapelle, O., Zien, A.: Semi-supervised classification by low density separation. In: Proceedings of the 10th International Conference on Artificial Intelligence and Statistics, pp. 57–64 (2005)
7. Huang, G., Song, S., Gupta, J.N.D., Wu, C.: A second order cone programming approach for semi-supervised learning. Pattern Recognition 46(12), 3548–3558 (2013)
8. Isii, K.: On sharpness of tchebycheff-type inequalities. Annals of the Institute of Statistical Mathematics 14(1), 185–197 (1962)
9. Joachims, T.: Transductive inference for text classification using support vector machines. In: ICML, vol. 99, pp. 200–209 (1999)
10. Joachims, T., et al.: Transductive learning via spectral graph partitioning. In: ICML, vol. 3, pp. 290–297 (2003)
11. Korte, B.B.H., Vygen, J.: Combinatorial optimization, vol. 21. Springer (2012)
12. Krummenacher, G., Ong, C.S., Buhmann, J.: Ellipsoidal multiple instance learning. In: Proceedings of the 30th International Conference on Machine Learning, pp. 73–81 (2013)
13. Lanckriet, G.R., Ghaoui, L.E., Bhattacharyya, C., Jordan, M.I.: A robust minimax approach to classification. The Journal of Machine Learning Research 3, 555–582 (2003)
14. Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using em. Machine Learning 39(2-3), 103–134 (2000)
15. Niu, G., Jitkrittum, W., Dai, B., Hachiya, H., Sugiyama, M.: Squared-loss mutual information regularization: A novel information-theoretic approach to semi-supervised learning. In: Proceedings of the 30th International Conference on Machine Learning, pp. 10–18 (2013)
16. Schölkopf, B., Smola, A.J.: Learning with kernels. MIT Press (2002)
17. Shivaswamy, P.K., Bhattacharyya, C., Smola, A.J.: Second order cone programming approaches for handling missing and uncertain data. The Journal of Machine Learning Research 7, 1283–1314 (2006)
18. Sindhwani, V., Keerthi, S.S.: Large scale semi-supervised linear svms. In: Proceedings of the 29th annual international ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 477–484. ACM (2006)
19. Sindhwani, V., Niyogi, P., Belkin, M.: Beyond the point cloud: from transductive to semi-supervised learning. In: ICML 2005, pp. 824–831. ACM (2005)
20. Vapnik, V.N.: Statistical learning theory (1998)
21. Weinberger, K.Q., Sha, F., Zhu, Q., Saul, L.K.: Graph laplacian regularization for large-scale semidefinite programming. In: NIPS, pp. 1489–1496 (2006)
22. Yoshiyama, K., Sakurai, A.: Manifold-regularized minimax probability machine. In: Schwenker, F., Trentin, E. (eds.) PSL 2011. LNCS, vol. 7081, pp. 42–51. Springer, Heidelberg (2012)
23. Zhu, X.: Semi-supervised learning literature survey. Computer Science, University of Wisconsin-Madison 2, 3 (2006)

Appendix: Kernel TMPM

Suppose that the labeled training data are re-arranged so that the first m_+ inputs belong to class $+1$, and the rest m_- inputs belong to class -1. Let $\mathbf{K} \in \mathbb{R}^{m\times m}$ $(m = m_+ + m_-)$ denotes the kernel matrix, whose first m_+ rows and last m_- rows are denoted by $\mathbf{K}_+$ and $\mathbf{K}_-$, respectively.

The kernel MPM is formulated as

$$\max_{\boldsymbol{\theta}\in\mathbb{R}^m} \quad \frac{\boldsymbol{\theta}^\top(\boldsymbol{\eta}_+ - \boldsymbol{\eta}_-)}{\sqrt{\boldsymbol{\theta}^\top(\boldsymbol{\Phi}_+^\top\boldsymbol{\Phi}_+ + \lambda_+\mathbf{K})\boldsymbol{\theta}} + \sqrt{\boldsymbol{\theta}^\top(\boldsymbol{\Phi}_-^\top\boldsymbol{\Phi}_- + \lambda_-\mathbf{K})\boldsymbol{\theta}}}$$

where

$$\begin{aligned}
\boldsymbol{\Phi}_+ &= (\mathbf{K}_+ - \mathbf{1}_{m_+}\boldsymbol{\eta}_+^\top)/\sqrt{m_+} \\
\boldsymbol{\Phi}_y &= (\mathbf{K}_- - \mathbf{1}_{m_-}\boldsymbol{\eta}_-^\top)/\sqrt{m_-} \\
[\boldsymbol{\eta}_+]_i &= \sum_{j=1}^{m_+}\mathbf{K}_{j,i}, \\
[\boldsymbol{\eta}_-]_i &= \sum_{j=m_++1}^{m}\mathbf{K}_{j,i},
\end{aligned}$$

and $\mathbf{1}_n$ denotes an all one vector of dimension n.

Based on the kernel MPM, we give the kernel TMPM in Algorithm 2.

Algorithm 2. The Kernel TMPM algorithm

```
1: Input: Labeled data $\{\mathbf{x}_i, y_i\}_{i=1}^m$; Unlabeled test inputs $\{\mathbf{x}_i\}_{i=m+1}^n$;
2: Parameters: $\lambda$ (for regularization)
3: Compute class ratio $r$ on $\mathbf{y}$ or using prior knowledge.
4: Initialize class assignment vector $\hat{\mathbf{y}} = [\mathbf{y}, \hat{\mathbf{y}}^u]$ by training a kernel MPM using
   labeled inputs, and assign $\lceil r(n-m)\rceil$ unlabeled test inputs with highest predicting
   value to class $+1$, and the rest to class $-1$.
5: Compute $\boldsymbol{\eta}_+$, $\boldsymbol{\eta}_-$, $\mathbf{K}_+$ and $\mathbf{K}_-$.
6: while true do
7:    $(\boldsymbol{\theta}, b)$ = Train kernel MPM $(\boldsymbol{\eta}_+, \boldsymbol{\eta}_-, \mathbf{K}_+, \mathbf{K}_-, \lambda)$
8:    $\bar{t}_+ = \boldsymbol{\theta}^\top\boldsymbol{\eta}_+ + b$,   $\bar{t}_- = \boldsymbol{\theta}^\top\boldsymbol{\eta}_- + b$
9:    $t$ = Predict kernel MPM $(\boldsymbol{\theta}, b, \mathbf{x})$
10:   if $\nexists(t_i, t_j)$ satisfying the conditions in (5) break
11:   while $\exists(t_i, t_j)$ satisfying conditions in (5) do
12:      Switch the labels of $\mathbf{x}_i$ and $\mathbf{x}_j$ $(\hat{y}_i \Leftrightarrow \hat{y}_j)$.
13:      Update $\boldsymbol{\eta}_+$, $\boldsymbol{\eta}_-$, $\mathbf{K}_+$ and $\mathbf{K}_-$.
14:      $\bar{s} = \boldsymbol{\theta}^\top\boldsymbol{\eta}_+ - b$;   $\bar{t} = \boldsymbol{\theta}^\top\boldsymbol{\eta}_- - b$;
15:   end while
16: end while
17: Output: Class assignment vector on test inputs $\hat{\mathbf{y}}^u$, MPM classifier $(\boldsymbol{\theta}, b)$.
```