Infinitesimal Annealing for Training Semi-Supervised Support Vector Machines

Kohei Ogawa OGAWA.MLLAB.NIT@GMAIL.COM
Motoki Imamura IMAMURA.MLLAB.NIT@GMAIL.COM
Ichiro Takeuchi TAKEUCHI.ICHIRO@NITECH.AC.JP
Nagoya Institute of Technology, Nagoya, Japan

Masashi Sugiyama SUGI@CS.TITECH.AC.JP
Tokyo Institute of Technology, Tokyo, Japan

Abstract

The *semi-supervised support vector machine* (S^3VM) is a maximum-margin classification algorithm based on both labeled and unlabeled data. Training S^3VM involves either a combinatorial or non-convex optimization problem and thus finding the global optimal solution is intractable in practice. It has been demonstrated that a key to successfully find a good (local) solution of S^3VM is to gradually increase the effect of unlabeled data, à la *annealing*. However, existing algorithms suffer from the trade-off between the resolution of annealing steps and the computation cost. In this paper, we go beyond this trade-off by proposing a novel training algorithm that efficiently performs annealing with an *infinitesimal* resolution. Through experiments, we demonstrate that the proposed infinitesimal annealing algorithm tends to produce better solutions with less computation time than existing approaches.

1. Introduction

Semi-supervised learning, the paradigm of learning from labeled and unlabeled data, has been extensively studied in the last decade (Chapelle et al., 2006). The *semi-supervised support vector machine* (S^3VM) or *transductive SVM* (Vapnik & Sterin, 1977; Joachims, 1999) is one of the popular semi-supervised classification algorithms that inherits the large-margin concept of supervised SVMs (Boser et al., 1992; Vapnik, 1996; Cortes & Vapnik, 1995). The basic idea of S^3VM is to improve the supervised SVM solution (obtained only from labeled data) with the help of unlabeled data (Joachims, 1999). A key challenge for the success of S^3VM is to optimally control how strongly the effect of unlabeled data is incorporated into a classifier.

From an algorithmic point of view, S^3VM tries to maximize the margin over both labeled and unlabeled data, and this is cast as either a combinatorial optimization problem of assigning labels to unlabeled data (Joachims, 1999) or a non-convex optimization problem of maximizing the margin for unlabeled data (Collobert et al., 2006). As described in Chapelle et al. (2007), it is practically difficult to find the global optimal solution for large problems. For that reason, a great deal of effort has been made to obtain good sub-optimal solutions efficiently (Joachims, 1999; Sindhwani et al., 2006; Chapelle, 2007).

It was pointed out in Chapelle et al. (2008) that most of the successful S^3VM training algorithms proposed so far can actually be viewed as *annealing* (Korte & Vygen, 2000; Hromkovic, 2001; Kirkpatrick & Gelatt, 1983; Colorni et al., 1991). That is, starting from the original supervised SVM formulation, a sequence of sub-problems where the effect of unlabeled data is increasingly strengthned, is solved to obtain a final solution. However, in this annealing procedure, there is a trade-off between the number of annealing steps and the computation cost. Thus, enhancement of the annealing resolution is possible only at the expense of increasing the computation cost, which is a critical limitation in the current implementations of S^3VM.

The goal of this paper is to go beyond this trade-off: We propose a new training algorithm for S^3VM that ef-

Proceedings of the 30^{th} International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

ficiently performs annealing with an *infinitesimal* resolution. Technically, our algorithm can be regarded as a non-trivial extension of the *parametric programming* (Allgower & George, 1993; Best, 1996; Ritter, 1984; Efron & Tibshirani, 2004; Hastie et al., 2004; Takeuchi et al., 2009; Karasuyama et al., 2012), and it gives a path of *local* optimal solutions when the effect of unlabeled data is continuously increased. Interestingly, through the analysis of necessary and sufficient conditions for the local optimality of S^{3}VM, we find that the local solution path followed by the infinitesimal annealing steps is not continuous; a solution path actually contains a finite number of abrupt *jumps*. Our algorithm can exactly identify such jumps and trace the entire path of local optimal solutions. To the best of our knowledge, this is technically a novel contribution to the parametric programming community.

Through experiments, we demonstrate that our infinitesimal annealing algorithm tends to produce better solutions with less computation time than existing approaches.

2. Semi-Supervised SVM (S^{3}VM)

We review S^{3}VM (Joachims, 1999) here. Suppose that we are given labeled instances $\{(x_i, y_i)\}_{i\in\mathcal{L}}$ and unlabeled instances $\{x_i\}_{i\in\mathcal{U}}$, where $x_i \in \mathbb{R}^d$ is an input vector and $y_i \in \{-1, 1\}$ is a class label. The decision function

$$f(x) = b + w^\top \phi(x),$$

is learned in S^{3}VM, where ϕ is a feature map, w are the parameters to learn, and $^\top$ denotes the transpose. In S^{3}VM, the bias term b is usually fixed as $b = 2r - 1$ in order to satisfy a class-balance constraint of unlabeled instances, where $r = \frac{1}{|\mathcal{L}|}\sum_{i\in\mathcal{L}} \max(0, y_i)$. See Chapelle & Zien (2005) for details.

The problem of S^{3}VM training is interpreted as a combinatorial optimization problem (Joachims, 1999) or a non-convex optimization problem (Collobert et al., 2006). Below, we review both interpretations.

S^{3}VM as Combinatorial Problem: The basic idea is to learn the decision function and labels of unlabeled instances simultaneously to maximize the margin:

$$\min_{f,\hat{y}} \; J(f,\hat{y}) := \frac{1}{2}\|w\|_2^2 + C \sum_{i\in\mathcal{L}} [1 - y_i f(x_i)]_+ + C^* \sum_{i\in\mathcal{U}} [1 - \hat{y}_i f(x_i)]_+, \quad (1)$$

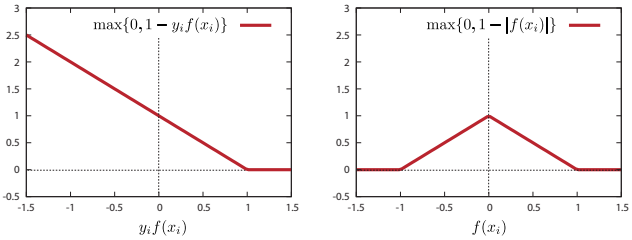

Figure 1. Loss functions for S^{3}VM. Left: Convex hinge loss for labeled instances. Right: Non-convex symmetric hinge loss for unlabeled instances.

where $\hat{y} \in \{-1, 1\}^{|\mathcal{U}|}$ is a vector of predicted labels of unlabeled instances and $[1 - z]_+$ is so-called the hinge loss function (see the left panel of Figure 1). C and C^* are regularization parameters for labeled and unlabeled instances, respectively. Because labeled instances are more reliable than unlabeled ones, they are chosen to satisfy $C^* \leq C$.

At the optimal solution of the minimization problem (1), the predicted labels should satisfy

$$\hat{y}_i f(x_i) \geq 0, i \in \mathcal{U} \quad (2)$$

because, if one of the predicted labels, $\hat{y}_i$, violates this condition, the objective function $J(f, \hat{y})$ can be strictly decreased by flipping it.

Introducing this condition, we can rewrite the S^{3}VM training criterion (1) as

$$\min_{\hat{y}} \left\{ \min_{f} J(f,\hat{y}) \text{ s.t. } (2) \right\}. \quad (3)$$

This formulation can be interpreted as a combinatorial optimization problem of finding the best predicted label vector $\hat{y}$ that minimizes $J(f, \hat{y})$ from all possible $2^{|\mathcal{U}|}$ candidates in $\{-1, 1\}^{|\mathcal{U}|}$.

S^{3}VM as Non-Convex Problem: If a predicted label $\hat{y}_i, i \in \mathcal{U}$ is chosen to satisfy (2), we can eliminate $\hat{y}_i$ by $\hat{y}_i f(x_i) = |f(x_i)|$. Then the optimization problem (1) can be rewritten as

$$\min_{f} \; \frac{1}{2}\|w\|_2^2 + C \sum_{i\in\mathcal{L}} [1 - y_i f(x_i)]_+ + C^* \sum_{i\in\mathcal{U}} [1 - |f(x_i)|]_+. \quad (4)$$

Because the loss for unlabeled instances, $[1 - |f(x)|]_+$, is non-convex as plotted in the right panel of Figure 1, (4) is a non-convex optimization problem.

3. Proposed Algorithm: S^{3}VM$^{\mathbf{path}}$

As explained in the previous section, S^{3}VM has either a combinatorial or a non-convex nature. Thus, the

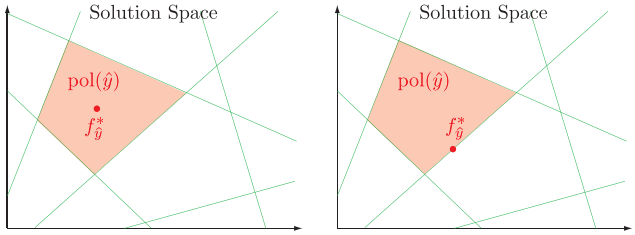

Figure 2. Illustration of S^{3}VM local optimal solutions. Each green line in the solution space represents $f(x_i) = 0, i \in \mathcal{U}$. If predicted labels $\hat{y}$ is given, a convex polytope pol($\hat{y}$) is defined by a set of these green lines. (Left) The conditionally optimal solution $f^*_{\hat{y}}$ in pol($\hat{y}$) is local optimal because it is located in the strict interior of pol($\hat{y}$). (Right) On the other hand, $f^*_{\hat{y}}$ is not local optimal because it is at the boundary of pol($\hat{y}$).

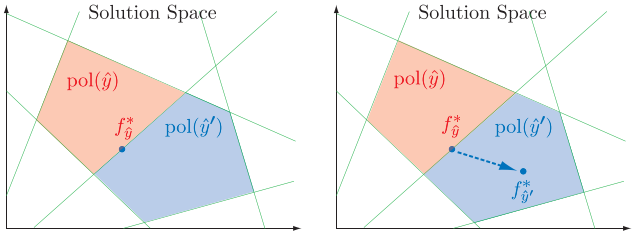

Figure 3. (Left) The conditionally optimal solution $f^*_{\hat{y}}$ is at a boundary of the convex polytope pol($\hat{y}$), which indicates that $f^*_{\hat{y}}$ is not a local optimal solution (see Theorem 4). By flipping some labels in $\hat{y}$, another convex polytope (blue one) is defined, and $f^*_{\hat{y}}$ is feasible also in this polytope. (Right) According to Theorem 5, $f^*_{\hat{y}}$ is not conditionally optimal in the adjacent polytope pol($\hat{y}'$), where $\hat{y}'$ is defined by (10). The conditionally optimal solution $f^*_{\hat{y}'}$ would be elsewhere in the blue polytope, and it is guaranteed to be strictly better than $f^*_{\hat{y}}$.

practical goal of existing S^{3}VM studies has been to develop an algorithm that can find a good local optimal solution (Joachims, 1999; Sindhwani et al., 2006; Chapelle, 2007).

As pointed out in Chapelle et al. (2008), these existing S^{3}VM algorithms utilize the concept of *annealing* either explicitly or implicitly to find a local optimal solution: Starting from the supervised SVM ($C^* = 0$), a sequence of sub-problems with increasing C^* is solved. In this annealing procedure, there is a trade-off between the number of annealing steps and the computation cost. This means that the annealing resolution can be enhanced only at the expense of increasing the computation cost, which is a critical limitation in the current implementations of S^{3}VM.

Our goal is to go beyond this limitation by developing an *infinitesimal annealing* algorithm for S^{3}VM named S^{3}VM$^{\text{path}}$. The basic idea of S^{3}VM$^{\text{path}}$ is to use a convex *parametric programming* technique (Allgower & George, 1993; Gal, 1995; Best, 1996) for computing the entire solution path of S^{3}VM for $C^* \in [0, C]$. However, since S^{3}VM is non-convex, our target is to compute a path of *local optimal* solutions.

To this end, we need to characterize the properties of S^{3}VM local optimal solutions. Given that we have a convex optimization problem defined in a convex polytope for fixed predicted labels $\hat{y}$ (the inner optimization problem in (3)), we define the following notion:

Definition 1 (Conditionally optimal solution) *For a given $\hat{y} \in \{-1, 1\}^{|\mathcal{U}|}$, we refer to the optimal solution of the convex problem*

$$f^*_{\hat{y}} := \arg \min_{f \in \text{pol}(\hat{y})} J(f, \hat{y}) \tag{5}$$

as the conditionally optimal solution in pol($\hat{y}$)*, where*

$$\text{pol}(\hat{y}) := \{f | \hat{y}_i f(x_i) \geq 0, i \in \mathcal{U}\} \tag{6}$$

is the convex polytope defined by the constraints in (2).

Roughly speaking, the solution space of S^{3}VM consists of many such convex polytopes. As we will show in the next section, the S^{3}VM solution space possesses the following two important properties:

- If a conditionally optimal solution is in the strict *interior* of the current convex polytope, it is an S^{3}VM local optimal solution (see Figure 2 and Theorem 4).
- If a conditionally optimal solution is at a *boundary* of the current convex polytope, it is not local optimal, and a better solution can be always found in its adjacent polytope (see Figure 3 and Theorem 5).

These two properties indicate that a path of S^{3}VM local optimal solutions is inevitably *discontinuous* and contains a finite number of abrupt *jumps*. To cope with the discontinuity, the proposed S^{3}VM$^{\text{path}}$ algorithm consists of the *continuous path (CP)* step and the *discrete jump (DJ)* step. More specifically, starting from $C^* = 0$, the S^{3}VM$^{\text{path}}$ algorithm iterates the CP step (following the local optimal solution path in a polytope) and the DJ step (once the path reaches a boundary of the polytope, find another local optimal solution in the adjacent polytope) until $C^* = C$. Figure 4 illustrates the behavior of the algorithm, in which the red ones indicate the CP step, while the blue ones indicate the DJ step.

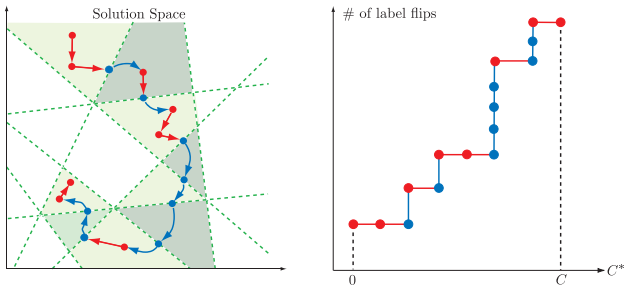

Figure 4. A Schematic illustration of the behavior of the S^3VM^{path} algorithm. (Left) S^3VM solution space consists of a number of convex polytopes, each characterized by the predicted label $\hat{y}$. As long as the conditionally optimal solution (see Definition 1) is in the strict interior of the current polytope, we can use a conventional parametric programming (a.k.a. regularization path following) to keep track of a local optimal solution path (the CP step: Red arrows and points). However, once the solution reaches a boundary of the current convex polytope, the algorithm jumps to the conditionally optimal solution in the adjacent polytope (the DJ step: Blue arrows and points). (Right) Illustration of how C^* is increased in S^3VM^{path}. We continuously increase C^* from 0 to C while we are computing a local optimal solution path (CP step: red lines and points). Once we reaches a boundary of the polytope, we fix C^* until we find a local optimal solution at that C^* (DJ step: blue lines and points).

In the next section, we formally discuss the properties of S^3VM local optimal solutions stated above. Then we describe implementation details of the S^3VM^{path} algorithm in Section 5.

4. S^3VM local optimal solutions

In this section, we formally discuss the properties of S^3VM local optimal solutions.

To begin with, the following proposition clarifies the relationship between conditionally optimal solutions and local optimal solutions:

Proposition 2 *Any local optimal solution f of S^3VM is the conditionally optimal solution in* $\mathrm{pol}(\hat{y})$*, where $\hat{y}$ satisfies $\hat{y}_i f(x_i) \geq 0, i \in \mathcal{U}$.*

This proposition is clear because, if a solution f is not conditionally optimal for $\hat{y}$, there exists a strictly better feasible solution in the neighborhood of f. Note that the converse is not always true, i.e., every conditionally optimal solution is not necessarily a local optimal solution of (4). Therefore, we need to clarify which conditionally optimal solutions are local optimal.

Since each local optimal solution corresponds to one of the conditionally optimal solutions, the Lagrangian multiplier theory (Boyd & Vandenberghe, 2004) implies that a local optimal solution can be written in the dual form as

$$f(x) = b + \sum_{i \in \mathcal{L}} \alpha_i K(x, x_i) + \sum_{i \in \mathcal{U}} \alpha_i K(x, x_i), \qquad (7)$$

where K is the kernel function and $\{\alpha_i\}_{i \in \mathcal{L} \cup \mathcal{U}}$ are the Lagrange multipliers. Note that, in the standard SVM, the 2nd term is usually written as $\sum_i \alpha_i y_i K(x, x_i)$. Here, we augment α_i to include y_i for notational simplicity.

Then we have the following necessary and sufficient conditions for the local optimality of S^3VM:

Lemma 3 *For $C^* \in (0, C]$, necessary and sufficient conditions for f to be local optimal are*

$$\begin{aligned}
y_i f(x_i) > 1, i \in \mathcal{L} &\Rightarrow y_i \alpha_i = 0, && (8a)\\
y_i f(x_i) = 1, i \in \mathcal{L} &\Rightarrow y_i \alpha_i \in [0, C], && (8b)\\
y_i f(x_i) < 1, i \in \mathcal{L} &\Rightarrow y_i \alpha_i = C, && (8c)\\
\hat{y}_i f(x_i) > 1, i \in \mathcal{U} &\Rightarrow \hat{y}_i \alpha_i = 0, && (8d)\\
\hat{y}_i f(x_i) = 1, i \in \mathcal{U} &\Rightarrow \hat{y}_i \alpha_i \in [0, C^*], && (8e)\\
0 < \hat{y}_i f(x_i) < 1, i \in \mathcal{U} &\Rightarrow \hat{y}_i \alpha_i = C^*, && (8f)\\
\hat{y}_i f(x_i) \neq 0, i \in \mathcal{U}, & && (8g)
\end{aligned}$$

and all the constraints (2) are non-active*, i.e.,*

$$\hat{y}_i f(x_i) \neq 0, \ \forall i \in \mathcal{U}. \qquad (9)$$

The proof of Lemma 3 is given in the supplementary.

A non-trivial part of the local optimality conditions is the non-activeness condition (9). To clarify this, we rephrase Lemma 3 as follows:

Theorem 4 *A conditionally optimal solution in* $\mathrm{pol}(\hat{y})$ *for a certain $\hat{y}$ is a local optimal solution if and only if it is strictly in the interior of* $\mathrm{pol}(\hat{y})$.

This theorem is directly deduced from Lemma 3 because (8) is a part of the KKT optimality conditions of the conditionally optimal solutions in $\mathrm{pol}(\hat{y})$, while (9) indicates that the solution cannot be at the boundary of $\mathrm{pol}(\hat{y})$ (see the proof of Lemma 3 in the supplementary for details). Theorem 4 indicates that the path of $f^*_{\hat{y}}$s is guaranteed to be local optimal as long as it stays in the interior of the convex polytope $\mathrm{pol}(\hat{y})$, and the solution is not local optimal anymore when the path arrives at a boundary of the convex polytope (see Figure 2).

In order to explain why a conditionally optimal solution $f^*(\hat{y})$ at the boundary cannot be a local optimal

solution, let us consider the "adjacent" convex polytope $\mathrm{pol}(\hat{y}')$, where

$$\hat{y}'_i := \begin{cases} -\hat{y}_i, & i \in \mathcal{S}, \\ \hat{y}_i, & i \notin \mathcal{S}, \end{cases} \quad \mathcal{S} := \{i \in \mathcal{U} | f^*_{\hat{y}}(x_i) = 0\}. \tag{10}$$

Then, as stated in the following theorem, the conditionally optimal solution $f^*_{\hat{y}'}$ in $\mathrm{pol}(\hat{y}')$ is guaranteed to be a strictly better S^{3}VM solution than $f^*_{\hat{y}}$:

Theorem 5 *Let $f^*_{\hat{y}}$ be the conditionally optimal solution in* $\mathrm{pol}(\hat{y})$*, and suppose that $\hat{y}_i f^*_{\hat{y}}(x_i) = 0, i \in \mathcal{S}$, holds for a non-empty set $\mathcal{S} \subseteq \mathcal{U}$. If we define a new label vector $\hat{y}'$ by (10), i.e., the labels $\hat{y}_i$ for $i \in \mathcal{S}$ are flipped, then the conditionally optimal solution $f^*_{\hat{y}'}$ on* $\mathrm{pol}(\hat{y}')$ *satisfies*

$$J(f^*_{\hat{y}'}, \hat{y}') < J(f^*_{\hat{y}}, \hat{y}), \tag{11}$$

i.e., $f^*_{\hat{y}'}$ is a strictly better S^{3}VM solution than $f^*_{\hat{y}}$.

This theorem can be proved by comparing the KKT optimality conditions of the two solutions $f^*_{\hat{y}}$ and $f^*_{\hat{y}'}$ and showing that the former cannot be optimal in $\mathrm{pol}(\hat{y}')$. Its detailed proof is given in the supplementary.

Theorem 5 shows that, after we flip the label as in (10), the new conditionally optimal solution $f^*_{\hat{y}'}$ in the adjacent convex polytope $\mathrm{pol}(\hat{y}')$ is strictly better than the previous one. This means that, once a solution path reaches a boundary of the current polytope, we can always find a better feasible solution by computing the conditionally optimal solution in the polytope $\mathrm{pol}(\hat{y}')$ (see Figure 3).

Actually, Theorem 5 proves the "only if" part of Theorem 4. That is, the strict improvement of the solution in Theorem 5 indicates that, if $f^*_{\hat{y}}$ is at the boundary of $\mathrm{pol}(\hat{y})$, then it cannot be local optimal because there exists a strictly better feasible solution in the adjacent polytope $\mathrm{pol}(\hat{y}')$.

5. Implementation Details of S^{3}VM$^{\mathrm{path}}$

In this section, we describe implementation details of the S^{3}VM$^{\mathrm{path}}$ algorithm.

The pseudo-code of the S^{3}VM$^{\mathrm{path}}$ algorithm is summarized in Algorithms 1, 2, and 3. In these pseudo-codes, we denote $f_{[(C^*)]}$ and $\hat{y}_{[C^*]}$ to represent a local optimal solution and the corresponding label vector at C^*, respectively. In addition, a path of local optimal solutions for $C^* \in [C^*_0, C^*_1]$ is written as $f_{[C^*_0, C^*_1]}$.

Entire Procedure (Algorithm 1): The S^{3}VM$^{\mathrm{path}}$ algorithm is initialized at $C^* = 0$ with the standard supervised SVM trained only on labeled instances $\{(x_i, y_i)\}_{i \in \mathcal{L}}$. The predicted labels $\hat{y}_{[0]}$ for unlabeled instances are initialized based on the sign of $f_{[0]}(x_i), i \in \mathcal{U}$, where $f_{[0]}(x_i)$ denotes the decision function obtained by the initial SVM. After the initialization, the algorithm enters the CP-step, where the path of conditionally optimal solutions $f^*_{\hat{y}}$s is computed with increasing C^* by using a convex parametric programming technique.

If the path arrives at a boundary of the convex polytope $\mathrm{pol}(\hat{y})$, then, it exits from the CP-step, and enters the DJ-step. In the DJ-step, C^* is fixed, and a better feasible solution is sought for by computing the conditionally optimal solution $f^*_{\hat{y}'}$ after the predicted labels for $i \in \{i \in \mathcal{U} | f_{[C^*]}(x_i) = 0\}$ are flipped as in (10). This process is repeated until the newly computed conditionally optimal solution $f^*_{\hat{y}'}$ is in the strict interior of the convex polytope $\mathrm{pol}(\hat{y}')$. In that case, the algorithm exits from the DJ-step, and enters the CP-step again.

By this procedure we can eventually find a local optimal solution at the C^* because the objective function $J(\hat{y}, f)$ is bounded below. Computational complexity of S^{3}VM$^{\mathrm{path}}$ algorithm is discussed in the supplementary.

Algorithm 1 Entire procedure of S^{3}VM$^{\mathrm{path}}$

Input: $\{(x_i, y_i)\}_{i \in \mathcal{L}}$, $\{x_i\}_{i \in \mathcal{U}}$, K and C;
Output: $f_{[0,C]}$;
$f_{[0]} \leftarrow$ Train a SVM with $\{(x_i, y_i)\}_{i \in \mathcal{L}}$;
$\hat{y}_{[0]i} \leftarrow \mathrm{sgn}(f_{[0]}(x_i)), i \in \mathcal{U}$;
$C^* \leftarrow 0$;
while $C^* \leq C$ **do**
  $C^*_0 \leftarrow C^*$;
  $\{C^*_1, f_{[C^*_0, C^*_1]}\} \leftarrow$ **CP-step**$(f_{[C^*_0]}, \hat{y}_{[C^*_0]}, C^*_0)$;
  $C^* \leftarrow C^*_1$;
  $\mathcal{S} \leftarrow \{i \in \mathcal{U} | f_{[C^*]}(x_i) = 0\}$;
  $\{f_{[C^*]}, \hat{y}_{[C^*]}\} \leftarrow$ **DJ-step**$(C^*, \hat{y}_{[C^*]}, \mathcal{S})$;
end while

CP Step (Algorithm 2): The CP-step is implemented with a convex parametric programming technique. Since the optimization problem (5) for computing conditionally optimal solutions is a convex quadratic program, we can use piecewise-linear parametric programming (Best, 1996) in the same way as the famous SVM regularization path algorithm (Hastie et al., 2004).

In piecewise-linear parametric programming, we compute the sensitivity of the optimal solutions to the parameter C^* based on the KKT optimality conditions.

Here, we consider the following four index sets:

$$\begin{aligned}\mathcal{O} &:= \{i \in \mathcal{L} \cup \mathcal{U} | y_i f(x_i) > 1 \text{ or } \hat{y}_i f(x_i) > 1\},\\ \mathcal{M} &:= \{i \in \mathcal{L} \cup \mathcal{U} | y_i f(x_i) = 1 \text{ or } \hat{y}_i f(x_i) = 1\},\\ \mathcal{I}_\ell &:= \{i \in \mathcal{L} | y_i f(x_i) < 1\},\\ \mathcal{I}_u &:= \{i \in \mathcal{U} | \hat{y}_i f(x_i) < 1\}.\end{aligned}$$

If we know that the members of these four index sets are unchanged in a short range of C^*, then we can easily observe that the optimal solutions (i.e., the set of Lagrange multipliers $\{\alpha_i\}_{i \in \mathcal{L} \cup \mathcal{U}}$) are linear in C^*. It suggests that the entire path of the conditionally optimal solutions is a piecewise-linear function of C^* because whenever one of the members in the above four index sets changes, the linearity (slope) of the solution path also changes. See Hastie et al. (2004) for the detail of piecewise-linear parametric programming in the context of SVMs.

The algorithm exits from the CP-step when the piecewise-linear path of $f^*_{\hat{y}}$s reaches a boundary of the convex polytope $\mathrm{pol}(\hat{y})$, i.e., any one of the unlabeled instances satisfies $f(x_i) = 0, i \in \mathcal{U}$. This moment can be exactly and easily identified by exploiting the piecewise linearity of the solution path, and it does not significantly increase the computational cost of piecewise-linear parametric programming.

Algorithm 2 CP-step

Input: $C^*_0, f, \hat{y}$;
Output: $C^*_1, f_{[C^*_0, C^*_1]}$;
$f^*_{\hat{y}} \leftarrow f$;
$C^* \leftarrow C^*_0$
while $\hat{y}_i f(x_i) > 0 \ \forall i \in \mathcal{U}$ and $C^* \leq C$ **do**
  Compute the path of $f^*_{\hat{y}}$s with increasing C^*;
end while
$C^*_1 \leftarrow C^*$;

DJ Step (Algorithm 3): If the path reaches a boundary of a convex polytope at a certain C^*, the algorithm exits the CP-step and enters the DJ-step. In the DJ-step, the parameter C^* is fixed until a local optimal solution at that C^* is found. In this step, we exploit Theorem 5. If the predicted labels $\hat{y}$ are flipped to $\hat{y}'$ as in (10), then the conditionally optimal solution defined in the new convex polytope $\mathrm{pol}(\hat{y}')$ is strictly better than the previous solution. By this strict improvement property and the fact that the number of possible $\hat{y} \in \{-1, 1\}^{|\mathcal{L}|}$ is finite, we can always find a local optimal solution at that C^* by repeating this process.

Although any convex optimization solver can be used in this step, we note that this step can be carried out very efficiently: The two conditionally optimal solutions $f^*_{\hat{y}}$ and $f^*_{\hat{y}'}$ should not be much different (the difference is only in a few constraints corresponding to $i \in \{i \in \mathcal{U} | f^*_{\hat{y}}(x_i) = 0\}$). If we use the former solution as the initial starting point of the latter optimization problem, it should be solved very efficiently. For the experiments in the next section, we have developed an active set method-type solver for the DJ-step. In our experience, new conditionally optimal solutions are usually obtained within tens of iterations.

Algorithm 3 DJ-step

Input: $C^*, \hat{y}, \mathcal{S}$;
Output: $f, \hat{y}$;
while $\mathcal{S}$ is not empty **do**
  $\hat{y}'_i \leftarrow \hat{y}_i, i \in \mathcal{S}$;
  Compute $f^*_{\hat{y}}$;
  $\mathcal{S} \leftarrow \{i \in \mathcal{U} | f^*_{\hat{y}'}(x_i) = 0\}$;
  $\hat{y} \leftarrow \hat{y}'$;
end while
$f \leftarrow f^*_{\hat{y}}$;

6. Experiments

In this section, we report experimental results.

Setup: The proposed $\mathrm{S^3VM^{path}}$ is compared with the supervised SVM (Vapnik, 1996) and two existing $\mathrm{S^3VM}$ algorithms in terms of generalization performance, optimization performance, and computation time: $\mathrm{S^3VM^{light}}$ (Joachims, 1999) which is also an annealing algorithm that computes a sequence of solutions with increasing C^*, and an algorithm (Collobert et al., 2006) based on a general non-convex solver called the *convex-concave procedure* (CCCP) (Yuille & Rangarajan, 2002).

We use the 10 benchmark datasets listed in Table 1. In each data set, a half of the instances are used for training. We set the number of labeled instances at $|\mathcal{L}| = \min\{30, 10\% \text{ of the training set size}\}$, and the rest of the training instances are used as unlabeled instances. From the remaining half of the data set, we choose validation instances of size equal to $|\mathcal{L}|$, which are used only for tuning hyper-parameters. The rest of instances are used for evaluating the generalization performance.

Generalization Performance: First, we compare the generalization performance for (i) unlabeled instances used for training (i.e., the transduction error), and (ii) test instances which are not used for training (i.e., the generalization error). We use the Gaussian kernel: $K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|)$. Model

Table 1. List of datasets. n denotes the number of instances and d denotes the input dimensionality

Data Set	ID	n	d
Breast Cancer	D1	569	30
Digit1	D2	1500	241
DNA	D3	2000	180
Four Class	D4	862	2
Ionosphere	D5	351	33
Segment	D6	2310	19
USPS	D7	9298	256
Splice	D8	1000	60
SVM Guide	D9	300	10
Vehicle	D10	864	18
Vowel	D11	528	10

selection is carried out by finding the best parameter combination that minimizes the validation error from $C = \{1, 10, 100, 1000\}$ and $\gamma = \{\frac{1}{4d}, \frac{1}{2d}, \frac{1}{d}, \frac{2}{d}, \frac{4}{d}\}$ where d is the input dimensionality. The best C^* is also selected based on the validation performance. In CCCP, C^* was chosen from $\{\frac{C}{8}, \frac{C}{4}, \frac{C}{2}, C\}$. For $\text{S}^3\text{VM}^{\text{light}}$ and $\text{S}^3\text{VM}^{\text{path}}$, C^* was selected from those computed in the annealing steps: In $\text{S}^3\text{VM}^{\text{light}}$, C^* was selected from $\{2^{-9}C, 2^{-8}C, \ldots, C\}$ following the suggestion in Chapelle et al. (2008); in $\text{S}^3\text{VM}^{\text{path}}$, C^* that exactly minimizes the validation error in the entire range of $[0, C]$ was selected. Note that this is possible because $\text{S}^3\text{VM}^{\text{path}}$ performs annealing with an infinitesimal resolution.

The results reported in Table 2 are the average and the standard deviation for 10 different random data splits. The table shows that the proposed $\text{S}^3\text{VM}^{\text{path}}$ gives the smallest generalization errors in many cases. This gain may be brought by the infinitesimal annealing effect.

Optimization Performance: Next, we compare the optimization performance of the three S^3VM algorithms. Since S^3VM involves a non-convex optimization problem, it is interesting to see how good local optimal solutions can be found by each algorithm. Indeed, as pointed out in Chapelle et al. (2008), finding good local optimal solutions of the problem (1) or (4) leads to good S^3VM generalization performances.

The optimization performance of each algorithm is compared in terms of the objective function value $J(f, \hat{y})$ defined in (1). After we obtained predicted labels $\hat{y}$ from each of the three algorithms, we computed the conditionally optimal solution $f^*_{\hat{y}}$, and $J(f^*_{\hat{y}}, \hat{y})$.

Figure 5 plots the experimental results, where $\gamma = \frac{1}{d}$ and $C = C^* = \{1, 10, 100, 1000\}$. The results show that the proposed S^3VM almost consistently outperforms $\text{S}^3\text{VM}^{\text{light}}$ and CCCP. We can also observe positive correlation between objective function values and generalization errors. We conjecture that $\text{S}^3\text{VM}^{\text{path}}$ yields better generalization performances in Table 2 partly because it finds better local optimal solutions.

Table 2. Mean generalization error over 10 runs. In each data set, the upper row shows the error on unlabeled instances and the lower row shows the error on test instances. Smaller generalization error is better. Numbers in bold face indicate the best method in terms of the mean generalization error for each setup.

Data	SVM	$\text{S}^3\text{VM}^{\text{light}}$	CCCP	$\text{S}^3\text{VM}^{\text{path}}$
D1	8.13(1.35)	10.08(1.29)	6.41(0.44)	**6.09(0.75)**
	7.62(0.79)	7.12(0.69)	5.95(0.70)	**5.90(0.42)**
D2	12.44(0.39)	13.68(0.63)	13.35(1.13)	**12.31(0.48)**
	12.58(0.50)	13.06(0.58)	12.83(1.23)	**11.44(0.41)**
D3	20.20(1.21)	11.41(0.62)	11.46(0.84)	**10.85(1.11)**
	20.74(1.22)	**10.30(0.75)**	10.93(0.67)	12.19(1.15)
D4	34.81(0.50)	**22.12(2.21)**	32.82(0.53)	32.54(0.66)
	35.16(0.27)	**29.48(1.18)**	35.36(1.83)	33.42(0.44)
D5	21.33(2.60)	15.76(2.17)	15.32(1.94)	**12.53(1.34)**
	19.81(2.40)	13.96(1.69)	12.26(2.21)	**10.14(1.35)**
D6	4.87(0.35)	7.10(1.11)	5.61(0.88)	**3.75(0.30)**
	4.97(0.29)	4.83(0.45)	5.81(0.98)	**4.44(0.45)**
D7	15.72(1.42)	12.75(1.25)	12.07(1.46)	**10.29(0.81)**
	15.92(1.42)	12.48(1.23)	12.22(1.43)	**11.53(1.52)**
D8	35.85(2.08)	**28.51(1.10)**	29.11(1.08)	29.00(0.98)
	36.55(2.07)	28.09(0.90)	28.28(1.01)	**25.34(1.16)**
D9	48.52(1.68)	**48.07(1.23)**	48.81(1.02)	49.93(1.01)
	47.26(1.91)	46.67(1.49)	47.93(1.13)	**41.78(2.91)**
D10	32.44(1.72)	**31.58(0.83)**	36.34(2.43)	32.93(1.48)
	30.97(1.93)	29.92(1.14)	34.58(2.40)	**25.74(1.77)**

Computation Time: Finally, we compare the computation time of each algorithm. Figure 6 plots the entire computation time for training S^3VM, where the horizontal axis indicates the number of annealing steps in $\text{S}^3\text{VM}^{\text{light}}$, and the number of candidates of C^*s in CCCP. The results show that the computation time grows as the number of annealing steps and the number of C^*-candidates increase. Although more annealing steps and more C^*-candidates are preferable for better generalization performance, Figure 6 indicates that increasing the numbers of annealing steps and C^*-candidates is possible only at the expense of increasing the computational costs in $\text{S}^3\text{VM}^{\text{light}}$ and CCCP.

We can interpret the proposed $\text{S}^3\text{VM}^{\text{path}}$ as computing solutions with infinitely many C^*-candidates by infinitesimal-step annealing. However, as Figure 6 shows, its computation time is usually much smaller than CCCP, and it is comparable to $\text{S}^3\text{VM}^{\text{light}}$ that has a much smaller number of annealing steps. This means that $\text{S}^3\text{VM}^{\text{path}}$ can go beyond the trade-off between the resolution of annealing steps and the computation cost.

Overall, our proposed method, $\text{S}^3\text{VM}^{\text{path}}$, is shown to be a promising alternative to existing S^3VM training algorithms.

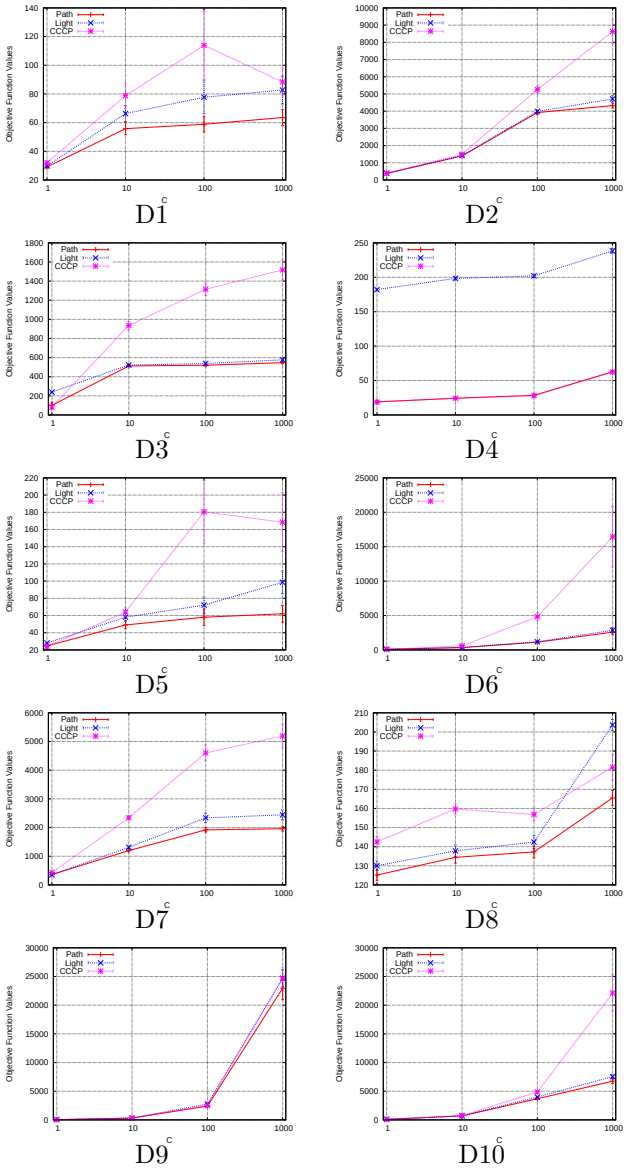

Figure 5. Objective function values $J(f, \hat{y})$ on labeled and unlabeled training instances at $C^* = C$ for $\mathrm{S^3VM^{path}}$, $\mathrm{S^3VM^{light}}$, and CCCP. Smaller values means that better local optimal solutions are found.

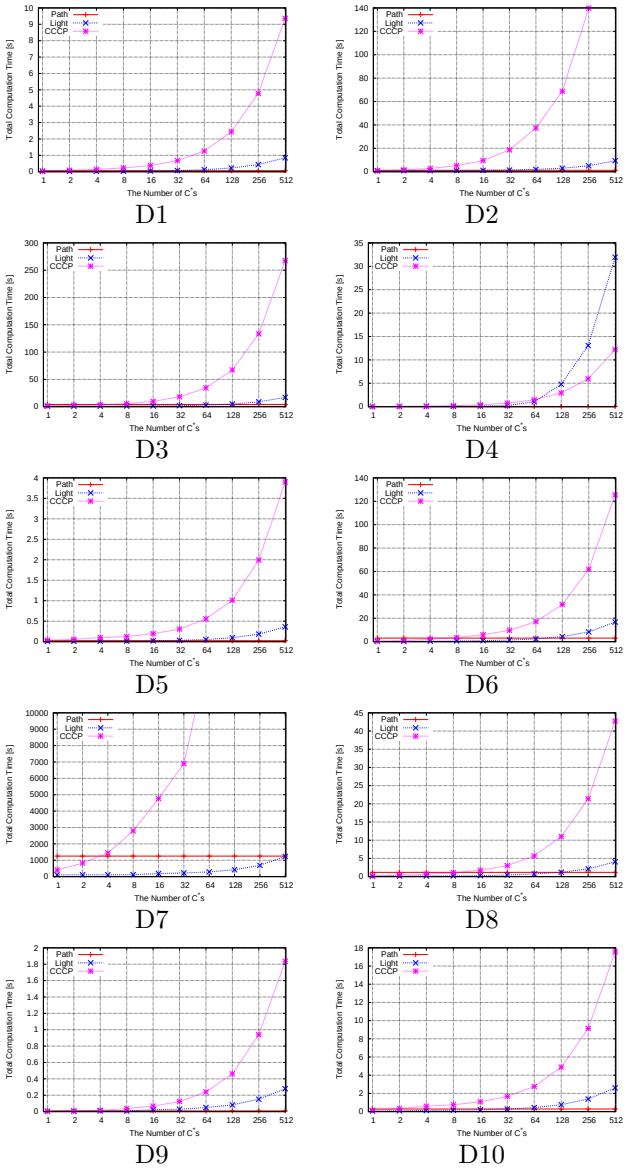

Figure 6. Total computation time for computing a sequence of solutions for different C^*s. The horizontal axis indicates the number of annealing steps in $\mathrm{S^3VM^{light}}$ and the number of candidates of C^* in CCCP. Note that the total computation time of $\mathrm{S^3VM^{path}}$ does not depend on the number of C^*s because it computes the entire path of solutions with an infinitesimal resolution.

7. Conclusions

In this paper, we proposed a novel training algorithm for $\mathrm{S^3VM}$ based on infinitesimal annealing. Our method efficiently tracks a path of local optimal solutions when the effect of unlabeled data is gradually increased. A notable difference from existing solution path algorithms is that our solution path includes *jumps*, due to non-convexity and non-smoothness of the loss function for unlabeled instances. Through experiments, we demonstrated that our algorithm, called $\mathrm{S^3VM^{path}}$, is promising in generalization performance, optimization performance, and computation time.

Acknowledgments

IT was supported by MEXT KAKENHI 23700165 and the Hori sciences and arts foundation. MS was supported by MEXT KAKENHI 23300069.

References

Allgower, E. L. and George, K. Continuation and path following. *Acta Numerica*, 2:1–63, 1993.

Best, M. J. An algorithm for the solution of the parametric quadratic programming problem. *Applied Mathemetics and Parallel Computing*, pp. 57–76, 1996.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pp. 144–152, 1992.

Boyd, S. and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, 2004.

Chapelle, O. Training a support vector machine in the primal. *Neural Computation*, 19(5):1155–1178, 2007.

Chapelle, O. and Zien, A. Semi-supervised classification by low density separation. *Tenth International Workshop on Artificial Intelligence and Statistics*, 2005.

Chapelle, O., Schölkopf, B., and Zien, A. (eds.). *Semi-Supervised Learning*. MIT Press, Cambridge, MA 2006.

Chapelle, O., Sindhwani, V., and Keerthi, S. Branch and bound for semi-supervised support vector machines. *Advances in Neural Information Processing Sysrtems*, 2007.

Chapelle, O., Sindhwani, V., and Keerthi, S. S. Optimization techniques for semi-supervised support vector machines. *J. Mach. Learning Res.*, 9:202–233, Feb. 2008.

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

Colorni, A., Dorigo, M., and Maniezzo, V. Distributed optimization by ant colonies. *Proc. European Conference on Artificial Life*, pp. 134–142, 1991.

Cortes, C. and Vapnik, V. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

Efron, B. and Tibshirani, R. Least angle regression. *Annals of Statistics*, 32(2):407–499, 2004.

Gal, T. *Postoptimal Analysis, Parametric Programming, and Related Topics*. Walter de Gruyter, 1995.

Hastie, T., Rosset, S., Tibshirani, R., and Zhu, J. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5: 1391–415, 2004.

Hromkovic, J. *Algorithmics for Hard Problems*. Springer, 2001.

Joachims, T. Transductive inference for text classification using support vector machines. *International Conference on Machine Learning*, 1999.

Karasuyama, M., Harada, N., Sugiyama, M., and Takeuchi, I. Multi-parametric solution-path algorithm for instance-weighted support vector machines. *Machine Learning*, 88(3):297–330, 2012.

Kirkpatrick, S. and Gelatt, C. D. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

Korte, B. and Vygen, J. *Combinatorial Optimization: Theory and Algorithms*. Springer-Verlag, Berlin, 2000.

Ritter, K. On parametric linear and quadratic programming problems. *mathematical Programming: Proceedings of the International Congress on Mathematical Programming*, pp. 307–335, 1984.

Sindhwani, V., Keerthi, S., and Chapelle, O. Deterministic annealing for semi-supervised kernel machines. *International Conference on Machine Learning*, 2006.

Takeuchi, I., Nomura, K., and Kanamori, T. Nonparametric conditional density estimation using piecewise-linear solution path of kernel quantile regression. *Neural Computation*, 21(2):539–559, 2009.

Vapnik, V. N. *The Nature of Statistical Learning Theory*. Springer, 1996.

Vapnik, V. N. and Sterin, A. On structural risk minimization or overall risk in a problem of pattern recognition. *Automation and Remote Control*, 1977.

Yuille, A. L. and Rangarajan, A. The concave-convex procedure (cccp). In *Advances in Neural Information Processing Systems*, volume 14, 2002.

A. Proof of Lemma 3 and Theorem 5

First, the KKT optimality conditions of a conditionally optimal solution is written as follows:

Lemma 6 *For a given $\hat{y} \in \{-1, 1\}^{|\mathcal{U}|}$, the necessary and sufficient conditions for a f to be the optimal solution of the convex problem (5) is (8) and*

$$\hat{y}_i \alpha_i \geq C^* \text{ for } i \in \{i \in \mathcal{U} | \hat{y}_i f(x_i) = 0\}. \tag{13}$$

We omit the proof of this lemma because they are straightforwardly derived by using Lagrange multiplier theory (Boyd & Vandenberghe, 2004). Here, we just note that the derivation is almost same as the standard SVM case because the predicted labels $\hat{y}$ are fixed here.

Based on Lemma 6, we first prove Theorem 5.

Proof of Theorem 5 Let $f^*_{\hat{y}}$ and $f^*_{\hat{y}'}$ be two conditionally optimal solutions defined in $\mathrm{pol}(\hat{y})$ and $\mathrm{pol}(\hat{y}')$, respectively, and consider a situation that the former $f^*_{\hat{y}}$ is at a boundary of $\mathrm{pol}(\hat{y})$. To prove the theorem, we suppose for the moment that it is also conditionally optimal in the next polytope $\mathrm{pol}(\hat{y}')$, i.e., $f^*_{\hat{y}} = f^*_{\hat{y}'}$.

Since $f^*_{\hat{y}}$ and $f^*_{\hat{y}'}$ are conditionally optimal, they satisfy the optimality condition (13):

$$\hat{y}_i \alpha_i \geq C^* \text{ for } i \in \{i \in \mathcal{U} | \hat{y}_i f^*_{\hat{y}}(x_i) = 0\}, \tag{14}$$

and

$$\hat{y}'_i \alpha_i \geq C^* \text{ for } i \in \{i \in \mathcal{U} | \hat{y}'_i f^*_{\hat{y}'}(x_i) = 0\}, \tag{15}$$

respectively. From our current assumption that $f^*_{\hat{y}} = f^*_{\hat{y}'}$ and the fact that

$$y'_i = -y_i, i \in \{i \in \mathcal{U} | y_i f^*_{\hat{y}}(x_i) = 0\}, \tag{16}$$

(14) is rewritten as

$$\hat{y}'_i \alpha_i \leq -C^* \text{ for } i \in \{i \in \mathcal{U} | \hat{y}'_i f^*_{\hat{y}'}(x_i) = 0\}. \tag{17}$$

Now, it is clear that the two conditions (15) and (17) cannot be satisfied at the same time, and it disprove our assumption that $f^*_{\hat{y}} = f^*_{\hat{y}'}$.

Noting that $f^*_{\hat{y}} \in \mathrm{pol}(\hat{y}')$ and that it is not the conditionally optimal solution in $\mathrm{pol}(\hat{y}')$, we immediately arrive at the conclusion that $f^*_{\hat{y}'}$ is a better S^{3}VM solutions than $f^*_{\hat{y}}$. **Q.E.D.**

Next, we prove Lemma 3, which is immediately obtained from Lemma 6 and Theorem 5.

Proof of Lemma 3 First, if the conditionally optimal solution $f^*_{\hat{y}}$ is in the strict interior of the convex polytope $\mathrm{pol}(\hat{y})$, it is clear that there is no better solution in the arbitrary neighborhood of $f^*_{\hat{y}}$. It suggests that $f^*_{\hat{y}}$ is a local optimal solution of S^{3}VM is it is in the strict interior of $\mathrm{pol}(\hat{y})$. On the other hand, from Theorem 5, $f^*_{\hat{y}}$ is not a local optimal solution of S^{3}VM because there exists a strictly better solution in the adjacent convex polytope $\mathrm{pol}(\hat{y}')$. Combining the fact that f is conditionally optimal if and only if (8) and (13) are satisfied, it is clear that (8) and (9) are the necessary and sufficient conditions of a local optimal solution. **Q.E.D.**

B. Computational Complexity of S^{3}VM Algorithm

The computational cost of the entire algorithm (from $C^* = 0$ to C) depends on the number of so-called *breakpoints* in the CP-step and the number of movements to adjacent polytopes in the DJ-step. It has been reported in many empirical studies (Efron & Tibshirani, 2004; Hastie et al., 2004) that the number of breakpoints is $\mathcal{O}(n)$, where n is the training set size. We also observed in our experiments that the total number of breakpoints in all CP steps scales almost linearly with respect to $\mathcal{O}(|\mathcal{L}| + |\mathcal{U}|)$.

The main computational cost in each breakpoint is the same as that in the SVM regularization path (Hastie et al., 2004). That is, at each breakpoint, we need to solve a rank-one update problem of a linear system of equations of size $O(|\mathcal{M}|)$, which costs $\mathcal{O}(|\mathcal{M}|^2)$. On the other hand, the number of movements between two polytopes depends on the number of unlabeled instances. In our experience, this number also scales linearly with respect to $\mathcal{O}(|\mathcal{U}|)$. Note that, if we use a warm-start strategy from the previous conditionally optimal solution, the computational cost of the DJ-step is negligibly small compared with the CP-step.