Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Parametric Dual Maximization for Non-Convex Learning Problems

Yuxun Zhou
Department of EECS
UC Berkeley
yxzhou@berkeley.edu

Zhaoyi Kang
Department of EECS
UC Berkeley
kangzy@berkeley.edu

Costas J. Spanos
Department of EECS
UC Berkeley
spanos@berkeley.edu

Abstract

We consider a class of non-convex learning problems that can be formulated as jointly optimizing regularized hinge loss and a set of auxiliary variables. Such problems encompass but are not limited to various versions of semi-supervised learning, learning with hidden structures, robust learning, etc. Existing methods either suffer from local minima or have to invoke a non-scalable combinatorial search. In this paper, we propose a novel learning procedure, namely Parametric Dual Maximization (PDM), that can approach global optimality efficiently with user specified approximation levels. The building blocks of PDM are two new results: (1) The equivalent convex maximization reformulation derived by parametric analysis. (2) The improvement of local solutions based on a necessary and sufficient condition for global optimality. Experimental results on two representative applications demonstrate the effectiveness of PDM compared to other approaches.

1 Introduction

To enhance the performance on more challenging tasks, variations of the classic large margin learning formulation are proposed to incorporate additional modeling flexibility. To name a few, semi-supervised SVM (S^3VM) is introduced in (Bennett, Demiriz, and others 1999; Joachims 1999) to combine labeled and unlabeled samples together for overall risk minimization. To learn a classifier for datasets having unobserved information, SVM with latent variables is proposed in (Felzenszwalb et al. 2010) for object detection and in (Yu and Joachims 2009; Zhou, Hu, and Spanos 2016) for structural learning. Inasmuch as the traditional large margin classifier with hinge loss can be sensitive to outliers, the authors of (Xu, Crammer, and Schuurmans 2006) suggest a ramp loss with which a robust version of SVM is proposed.

Nonetheless, unlike classic SVM learning objective that possesses amiable convexity, those variations introduce non-convex learning objectives, hindering their generalization performance and scalable deployment due to optimization difficulties. In literature, much effort has been made to obtain at least a locally optimal solution: Viewing the problem as a biconvex optimization leads to a series of alternating optimization (AO) algorithms. For example, in (Felzenszwalb et al. 2010), latent SVM was trained by alternately solving standard SVM and updating latent variables. Another widely applied technique is the concave-convex Procedure (CCCP) (Yuille, Rangarajan, and Yuille 2002). Among many others, (Yu and Joachims 2009; Ping, Liu, and Ihler 2014) used CCCP for latent structural SVM training. Direct application of the gradient-based method is especially attractive for large scale problems owing to its low computational cost (Bottou 2010). Such examples include the stochastic gradient descent (SGD) for large margin polytope machine (Kantchelian et al. 2014; Zhou, Jin, and Spanos 2015) and S^3VM (Chapelle and Zien 2005). Combinatorial optimization methods, e.g., the local search method (Joachims 1999) and branch and bound (B & B) (Chapelle, Sindhwani, and Keerthi 2006), were also implemented for small-scale problems. It's worth mentioning that other heuristic approaches and relaxations such as continuation method (Chapelle, Chi, and Zien 2006) and semidefinite program (SDP) relaxation (Bei and Cristianini 2006)(Xu, Crammer, and Schuurmans 2006) have also been examined for several applications.

Yet except B & B, all of the aforementioned methods, i.e., AO, CCCP, and SGD, only converge to local minimums and could be very sensitive to initial conditions. Although SDP approximation yields a convex problem, the quality of the relaxation is still an open question in both theory and practice (Park and Boyd 2015). On the other hand, it has long been realized that global optimal solution can return excellent generalization performance in situations where local optimal solutions fail completely (Chapelle, Sindhwani, and Keerthi 2006). The major issue with B & B is its scalability: the size of the search tree can grow exponentially with the number of integer variables (Krishnamoorthy 2008), making it only suitable for small scale problems. Interested readers are referred to (Chapelle, Sindhwani, and Keerthi 2008) for a thorough discussion.

In this work, we propose a learning procedure, namely Parametric Dual Maximization (PDM), based on a different view of the problem. We first demonstrate that the learning objectives can be rewritten into jointly optimizing regularized hinge loss and a set of auxiliary variables. Then we show that they are equivalent to non-smooth convex maximization through a series of parametric analysis techniques. Finally, we establish PDM by exploiting a necessary and sufficient global optimality condition. Our contributions are highlighted as follows. (1) The equivalence to non-smooth

convex maximization unveils a novel view of an important class of learning problems such as S^{3}VM. Now we know that they are *NP-hard*, but possesses gentle geometric properties that allow new solution techniques. (2) We develop a set of new parametric analysis techniques, which can be reused for many other tasks, e.g., solution path calculation. (3) By checking a necessary and sufficient optimality condition, the proposed PDM can approach the global optimum efficiently with user specified approximation levels.

The rest of the paper is organized as follows. In Section 2, we detail the reformulation of the problem with examples. In Section 3, we derive the equivalent non-smooth convex maximization by parametric analysis. In Section 4, the optimality condition is presented, and the corresponding algorithm is proposed. Numerical experiments are given in Section 5.

2 A Class of Large Margin Learning

A labeled data sample is denoted as $(\boldsymbol{x}_i, y_i)$, with $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$. We focus on the following joint minimization problem

$$\min_{\boldsymbol{p}\in\mathbb{P}} \min_{\boldsymbol{w},b} \quad \mathcal{P}(\boldsymbol{w}, b; \boldsymbol{p}) = \frac{1}{2}||\boldsymbol{w}||^2_{\mathcal{H}} + \sum_{i=1}^{N} c_i p_i V(y_i, h_i) \tag{OPT1}$$

where $h_i = \kappa(\boldsymbol{w}, \boldsymbol{x}_i) + b$ with $\kappa(\cdot,\cdot)$ a Mercer kernel function. The function V is the Hinge loss, i.e., $V(y_i, h_i) = \max(0, 1 - y_i h_i)$. We call $\boldsymbol{p} \triangleq [p_1, \cdots, p_N]^T \in \mathbb{P}$ the auxiliary variable of the problem, and assume its feasible set $\mathbb{P}$ to be convex. note that with $\boldsymbol{p}$ fixed, the inner problem resembles traditional large margin learning. Depending on the context, the auxiliary variable $\boldsymbol{p}$ can be regarded as hidden states or probability assignments for loss terms. We focus on (OPT1) in this work, because many large margin learning variations, including S^{3}VM, latent SVM, robust SVM, etc., can be rewritten in this form. The following is an example of such reformulation.

Example 1 *Consider the learning objective of Semi Supervised Support Vector Machine (S^{3}VM):*

$$\min_{\boldsymbol{w},b,\widehat{\boldsymbol{y}}_u} \frac{1}{2}||\boldsymbol{w}||^2_{\mathcal{H}} + C_1 \sum_{i=1}^{l} V(y_i, h_i) + C_2 \sum_{i=l+1}^{n} V(\widehat{y}_i, h_i)$$

where l is the number of labeled samples and $n - l$ unlabeled samples are included in the loss with "tentative" label $\widehat{\boldsymbol{y}}_u$, which constitute additional variables to minimize over. Interestingly, the learning objective has the following equivalent form:

$$\min_{\boldsymbol{w},b} \min_{\boldsymbol{p}} \frac{1}{2}||\boldsymbol{w}||^2_{\mathcal{H}} + C_1 \sum_{i=1}^{l} V(y_i, h_i) + C_2 \sum_{i=l+1}^{n} [p_i V(1, h_i) + (1 - p_i) V(-1, h_i)]$$

The equivalence is due to the fact that minimizing over p_i will cause all its mass to concentrate on the smaller of $V(1, h_i)$ and $V(-1, h_i)$. Formally for any variables $\xi_1, \cdots, \xi_M$ we have $\min_m\{\xi_1, \cdots, \xi_M\} = \min_{\boldsymbol{p}\in\mathbb{S}^M} \sum_{m=1}^{M} p_m \xi_m$, where $\mathbb{S}^M$ is the simplex in $\mathbb{R}^M$. Due to strict feasibility and biconvexity in $(\boldsymbol{w}, b)$ and $\boldsymbol{p}$, we can exchange the order of minimization and obtain an equivalent form similar to (OPT1). The variable p_i is the "probability" of $\widehat{y}_i = 1$.

Many other learning variations could be rewritten in a similar way[1]. Observing that the inner problem of OPT1 is convex quadratic with fixed $\boldsymbol{p}$, we replace it with its dual problem and cast OPT1 into

$$\max_{\boldsymbol{p}\in\mathbb{P}} \min_{\boldsymbol{\alpha}\in\mathbb{A}(\boldsymbol{p})} \mathcal{J}(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i,j} \alpha_i y_i \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) y_j \alpha_j - \sum_i \alpha_i$$
$$\text{where } \mathbb{A}(\boldsymbol{p}) = \left\{\boldsymbol{\alpha} \mid 0 \le \alpha_i \le c_i p_i \ \forall i,\ \boldsymbol{y}^T\boldsymbol{\alpha} = 0\right\} \tag{OPT2}$$

In the above equivalent formulation, we can view the inner optimization as minimizing a quadratic function subject to polyhedron constraints that are *parametrized* by the auxiliary variable $\boldsymbol{p}$. Assuming the kernel matrix $\boldsymbol{K}$, defined by $\boldsymbol{K}_{i,j} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, is strictly positive[2], then the optimum is unique by strict convexity, and the solution $\boldsymbol{\alpha}^*$ is a function of $\boldsymbol{p}$. Ideally, if one can write out the functional dependence explicitly, OPT2 is essentially $\max_{\boldsymbol{p}\in\mathbb{P}} \mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$, which minimizes over the "parameters" $\boldsymbol{p}$ of the inner problem. In the terminology of operational research and optimization, the task of analyzing the dependence of an optimal solution on multiple parameters is called parametric programming. Inspired by this new view of OPT2 (and thence OPT1), our solution strategy is: Firstly, determining the functional $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$ by parametric analysis, and then minimizing over $\boldsymbol{p} \in \mathbb{P}$ by exploiting the unique property of $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$.

Note that the first step in effect involves a convex quadratic parametric programming (CQPP), which has been addressed in optimization and control community for sensitivity analysis and explicit controller design (Tondel, Johansen, and Bemporad 2003) (Wachsmuth 2013). Moreover, the study of solution path algorithms in our field (Hastie et al. 2004) (Karasuyama and Takeuchi 2011) can also be regarded as special cases of CQPP. Nonetheless, existing work on CQPP is technically insufficient, because (1) Due to the presence of the constraint $\boldsymbol{\alpha}^T\boldsymbol{y} = 0$, the problem at hand corresponds to a "degenerate" case for which existing solution is still lacking. (2) Some important properties of the parametric solution, specifically its geometric structure, are not entirely revealed in prior works.

In the next section, we target the the inner minimization for parametric analysis. Our results not only provide the analytical form of the solution in critical regions (defined later), but also demonstrate that the overall learning problem (OPT2) is equivalent to a convex maximization.

[1] More examples of reformulation are given in the supplementary material.

[2] Then the induced matrix $\boldsymbol{Q} \triangleq \boldsymbol{K} \circ \boldsymbol{y}\boldsymbol{y}^T$ is also strictly positive, hence the optimization is strictly convex. For situations in which $\boldsymbol{K}$ is only positive semidefinite, a decomposition technique detailed in the supplementary material, can be used to reduce the problem to the strictly positive case.

3 Deriving the Equivalent Convex Maximization Problem

To begin with, the inner minimization is rewritten in a more compact form:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \mathcal{J}(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{Q}\boldsymbol{\alpha} - \mathbb{1}^T\boldsymbol{\alpha} \\ \text{subject to} \quad & \boldsymbol{C}^\alpha \boldsymbol{\alpha} \leq \boldsymbol{C}^p \boldsymbol{p} + \boldsymbol{C}^0, \quad \boldsymbol{y}^T\boldsymbol{\alpha} = 0. \end{aligned} \tag{IO}$$

where $Q_{ij} = y_i \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) y_j$, and $\boldsymbol{C}^\alpha$, $\boldsymbol{C}^p$ and $\boldsymbol{C}^0$ are constant matrices encapsulating the constraints.

A Mild Sufficient Condition for Existence We first demonstrate that, interestingly, a mild sample partition condition is sufficient for the existence and uniqueness of the parametric solution of (IO).

Definition 1 ***(Active Constraint)*** *After a solution of (IO) has been obtained as $\boldsymbol{\alpha}^*(\boldsymbol{p})$. The i^{th} row of the constraint is said to be active at $\boldsymbol{p}$, if $\boldsymbol{C}_i^\alpha \boldsymbol{\alpha}^*(\boldsymbol{p}) = \boldsymbol{C}_i^p \boldsymbol{p} + \boldsymbol{C}_i^0$, and inactive if $\boldsymbol{C}_i^\alpha \boldsymbol{\alpha}^*(\boldsymbol{p}) < \boldsymbol{C}_i^p \boldsymbol{p} + \boldsymbol{C}_i^0$. We denote the index set of active inequalities by $\mathcal{A}$, and inactive ones by $\mathcal{A}^C$. We use $\boldsymbol{C}_{\mathcal{A}}^\alpha$ to represent row selection of matrix $\boldsymbol{C}^\alpha$, i.e, $\boldsymbol{C}_{\mathcal{A}}^\alpha$ contains rows of $\boldsymbol{C}^\alpha$ whose index is in $\mathcal{A}$.*

Definition 2 ***(Partition of Samples)*** *Based on the value of α_i at optimal, the i^{th} sample is called:*

- *Non-support vectors, denoted by $i \in \mathcal{O}$, if $\alpha_i^* = 0$.*
- *Unbounded support vectors, denoted by $i \in \mathcal{S}_u$ if we have strictly $0 < \alpha_i^* < c_i p_i$.*
- *Bounded support vectors, denoted by $\mathcal{S}_b$, if $\alpha_i^* = c_i p_i$.*

Definition 3 ***(Non-degeneracy by Sample Partition)*** *We say that a solution of (IO) is non-degenerate if the unbounded support vector set $\mathcal{S}_u$ contains at least one $\{i \mid y_i = +1\}$ and at least one $\{i' \mid y_i' = -1\}$*

Now we connect non-degeneracy, defined as a sample partition property of large margin learning, to the existence and uniqueness of the parametric solution.

Lemma 1 *If the solution $\boldsymbol{\alpha}^*$ of (IO) is non-degenerate, then*

- *The matrix $\boldsymbol{H} \triangleq \frac{\boldsymbol{Q}^{-1}\boldsymbol{y}\boldsymbol{y}^T\boldsymbol{Q}^{-1}}{\boldsymbol{y}^T\boldsymbol{Q}^{-1}\boldsymbol{y}} - \boldsymbol{Q}^{-1}$ is negative semidefinite, and $\boldsymbol{R} \triangleq \boldsymbol{C}_{\mathcal{A}}^\alpha \boldsymbol{H} \boldsymbol{C}_{\mathcal{A}}^{\alpha T}$ is strictly negative definite, hence is invertible.*
- *The parametric solution $\boldsymbol{\alpha}^*(\boldsymbol{p})$ exists and is unique.*

Remark The invertibility of the matrix guarantees the uniqueness of the Lagrangian multipliers of (IO) and hence the existence and uniqueness of the parametric solution. The non-degeneracy condition is indeed a mild requirement: in fact in large margin learning formalism, the unbounded support vectors are essentially the sample points that lie on the decision boundaries, constructing the normal vector and the interception of the hyperplane. In practice to have meaningful classification this condition is a necessity and is expected to be satisfied.

Local Explicit Form of the Parametric Optimality With the previous definitions and Lemma 1, the following theorem provides the explicit form of $\boldsymbol{\alpha}^*(\boldsymbol{p})$, as well as explicit "critical regions" in which the dependence stands.

Theorem 1 *Assume that the solution of (IO) is non-degenerate and induces a set of active and inactive constraints $\mathcal{A}$ and $\mathcal{A}^C$, respectively. With $\boldsymbol{H}$, $\boldsymbol{R}$ defined previously and $\boldsymbol{T} \triangleq \boldsymbol{H}(\boldsymbol{C}_{\mathcal{A}}^\alpha)^T$, $\boldsymbol{v} \triangleq \boldsymbol{C}_{\mathcal{A}}^\alpha \boldsymbol{H} \mathbb{1}$, we have*
(1) The optimal solution is a continuous piecewise affine function of $\boldsymbol{p}$. And in the critical region defined by

$$\begin{cases} \boldsymbol{R}^{-1}(\boldsymbol{C}_{\mathcal{A}}^p \boldsymbol{p} + \boldsymbol{C}_{\mathcal{A}}^0 + \boldsymbol{v}) \geq 0 \\ \boldsymbol{C}_{\mathcal{A}^C}^p \boldsymbol{p} + \boldsymbol{C}_{\mathcal{A}^C}^0 - \boldsymbol{C}_{\mathcal{A}^C}^\alpha \boldsymbol{T}\boldsymbol{R}^{-1}(\boldsymbol{C}_{\mathcal{A}}^p \boldsymbol{p} + \boldsymbol{C}_{\mathcal{A}}^0 + \boldsymbol{v}) \geq 0 \end{cases} \tag{1}$$

the optimal solution $\boldsymbol{\alpha}^*$ of (IO) admits a closed form

$$\boldsymbol{\alpha}^*(\boldsymbol{p}) = \boldsymbol{T}\boldsymbol{R}^{-1}(\boldsymbol{C}_{\mathcal{A}}^p \boldsymbol{p} + \boldsymbol{C}_{\mathcal{A}}^0 + \boldsymbol{v}) \tag{2}$$

*(2) The optimal objective $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$ is a **continuous** piecewise quadratic (**PWQ**) function of $\boldsymbol{p}$.*

Remark The theorem indicates that each time the inner optimization (IO) is solved, full information in a well-defined neighborhood (critical region) can be retrieved as a function of the auxiliary variable. Hence one can efficiently calculate the closed form optimal solution and its gradient in that region, without having to solve (IO) again. (2) shows that $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$ is continuous but non-smooth.

Global Structure of the Optimality Recall that our goal is to solve $\max_{\boldsymbol{p}\in\mathbb{P}} \ \mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$. In this part, we show that the problem is equivalent to convex maximization by revealing several important geometric properties of $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$ as a function of $\boldsymbol{p}$.

Theorem 2 *Still assuming non-degeneracy, then*
*(1) There are finite number of critical regions $CR_1, \cdots, CR_{N_r}$ which constitute a **partition** of the feasible set of $\boldsymbol{p}$, i.e., each feasible $\boldsymbol{p}$ belongs to one and only one critical region.*
*(2) $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$ is a globally **convex** function of $\boldsymbol{p}$, and is **almost everywhere differentiable**.*
*(3) $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$ is **difference-definite**, i.e., the differences between its expressions on neighboring polyhedron critical regions have positive or negative semidefinite Hessian.*
(4) Let the common boundary of any two neibouring critical regions CR_i and CR_j be $\boldsymbol{a}^T\boldsymbol{p} + b$, then there exist a scalar β and a constant c, such that $\mathcal{J}_i(\boldsymbol{\alpha}^*(\boldsymbol{p})) = \mathcal{J}_j(\boldsymbol{\alpha}^*(\boldsymbol{p})) + \left[\boldsymbol{a}^T\boldsymbol{p} + b\right]\left[\beta \boldsymbol{a}^T\boldsymbol{p} + c\right]$.

Remark Although the number of critical regions is finite, in the worst case it could be exponential to the dimension of $\boldsymbol{p}$. Hence one cannot solve $\max_{\boldsymbol{p}\in\mathbb{P}} \ \mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$ by naively enumerating all possible critical regions. The globally convex PWQ property of $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{\theta}))$ revealed by (2) is critical: now that the class of learning problem formulated in (OPT1) is equivalent to maximizing a non-smooth convex function, which is well known to be *NP-hard*. Fortunately, we will show in next section that there exists an optimality condition that can be exploited to design efficient global optimization algorithms. Lastly, (3) and (4) imply that the expressions of $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$ on neighboring critical regions cannot be arbitrary, but is to some extent bounded. Those properties can be further harnessed for solution approximation.

4 Global Optimality Condition and Parametric Dual Maximization

To ease the notation, we hide the intermediate variable and denote

$$\mathcal{F}(\boldsymbol{p}) \triangleq \mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p})) \tag{3}$$

then (OPT2) becomes $\max_{\boldsymbol{p}\in\mathbb{P}} \mathcal{F}(\boldsymbol{p})$. From the properties of $\mathcal{F}(\boldsymbol{p})$, or $\mathcal{J}(\boldsymbol{\alpha}^*(\boldsymbol{p}))$, given in Theorem 1 and Theorem 2, we know that the problem is in effect a *convex piece-wise quadratic maximization*. In this section, we propose a global optimization algorithm based on an optimality condition and a level set approximation technique.

A Global Optimality Condition Several global optimality conditions for maximizing convex function, particularly convex quadratic functions, have been proposed before (Tsevendorj 2001) (Georgiev, Chinchuluun, and Pardalos 2011). In this work, we adapt a version of Strekalovsky's condition for non-smooth case. First of all, the notion of level set is defined as the set of variables that produce the same function values, i.e.,

Definition 4 *The level set of the function $\mathcal{F}$ at $\boldsymbol{p}$ is defined by*

$$E_{\mathcal{F}(p)} = \{q \in \mathbb{R}^n \mid \mathcal{F}(q) = \mathcal{F}(p)\}$$

A sufficient and necessary condition for a point $\boldsymbol{p}^*$ to be the global maximizer of $\mathcal{F}(\boldsymbol{p})$ reads,

Theorem 3 *$\boldsymbol{p}^*$ is a global optimal solution of the problem $\max_{\boldsymbol{p}\in\mathbb{P}} \mathcal{F}(\boldsymbol{p})$, if and only if for all $\boldsymbol{p} \in \mathbb{P}$, $\boldsymbol{q} \in E_{\mathcal{F}(\boldsymbol{p}^*)}$, $g(\boldsymbol{q}) \in \partial\mathcal{F}(\boldsymbol{q})$, we have*

$$(\boldsymbol{p} - \boldsymbol{q})^T g(\boldsymbol{q}) \leq 0 \tag{4}$$

where $\partial\mathcal{F}(\boldsymbol{q})$ is the set of subgradients of $\mathcal{F}$ at $\boldsymbol{p}$.

By virtue of Theorem 3, we can verify the optimality of any point $\boldsymbol{p}$ by solving

$$\Delta(\boldsymbol{p}) \triangleq \max_{\substack{\boldsymbol{q}\in E_{\mathcal{F}(\boldsymbol{p})},\ \boldsymbol{p}'\in\mathbb{P} \\ g(\boldsymbol{q})\in\partial\mathcal{F}(\boldsymbol{q})}} (\boldsymbol{p}' - \boldsymbol{q})^T g(\boldsymbol{q}) \tag{5}$$

and checking if $\Delta(\boldsymbol{p}) \leq 0$. We call the above maximization the *auxiliary problem* at $\boldsymbol{p}$. The major difficulty is that the level set $E_{\mathcal{F}(\boldsymbol{p})}$ is hard to calculate explicitly. Next, we study solution method for (5) by approximating the level set with a collection of representative points.

Approximate Level Set

Definition 5 *Given a user specified approximation degree m, the approximation level set for $E_{\mathcal{F}(\boldsymbol{p})}$ is defined by*

$$A^m_{\boldsymbol{p}} = \left\{\boldsymbol{q}^1, \boldsymbol{q}^2, \cdots, \boldsymbol{q}^m \mid \boldsymbol{q}^i \in E_{\mathcal{F}(\boldsymbol{p})}\ i = 1, 2, \cdots, m\right\}$$

Consider solving the auxiliary problem approximately by replacing $E_{\mathcal{F}(\boldsymbol{p})}$ with $A^m_{\boldsymbol{p}}$, then for each $\boldsymbol{q}^i$, (5) becomes

$$\max_{\boldsymbol{p}\in\mathbb{P},\ g(\boldsymbol{q}^i)\in\partial\mathcal{F}(\boldsymbol{q}^i)} (\boldsymbol{p} - \boldsymbol{q}^i)^T g(\boldsymbol{q}^i) \tag{6}$$

Since $\mathcal{F}(\boldsymbol{p})$ is almost everywhere differentiable, in most cases $g(\boldsymbol{q}^i)$ is unique and equals to the gradient $\nabla\mathcal{F}(\boldsymbol{q}^i)$. Then the auxiliary problem is a simple *linear program*. In the cases when $\boldsymbol{q}^i$ is on the boundary of critical regions, $\partial\mathcal{F}(\boldsymbol{q}^i)$ becomes a convex set, and the auxiliary problem becomes a bilinear program. General bilinear program is hard, but fortunately (6) has disjoint feasible sets, and one can show that

Algorithm 1 Parametric Dual Maximization

Choose $\boldsymbol{p}^{(0)} \in \mathbb{P}$; set $k = 0$; compute $\boldsymbol{p}_*$ with subgradient descent.
while $k \leq$ iter_max **do**
  Starting from $\boldsymbol{p}^{(k)}$, find a local maximizer $\boldsymbol{r}^{(k)} \in \mathbb{P}$ with a local solver.
  Construct $A^m_{\boldsymbol{r}^{(k)}}$ at $\boldsymbol{r}^{(k)}$ by (9) (10); Solve (IO) if a new critical region is encountered, otherwise use (2).
  for $\boldsymbol{q}^i \in A^m_{\boldsymbol{r}^{(k)}}$ **do**
    for $\boldsymbol{g}^j \in V(\partial\mathcal{F}(\boldsymbol{q}^i))$ **do**
      Solve linear programming $\boldsymbol{u}_{ij} = \text{argmax}_{\boldsymbol{p}\in\mathbb{P}}\ (\boldsymbol{p} - \boldsymbol{q}^i)^T \boldsymbol{g}^j$
    end for
    Let $j^* = \text{argmax}_j\{\boldsymbol{u}_{ij}\}$; $(\boldsymbol{u}^i, \boldsymbol{s}^i) = (\boldsymbol{u}_{ij^*}, \boldsymbol{g}^{j^*})$;
  end for
  Let $i^* = \text{argmax}_i\{(\boldsymbol{u}^i - \boldsymbol{q}^i)^T \boldsymbol{s}^i\}$; $\boldsymbol{u}^{(k)} = \boldsymbol{u}^{i^*}$;
  if $(\boldsymbol{u}^{i^*} - \boldsymbol{q}^{i^*})^T \boldsymbol{s}^{i^*} > 0$ **then**
    Set $\boldsymbol{p}^{(k+1)} = \boldsymbol{u}^{(k)}$; $k = k + 1$; # improvement found
  else
    Terminate and output $\boldsymbol{p}^{(k)}$; # optimality checked
  end if
  Collecting explored critical region and explicit forms given in (2)(1).
end while

Proposition 1 *Problem (6) is equivalent to*

$$\max_{\boldsymbol{p}\in\mathbb{P}} \left\{ \max_{g(\boldsymbol{q}^i)\in V(\partial\mathcal{F}(\boldsymbol{q}^i))} (\boldsymbol{p} - \boldsymbol{q}^i)^T g(\boldsymbol{q}^i) \right\} \tag{7}$$

which indicates that the optimal solution to (6) must be on the vertex of the feasible polyhedron. As such, (6) can be expanded into a set of linear programs, each of which is substantiated by an element in $A^m_{\boldsymbol{p}}$ and a vertex of $\partial\mathcal{F}(\boldsymbol{q}^i)$.

The PDM Algorithm With the approximate auxiliary problem solved, we can immediately determine if an *improvement* can be made at the current $\boldsymbol{p}$. More specifically, let $\{(\boldsymbol{u}^i, \boldsymbol{s}^i),\ i = 1, \cdots, m\}$ be the solution of (6) on $A^m_{\boldsymbol{p}}$, i.e.,

$$(\boldsymbol{u}^i - \boldsymbol{q}^i)^T \boldsymbol{s}^i = \max_{\boldsymbol{p}\in\mathbb{P},\ g(\boldsymbol{q}^i)\in V(\partial\mathcal{F}(\boldsymbol{q}^i))} (\boldsymbol{p} - \boldsymbol{q}^i)^T g(\boldsymbol{q}^i) \tag{8}$$

and define $\Delta(A^m_{\boldsymbol{p}}) = \max_{i=1,\cdots,m} (\boldsymbol{u}^i - \boldsymbol{q}^i)^T \boldsymbol{s}^i$. Then with the convexity of $\mathcal{F}$ we have

Proposition 2 *For any $\boldsymbol{p} \in \mathbb{P}$, if there exist $\boldsymbol{q}^i \in A^m_{\boldsymbol{p}}$, $g(\boldsymbol{q}^i) \in V(\partial\mathcal{F}(\boldsymbol{q}^i))$, and $\boldsymbol{u}^i$ defined in (8), such that $(\boldsymbol{u}^i - \boldsymbol{q}^i)^T g(\boldsymbol{q}^i) > 0$, then we must have $\mathcal{F}(\boldsymbol{u}^i) > \mathcal{F}(\boldsymbol{p})$.*

Now the remaining work is to construct the approximate level set given the current $\boldsymbol{p}$ and the degree m. The following lemma shows that this is possible if a global minimizer is available.

Lemma 2 *Let the global **minimizer** of $\mathcal{F}(\boldsymbol{p})$ be $\boldsymbol{p}_*$, then for $\boldsymbol{p} \neq \boldsymbol{p}_*$ and $\boldsymbol{h} \in \mathbb{R}^n$, there exist a **unique** positive scalar γ, such that $\boldsymbol{p}_* + \gamma\boldsymbol{h} \in E_{\mathcal{F}(\boldsymbol{p})}$.*

With this guarantee, we write approximate level set by

$$A^m_{\boldsymbol{p}} = \left\{\boldsymbol{q}^1, \boldsymbol{q}^2, \cdots, \boldsymbol{q}^m \mid \boldsymbol{q}^i = \boldsymbol{p}_* + \gamma_i \boldsymbol{h}^i \in E_{\mathcal{F}(\boldsymbol{p})}\right\} \tag{9}$$

To explore directions for improvement, a natural choice of $\boldsymbol{h}$ is a set of orthogonal basis. Specifically, we could start with a random $\boldsymbol{h}^1$ and use Gram-Schmidt algorithm to extend it to m orthogonal basis. For each $\boldsymbol{h}^i$, the corresponding γ_i is found by solving:

$$\Phi(\gamma_i) \triangleq \mathcal{F}(\boldsymbol{p}_* + \gamma_i \boldsymbol{h}^i) - \mathcal{F}(\boldsymbol{p}) = 0 \tag{10}$$

As stated in Lemma 2, the above function has a unique root, which can be computed efficiently with line searching method. To obtain the global minimizer, we have to solve $\boldsymbol{p}_* = \text{argmin}\mathcal{F}(\boldsymbol{p})$, which is a convex minimization problem. Using Theorem 1, we show (in supplementary material) that a sub-gradient descent method with T iterations converges to the global minimum within $O(1/\sqrt{T})$.

Organizing all building blocks developed so far, we summarize the PDM procedure in Algorithm 1. Given the current solution $\boldsymbol{p}^{(k)}$, the algorithm first tries to improve it with existing methods such as AO, CCCP, SGD, etc. After finding a local solution $\boldsymbol{r}^{(k)}$, the approximate level set $A^m_{\boldsymbol{r}^{(k)}}$ is obtained by solving (10) and constructing (9). With $A^m_{\boldsymbol{r}^{(k)}}$ and the current sub-gradient, one or several linear program is solved to pick up the vector $\boldsymbol{u}^{(k)}$ that maximizes the condition (4) of Theorem 3. If this maximal value, i.e., $\Delta(A^m_{\boldsymbol{p}})$, is greater than 0, then by Proposition 2, $\boldsymbol{u}^{(k)}$ must be a strictly improved solution compared to $\boldsymbol{r}^{(k)}$. As such, the algorithm continues with $\boldsymbol{p}^{(k+1)} = \boldsymbol{u}^{(k)}$. Otherwise if $\Delta(A^m_{\boldsymbol{p}}) \leq 0$, the algorithm terminates since no improvement could be found at the current point with the user specified approximation degree. For convergence, we have

Theorem 4 *Algorithm I generates a sequence* $\{\boldsymbol{p}^{(1)}, \cdots, \boldsymbol{p}^{(k)}, \cdots\}$ *having non-decreasing function values. The sequence converges to an approximate maximizer of* $\mathcal{F}(\boldsymbol{p})$ *in a finite number of steps.*

In each iteration, we only have to solve $m|V(\partial\mathcal{F}(\boldsymbol{q}^i))|$ linear programs, and in most cases $|V(\partial\mathcal{F}(\boldsymbol{q}^i))| = 1$ due to the almost everywhere differentiability shown in Theorem 2. When constructing the approximiate level set, we need to solve at most m convex quadratic programs (IO)s, which seems computationally expensive. However, note that this problem resembles the classic SVM dual, hence a variety of existing methods can be reused for acceleration (Chang and Lin 2011). Moreover, by virtue of the optimality structure revealed in Theorem 1 and 2, a list of explored critical regions and the corresponding explicit optimalities can be stored. If the current $\boldsymbol{p}$ is on this list, all information could be retrieved in an explicit form, and there is no need to solve the quadratic problem again. To further accelerate the algorithm, one can "enlarge" critical regions. See supplementary material for a discussion.

5 Experiments

In this section, we report optimization and generalization performance of PDM for the training of S^{3}VM and Latent SVM (LSVM). More results and a Matlab implementation could be found online.

Datasets and Experiment Setup Details about the datasets are listed in Table 1. For S^{3}VM, we report results on four popular data sets for semi-supervised learning, i.e., *2moons* (D1), *coil* (D2), *robot* (D3) and *2spiral* (D4, with simulator). In each experiment, 60% of the samples are used for training, in which only a small portion are assumed to be labeled samples. 10% of the data are used as a validation set for choosing hyperparameters. With the remaining 30%, we evaluate the generalization performance. For LSVM we adopt the same training, validation and testing partition on *Vowel* (D5), *Music* (D6), *Bank* (D7) and *Wave* (D8, with simulator) data sets. To create a latent data structure, we assume only grouped binary labels are known.

Table 1: Data sets. D4-D3 for S^{3}VM and D5-D8 for LSVM

Data set	ID	# classes	# samples	# features	labeled
2moons	D1	2	200	2	2
Coil	D2	3	216	1024	6
Robot	D3	4	2456	25	40
2spiral	D4	2	100000	2	4
Vowel	D5	10	990	11	grouped
Music	D6	10	2059	68	grouped
Bank	D7	9	7166	649	grouped
Wave	D8	30	100000	40	grouped

The Gaussian kernel $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \exp\left\{\|\boldsymbol{x} - \boldsymbol{y}\|^2/2\sigma^2\right\}$ is used for all experiments. Following model selection suggestions (Chapelle, Sindhwani, and Keerthi 2008)(Felzenszwalb et al. 2010), best hyperparamter combination C_1, C_2, σ^2 are chosen with cross validation from $C_1 \in \{10^{0:0.5:3}\}$, $\sigma^2 \in \{(1/2)^{-3:1:3}\}$ and $C_2 \in \{10^{-8:1:0}\}$ for S^{3}VM and $C_2 \in \{10^{-4:1:4}\}$ for LSVM. A simple gradient ascent is used as the local minimizer for PDM. All experiments are conducted on a workstation with Dual Xeon x5687 CPUs and 72GB memory.

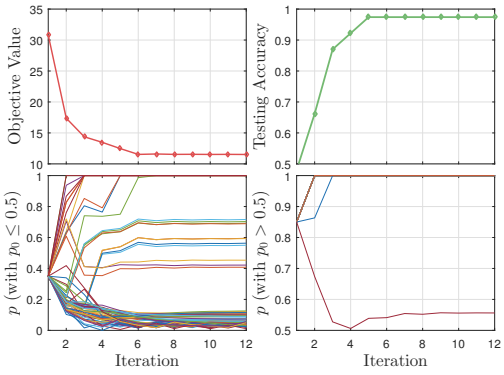

Figure 1: PDM in each iteration for S^{3}VM training. Randomized initiation; $m = 20$; D1 dataset

Small-scale Demo To get more intuition on how PDM works, we use PDM to train S^{3}VM on the D1 dataset, and plot the iterative evolution of objective function ($-\mathcal{J}$), testing accuracy and the values of $\boldsymbol{p}$ in Figure 1. The approximation level m is set to $0.1length(p) = 20$, and initial $p^{(0)}$ is chosen randomly. We observe that PDM converges within 12 iterations (top left subfigure). The testing accuracy increases from 48% to above 98% (top right subfigure), showing improvements in both optimization and generalization performance. Moreover, the auxiliary variable $\boldsymbol{p}$ approaches global optimum even with random initial values (bottom subfigures). Note that in this process, a total number of 36 (IO)s are solved and about 2/3 of the critical regions have been reused more than once.

Table 2: Normalized objective value (OPT1. First row for each dataset. The lower the better). Time usage (Second row for each dataset. $s = seconds; h = hours$)

	Data	GD	CCCP	AO	LCS	IA	BB	PDM1	PDM2
S^3VM	D1	2.39	2.82	4.83	5.55	1.79	**1.00**	1.03	**1.00**
		1.7s	6.2s	2.7s	6.7s	3.4s	210s	16s	35s
	D2	3.74	3.92	3.46	4.98	2.35	**1.00**	1.19	1.03
		5.3s	6.8s	**4.3s**	7.9s	5.6s	362s	43s	83s
	D3	3.95	4.23	3.48	6.96	2.85	*	1.11	**1.00**
		33s	56s	28s	43s	**27s**	*	231s	489s
	D4	6.98	4.91	4.90	6.16	4.22	*	1.31	**1.00**
		0.19h	0.41h	0.33h	0.37h	0.46h	*	1.4h	2.7h
LSVM	D5	4.45	5.31	4.85	4.09	*	*	1.13	**1.00**
		26s	54s	33s	68s	*	*	209s	451s
	D6	6.51	5.34	4.77	6.82	*	*	1.28	**1.00**
		63s	90s	72s	101s	*	*	468s	997s
	D7	6.78	7.69	4.17	6.22	*	*	1.26	**1.00**
		326s	371s	**263s**	477s	*	*	1217s	2501s
	D8	10.2	5.16	6.35	7.57	*	*	1.54	**1.00**
		0.23h	0.73h	0.66h	0.93h	*	*	2.5h	4.8h

Optimization and Generalization Performance We next compare PDM with different optimization methods in terms of their optimization and generalization performance. The algorithms considered for S^3VM training are: Gradient Descent (GD) in (Chapelle and Zien 2005), CCCP in (Collobert et al. 2006), Alternating Optimization (AO) in (Sindhwani, Keerthi, and Chapelle 2006), Local Combinatorial Search (LCS) in (Joachims 1999), Infinitesimal Annealing (IA) in (Ogawa et al. 2013), Branch and Bound (BB) in (Chapelle, Sindhwani, and Keerthi 2006). The algorithms included for LSVM are GD in (Kantchelian et al. 2014), CCCP in (Yu and Joachims 2009), AO in (Dundar et al. 2008), adapted LCS in (Joachims 1999). The proposed PDM is tested with two versions by setting the approximation degree $m = 0.1length(p)$ (PDM1) and $m = 0.2length(p)$ (PDM2).

In Table 2, objective function values of OPT1 (normalized by the smallest one) are shown in the upper row, and the corresponding computation times are given in the second row for each data. Note that although BB provides exact global optimum for small data set D1 and D2, it runs out of memory (72GB!) for other datasets due to the exponential growth of its search tree. On the other hand, PDM1&2 provides a near optimal solution to BB with much less time and space usage. For larger data sets (D4-D8) on which BB can not be executed, PDM outperforms all the other local optimization methods: We observe that PDM achieves a significantly improved objective value, and the runner up is at least 2.8 times larger. Although the running time is longer than local methods, PDM is still scalable (D4 & D8 have 10^5 samples), hence can be carried out for large scale problems.

In Table 3, we compare the generalization performance of different algorithms in terms of testing error rate. It appears clearly that the global optimal solution provided by BB and PDM has excellent generalization error rate, while other local optimization methods perform much worse, and even fail completely (e.g., on D1, D2, D4, D8). This observation is consistent with previous findings (Chapelle, Sindhwani, and Keerthi 2006) (Chapelle, Sindhwani, and Keerthi 2008), justifying the extra computational overhead required to pursue the global optimum.

Table 3: Generalization Performance (error rates). Averaged over 10 random data partitions. Error rate greater than or close to 50% should be interpreted as "failed".

	Data	GD	CCCP	AO	LCS	IA	BB	PDM1	PDM2
S^3VM	D1	51.4	60.0	52.8	65.5	37.5	**0.0**	1.9	0.2
	D2	57.9	66.1	47.9	61.1	57.2	**0.0**	5.3	1.1
	D3	26.6	29.3	59.8	38.8	27.4	*	9.5	**3.3**
	D4	52.1	39.8	40.0	45.4	31.4	*	3.5	**2.0**
LSVM	D5	15.8	16.2	13.5	9.9	*	*	2.5	**1.7**
	D6	39.8	43.7	40.8	39.4	*	*	12.1	**7.6**
	D7	20.0	19.4	19.8	22.5	*	*	8.9	**5.1**
	D8	53.1	36.7	39.7	46.2	*	*	19.9	**13.1**

Choice of Approximation Degree m Comparing PDM1 and PDM2 in Table 2&3, we note that in general, increasing the approximation degree m will produce better optimization and generalization performance. To investigate the effect of m, we use PDM to train S^3VM on D3, and plot in Figure 2 the optimum value, testing accuracy, time and space usage as a function of m (from 80 to 650). It appears that further increasing m after some large enough value (e.g., 300 in Figure 2) only provide marginal improvement in both training and testing. Also, seeing that the computational time usage grows (slightly) super-linearly and that the space usage grows almost linearly, we suggest using an $m \in [0.1length(p), 0.2length(p)]$, a tradeoff between training/testing accuracy and computational overhead.

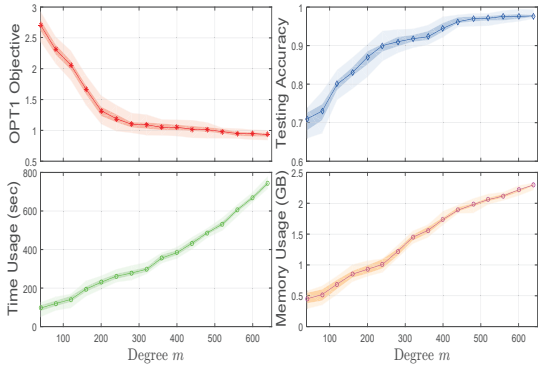

Figure 2: The effect of m for PDM. D3 Dataset; Average and CIs for 50 runs.

6 Conclusion

In this paper we propose a novel global optimization procedure, PDM, to solve a class of non-convex learning problem. Our parametric analysis reveals an entirely different perspective that this class of learning problems are equivalent to maximizing a convex PWQ function. We then develop the PDM algorithm based on a global optimality condition for non-smooth convex maximization. Experimental results justified the effectiveness of PDM regarding both optimization and generalization performance.

7 Acknowledgments

This research is funded by the Republic of Singapore's National Research Foundation through a grant to the Berkeley Education Alliance for Research in Singapore (BEARS) for the Singapore-Berkeley Building Efficiency and Sustainability in the Tropics (SinBerBEST) Program. BEARS has been established by the University of California, Berkeley as a center for intellectual excellence in research and education in Singapore.

References

Bei, T., and Cristianini, N. 2006. Semi-supervised learning using semi-definite programming. In *Semi-supervised Learning*. MIT Press. 177–186.

Bennett, K.; Demiriz, A.; et al. 1999. Semi-supervised support vector machines. *Advances in Neural Information processing systems* 368–374.

Bottou, L. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer. 177–186.

Chang, C.-C., and Lin, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2:27:1–27:27.

Chapelle, O., and Zien, A. 2005. Semi-supervised classification by low density separation. In *Proceedings of the tenth international workshop on artificial intelligence and statistics*, volume 1, 57–64.

Chapelle, O.; Chi, M.; and Zien, A. 2006. A continuation method for semi-supervised svms. In *Proceedings of the 23rd international conference on Machine learning*, 185–192. ACM.

Chapelle, O.; Sindhwani, V.; and Keerthi, S. S. 2006. Branch and bound for semi-supervised support vector machines. In *Advances in neural information processing systems*, 217–224.

Chapelle, O.; Sindhwani, V.; and Keerthi, S. S. 2008. Optimization techniques for semi-supervised support vector machines. *The Journal of Machine Learning Research* 9:203–233.

Collobert, R.; Sinz, F.; Weston, J.; and Bottou, L. 2006. Large scale transductive svms. *The Journal of Machine Learning Research* 7:1687–1712.

Dundar, M. M.; Wolf, M.; Lakare, S.; Salganicoff, M.; and Raykar, V. C. 2008. Polyhedral classifier for target detection: a case study: colorectal cancer. In *ICML*.

Felzenszwalb, P. F.; Girshick, R. B.; McAllester, D.; and Ramanan, D. 2010. Object detection with discriminatively trained part-based models. *PAMI, IEEE Transactions on* 32(9):1627–1645.

Georgiev, P. G.; Chinchuluun, A.; and Pardalos, P. M. 2011. Optimality conditions of first order for global minima of locally lipschitz functions. *Optimization* 60(1-2):277–282.

Hastie, T.; Rosset, S.; Tibshirani, R.; and Zhu, J. 2004. The entire regularization path for the support vector machine. *The Journal of Machine Learning Research* 5:1391–1415.

Joachims, T. 1999. Transductive inference for text classification using support vector machines. In *ICML*, volume 99, 200–209.

Kantchelian, A.; Tschantz, M. C.; Huang, L.; Bartlett, P. L.; Joseph, A. D.; and Tygar, J. 2014. Large-margin convex polytope machine. In *Advances in Neural Information Processing Systems*, 3248–3256.

Karasuyama, M., and Takeuchi, I. 2011. Suboptimal solution path algorithm for support vector machine. *ICML*.

Krishnamoorthy, B. 2008. Bounds on the size of branch-and-bound proofs for integer knapsacks. *Operations Research Letters* 36(1):19–25.

Ogawa, K.; Imamura, M.; Takeuchi, I.; and Sugiyama, M. 2013. Infinitesimal annealing for training semi-supervised support vector machines. In *Proceedings of the 30th International Conference on Machine Learning*, 897–905.

Park, J., and Boyd, S. 2015. A semidefinite programming method for integer convex quadratic minimization. *arXiv preprint arXiv:1504.07672*.

Ping, W.; Liu, Q.; and Ihler, A. 2014. Marginal structured svm with hidden variables. *arXiv preprint arXiv:1409.1320*.

Sindhwani, V.; Keerthi, S. S.; and Chapelle, O. 2006. Deterministic annealing for semi-supervised kernel machines. In *Proceedings of the 23rd international conference on Machine learning*, 841–848. ACM.

Tondel, P.; Johansen, T. A.; and Bemporad, A. 2003. An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica*.

Tsevendorj, I. 2001. Piecewise-convex maximization problems. *Journal of Global Optimization* 21(1):1–14.

Wachsmuth, G. 2013. On licq and the uniqueness of lagrange multipliers. *Operations Research Letters* 41(1):78–80.

Xu, L.; Crammer, K.; and Schuurmans, D. 2006. Robust support vector machine training via convex outlier ablation. In *AAAI*, volume 6, 536–542.

Yu, C.-N. J., and Joachims, T. 2009. Learning structural svms with latent variables. In *26th international conference on machine learning*, 1169–1176.

Yuille, A. L.; Rangarajan, A.; and Yuille, A. 2002. The concave-convex procedure (cccp). *Advances in neural information processing systems* 2:1033–1040.

Zhou, Y.; Hu, N.; and Spanos, C. J. 2016. Veto-consensus multiple kernel learning. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Zhou, Y.; Jin, B.; and Spanos, C. J. 2015. Learning convex piecewise linear machine for data-driven optimal control. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, 966–972. IEEE.