Incremental and Decremental Optimal Margin Distribution Learning

Li-Jun Chen, **Teng Zhang***, **Xuanhua Shi** and **Hai Jin**
National Engineering Research Center for Big Data Technology and System
Services Computing Technology and System Lab, Cluster and Grid Computing Lab
School of Computer Science and Technology, Huazhong University of Science and Technology, China
{ljchen, tengzhang, xhshi, hjin}@hust.edu.cn

Abstract

Incremental and decremental learning (IDL) deals with the tasks where new data arrives sequentially as a stream or old data turns unavailable continually due to the privacy protection. Existing IDL methods mainly focus on support vector machine and its variants with *linear*-type loss. There are few studies about the *quadratic*-type loss, whose Lagrange multipliers are unbounded and much more difficult to track. In this paper, we take the latest statistical learning framework *optimal margin distribution machine* (ODM) which involves a quadratic-type loss due to the optimization of margin variance, for example, and equip it with the ability to handle IDL tasks. Our proposed ID-ODM can avoid updating the Lagrange multipliers in an infinite range by determining their optimal values beforehand so as to enjoy much more efficiency. Moreover, ID-ODM is also applicable when multiple instances come and leave simultaneously. Extensive empirical studies show that ID-ODM can achieve $9.1\times$ speedup on average with almost no generalization lost compared to retraining ODM on new data set from scratch.

1 Introduction

For machine learning tasks, once data changes, the models should be refined. However, retraining models from scratch usually incurs unbearable cost or is even impossible in some situations, which gives rise to the *incremental and decremental learning* (IDL) [Schlimmer and Fisher, 1986; Cauwenberghs and Poggio, 2000]. This paradigm can efficiently update the model and be applied in many learning problems, just to name a few, anomaly detection [Laxhammar and Falkman, 2014; Nguyen *et al.*, 2019], concept drifting [Syed *et al.*, 1999; Gâlmeanu and Andonie, 2022], active learning [Ertekin *et al.*, 2007; Huang *et al.*, 2020], privacy protection [Nguyen *et al.*, 2020; Sekhari *et al.*, 2021] , and model estimation [Cherubin *et al.*, 2021].

Existing IDL methods mainly focus on *support vector machine* (SVM) [Cortes and Vapnik, 1995] and its variants. We summarize the representative works of this line in Table 1. Based on the path following method, IDSVM [Cauwenberghs and Poggio, 2000] takes both incremental learning and decremental learning into account to update SVM. MID-SVM [Karasuyama and Takeuchi, 2009] is a batch version of IDSVM. AONSVM [Gu *et al.*, 2012] and FISVDD [Jiang *et al.*, 2019] extends the ν-SVM [Schölkopf *et al.*, 2000] and SVDD [Tax and Duin, 2004] to incremental learning, respectively. All these models use hinge loss. Besides, IL-S3VM [Gu *et al.*, 2018] and ILTSVM [Chen *et al.*, 2023] are incremental semi-supervised SVMs with symmetric hinge loss or its variant ramp loss.

On the one hand, the aforementioned methods mainly focus on large margin models, but the latest theoretical studies [Gao and Zhou, 2013] disclose that the margin distribution is much more critical to the generalization performance. On the other hand, these methods are limited to *linear*-type loss, in which the hinge loss is sensitive to outliers and symmetric hinge loss leads to non-convex optimization problems. In contrast, the *quadratic*-type loss is smoother and numerically easier to deal with. It results in a strictly diagonally dominant matrix in dual quadratic form which is always invertible, thus relaxes the restriction of *linear*-type loss that data should be linearly independent. As far as we know, there are few IDL methods concerning models with *quadratic*-type loss.

In this paper, we take the latest statistical learning framework *optimal margin distribution machine* (ODM) [Zhang and Zhou, 2019] which involves a *quadratic*-type loss due to the optimization of margin variance as an example and equip it with the ability to handle IDL tasks. The drawback of *quadratic*-type loss lies in the optimization aspect, that is the Lagrange multipliers are unbounded and much more difficult to track when data varies, and existing IDL methods with *linear*-type loss can hardly be adapted to these circumstances. To overcome this difficulty, we estimate the optimal value of the Lagrange multiplier of the varying data in advance to avoid the optimization in an infinite range, and then update along the fastest direction to make the interruption by breakpoints less frequent. We also theoretically analyze the time complexity and convergence of our proposed incremental and decremental ODM (ID-ODM). The remarkable advantages are as follows:

- It is the first attempt to implement IDL for kernel methods optimizing the margin distribution.

*Corresponding Author

Methods	Models	Tasks	Loss	Applications
ISVM [Rüping, 2001; Liang and Li, 2009]	SVM	BC	hinge loss, linear	single, incremental
IDSVM [Cauwenberghs and Poggio, 2000]	SVM	BC	hinge loss, linear	single, inc. & dec.
WIDSVM [Gâlmeanu and Andonie, 2022]	W-SVM	BC	hinge loss, linear	single, inc. & dec.
MID-SVM [Karasuyama and Takeuchi, 2009]	SVM	BC	hinge loss, linear	multiple, inc. & dec.
INSVR [Gu *et al.*, 2015b]	ν-SVR	RG	hinge loss, linear	single, incremental
ISVOR [Gu *et al.*, 2015a]	SVOR	OR	hinge loss, linear	single, incremental
FISVDD [Jiang *et al.*, 2019]	SVDD	OCC	hinge loss, linear	single, incremental
oSVM$^{\mathrm{R}}$ [Wang and Vucetic, 2009]	SVM$^{\mathrm{R}}$	BC	ramp loss, linear	single, incremental
IL-S3VM [Gu *et al.*, 2018]	S3VM	SSC	ramp loss, linear	single, incremental
ILTSVM [Chen *et al.*, 2023]	TSVM	SSC	symmetric hinge loss, linear	single, incremental
ID-ODM	**ODM**	BC	**ODM loss, quadratic**	**multiple, inc. & dec.**

Table 1: An overview of the existing literature on incremental and decremental learning, in which BC, RG, OR, OCC, and SSC are the abbreviations of binary classification, regression, ordinary regression, one-class classification, and semi-supervised classification, respectively. "single" and "multiple" represent the number of varying instances.

- It is the first effort to track unbounded Lagrange multipliers brought by quadratic-type loss.
- It is applicable when multiple instances come and leave simultaneously.

The rest of this paper is organized as follows. First, we introduce some preliminaries. Then we detail the ID-ODM, followed by the time complexity and convergence analysis. After that, we present the experimental results and empirical observations. Finally, we conclude the paper with future work.

2 Preliminaries

We first introduce some conventions and notations used throughout the paper. The normal font letters (e.g., y) denote scalars. The boldface letters (e.g., $\boldsymbol{w}$ and $\mathbf{W}$) denote vectors and matrices, respectively. The upper case letters with mathcal font (e.g., $\mathcal{S}$) indicate the sets. Particularly, $[m]$ is defined as the integer set $\{1, 2, \ldots, m\}$. $\boldsymbol{w}_{\mathcal{S}}$ indicates the vector consisting of the entries of $\boldsymbol{w}$ specified by index set $\mathcal{S}$. Similarly, $\mathbf{W}_{\mathcal{S},\mathcal{S}'}$ is the submatrix of $\mathbf{W}$ specified by row index set $\mathcal{S}$ and column index set $\mathcal{S}'$.

For a binary classification problem, let $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} = \{1, -1\}$ denote the instance space and label set, respectively. The training set $\{(\boldsymbol{x}_i, y_i)\}_{i\in[m]} \in (\mathcal{X} \times \mathcal{Y})^m$ is drawn i.i.d from some underlying distribution on $\mathcal{X} \times \mathcal{Y}$. Let $\phi : \mathcal{X} \mapsto \mathbb{H}$ be a feature mapping associated with some positive definite kernel κ where $\mathbb{H}$ is the corresponding *reproducing kernel Hilbert space* (RKHS). The hypothesis is defined as the linear function $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \phi(\boldsymbol{x})\rangle_{\mathbb{H}}$.

This linear decision function naturally leads to the definition of margin, i.e., $\gamma(\boldsymbol{x}, y) = y\langle \boldsymbol{w}, \phi(\boldsymbol{x})\rangle_{\mathbb{H}}$. Notice that f misclassifies $(\boldsymbol{x}, y)$ if and only if it produces a negative margin, thus it represents the confidence of the prediction results. Recent studies on margin theory reveal that margin distribution is more important to generalization than a single margin, which gives rise to the ODM:

$$\min_{\boldsymbol{w},\xi_i,\epsilon_i} \frac{1}{2}\|\boldsymbol{w}\|_{\mathbb{H}}^2 + \frac{\lambda}{2}\sum_{i\in[m]}(\xi_i^2 + \epsilon_i^2), \tag{1}$$
$$\text{s.t. } 1 - \theta - \xi_i \le \gamma_i \le 1 + \theta + \epsilon_i,\ \forall i \in [m],$$

where $\gamma_i = y_i f(\boldsymbol{x}_i)$ and λ is the trade-off hyperparameter. The margin mean has been fixed as 1 since scaling $\boldsymbol{w}$ does not affect the decision boundary. The slack variables ξ_i and ϵ_i represent the deviation from the strip centered at margin mean with width 2θ. The hyperparameter θ is to tolerate the tiny deviations smaller than θ. When it is set as 0, the third term is exactly the margin variance.

Introducing the Lagrange multipliers $\zeta_i \ge 0$ and $\beta_i \ge 0$ for constraints leads to the Lagrangian of Eqn. (1):

$$L = \frac{1}{2}\|\boldsymbol{w}\|_{\mathbb{H}}^2 - \sum_{i\in[m]}\zeta_i(\gamma_i - (1 - \theta - \xi_i))$$
$$+ \frac{\lambda}{2}\sum_{i\in[m]}(\xi_i^2 + \epsilon_i^2) + \sum_{i\in[m]}\beta_i(\gamma_i - (1 + \theta + \epsilon_i)).$$

The KKT conditions [Boyd and Vandenberghe, 2004] are

$$\boldsymbol{w} = \sum_{i\in[m]} y_i(\zeta_i - \beta_i)\phi(\boldsymbol{x}_i),\ \lambda\xi_i = \zeta_i,\ \lambda\epsilon_i = \beta_i,$$
$$\zeta_i(\gamma_i - (1 - \theta) + \xi_i) = 0, \tag{2}$$
$$\beta_i(\gamma_i - (1 + \theta) - \epsilon_i) = 0. \tag{3}$$

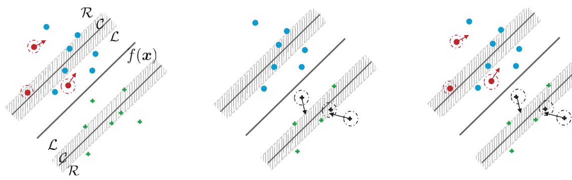
The analysis on complementary slackness conditions, i.e., Eqn. (2)-(3), leads to the partition of data into three sets:

- $\mathcal{L} = \{i \mid \gamma_i < 1 - \theta,\ \zeta_i > 0,\ \beta_i = 0\}$,
- $\mathcal{C} = \{i \mid \gamma_i \in [1 - \theta, 1 + \theta],\ \zeta_i = 0,\ \beta_i = 0\}$,
- $\mathcal{R} = \{i \mid \gamma_i > 1 + \theta,\ \zeta_i = 0,\ \beta_i > 0\}$,

which are shown in Figure 1(a), respectively. Besides, let $\alpha_i = \zeta_i - \beta_i$ and $\mathcal{S} = \mathcal{L} \cup \mathcal{R}$ is the index set of instances whose $\alpha_i \ne 0$, the decision function can be rewritten as

$$f(\boldsymbol{x}_i) = \langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle_{\mathbb{H}} = \sum_{j\in\mathcal{S}} \alpha_j y_j \kappa_{ij} \tag{4}$$

with $\kappa_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$.



(a) (b) (c)

Figure 1: Description of different tasks. (a): Add multiple new instances (red points). (b): Remove multiple invalid instances (black points). (c): Add new instances and remove invalid instances simultaneously.

3 ID-ODM for Single Instance Varying

In this section, we investigate the case when single instance comes or leaves.

3.1 Incremental Case

Suppose the new instance is $(\boldsymbol{x}_k, y_k)$. If $\gamma_k \in [1-\theta, 1+\theta]$, acccording to Eqn. (4), this instance can be directly added to $\mathcal{C}$ without any update. Thus without loss of generality, we assume $\gamma_k \notin [1-\theta, 1+\theta]$ and initialize the Lagrange multiplier α_k as 0. Next, we first derive the relationship between $\boldsymbol{\alpha}_\mathcal{S}$ and α_k, and then detail how to update $\boldsymbol{\alpha}_\mathcal{S}$ by varying α_k gradually.

The relationship between $\boldsymbol{\alpha}_\mathcal{S}$ and α_k. Notice that $1 \pm \theta$ are constants, to make the KKT conditions hold, i.e.,

$$\gamma_i + \xi_i = 1 - \theta, \quad \forall i \in \mathcal{L},$$
$$\gamma_i - \epsilon_i = 1 + \theta, \quad \forall i \in \mathcal{R},$$

the Lagrange multiplier $\boldsymbol{\alpha}_\mathcal{S}$ should vary with α_k such that

$$\Delta\gamma_i + \Delta\xi_i = 0, \quad \forall i \in \mathcal{L}, \tag{5}$$
$$\Delta\gamma_i - \Delta\epsilon_i = 0, \quad \forall i \in \mathcal{R}, \tag{6}$$

where Δ denotes the variations of variables. With Eqn. (4), we have

$$\Delta\gamma_i = y_i \Delta f(\boldsymbol{x}_i) = \sum_{j \in \mathcal{S}} \Delta\alpha_j y_i y_j \kappa_{ij} + \Delta\alpha_k y_i y_k \kappa_{ik}. \tag{7}$$

Notice that $\alpha_i = \lambda\xi_i$ for $\forall i \in \mathcal{L}$ and $\alpha_i = -\lambda\epsilon_i$ for $\forall i \in \mathcal{R}$, we can derive the following linear system

$$\sum_{j \in \mathcal{S}} \Delta\alpha_j h_{ij} + \Delta\alpha_k h_{ik} + \Delta\alpha_i/\lambda = 0, \ \forall i \in \mathcal{S}, \tag{8}$$

where $h_{ij} = y_i y_j \kappa_{ij}$. Denote $\mathbf{H} = (h_{ij})_{i,j \in [k]}$, then Eqn. (8) can be rewritten in a matrix form

$$(\mathbf{H}_{\mathcal{S}\mathcal{S}} + \mathbf{I}_{|\mathcal{S}|}/\lambda)\Delta\boldsymbol{\alpha}_\mathcal{S} = -\mathbf{H}_{\mathcal{S}k}\Delta\alpha_k,$$

which implies that $\Delta\boldsymbol{\alpha}_\mathcal{S}$ varies linearly with $\Delta\alpha_k$:

$$\Delta\boldsymbol{\alpha}_\mathcal{S} = -\mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}k}\Delta\alpha_k = \boldsymbol{t}\Delta\alpha_k, \tag{9}$$

where $\mathbf{Q}_\mathcal{S} = \mathbf{H}_{\mathcal{S}\mathcal{S}} + \mathbf{I}_{|\mathcal{S}|}/\lambda$ and $\boldsymbol{t} = -\mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}k}$.

Update α_k. By substituting Eqn. (9) into Eqn. (7) to eliminate $\Delta\boldsymbol{\alpha}_\mathcal{S}$, the remaining variable is only $\Delta\alpha_k$. We can monitor the variation of margin with $\Delta\alpha_k$ for all instances.

Different from previous works like [Cauwenberghs and Poggio, 2000] in which α_k belongs to $[0, C]$ and can be simply updated by increasing gradually from 0 to C, here $\alpha_k \in \mathbb{R}$ is *unbounded*, and we do not even know whether to increase or decrease α_k. Fortunately, if $(\boldsymbol{x}_k, y_k)$ is successfully added, by KKT conditions, we have

$$\gamma_k + \Delta\gamma_k^\star + (\alpha_k + \Delta\alpha_k^\star)/\lambda = d, \tag{10}$$

where

$$d = \begin{cases} 1-\theta, & \gamma_k < 1-\theta, \\ 1+\theta, & \gamma_k > 1+\theta. \end{cases} \tag{11}$$

With Eqn. (7) and Eqn. (9), we have a closed-form solution

$$\Delta\alpha_k^\star = \frac{d - \gamma_k - \alpha_k/\lambda}{1 + 1/\lambda + \mathbf{H}_{k\mathcal{S}}\boldsymbol{t}}. \tag{12}$$

The above analysis is based on the assumption that the three index sets $\mathcal{L}$, $\mathcal{C}$, and $\mathcal{R}$ remain unchanged when α_k is varying, thus Eqn. (12) is not an actual optimal solution, but it can provide a credible update direction. In practice, we can let $\Delta\alpha_k = \eta\Delta\alpha_k^\star$ and increases η from 0 to 1 gradually. At the same time, we monitor the three index sets. When one of the following events occurs

- $i \in \mathcal{S}$ migrates to $\mathcal{C}$, i.e., nonzero α_i turns to 0,
- $i \in \mathcal{C}$ migrates to $\mathcal{S}$, i.e., $f(\boldsymbol{x}_i)$ reaches the boundary of the strip as shown in Figure 1,

we call a *breakpoint* is detected and sequentially perform the following four steps

1. Stop the increase of η.
2. Update the index sets $\mathcal{L}$, $\mathcal{C}$, and $\mathcal{R}$.
3. Record the current α_k and $\boldsymbol{\alpha}_\mathcal{S}$.
4. Update $\mathbf{H}$, $\boldsymbol{t}$, and $\Delta\alpha_k^\star$.

The above procedure is repeated until no breakpoint is detected during the increase of η from 0 to 1, then $(\boldsymbol{x}_k, y_k)$ can be added to $\mathcal{L}$ or $\mathcal{R}$ based on current γ_k.

3.2 Decremental Case

Suppose the invalid instance is $(\boldsymbol{x}_k, y_k)$. If $k \in \mathcal{C}$, i.e., $\alpha_k = 0$, it has no contribution to the model and can be removed directly. Thus without loss of generality, we assume $k \in \mathcal{S}$. Now we need to turn α_k to 0 to totally eliminate its effect on the model. Analogous to the incremental case, we first update $\mathcal{S} = \mathcal{S} \setminus \{k\}$ and derive the relationship between $\boldsymbol{\alpha}_\mathcal{S}$ and α_k, and then show how to update $\boldsymbol{\alpha}_\mathcal{S}$ by varying α_k.

The relationship between $\boldsymbol{\alpha}_\mathcal{S}$ and α_k. Since we should make KKT conditions hold at any time, the deletion of α_k from $\mathcal{S}$ can be viewed as the reverse process of addition of α_k to $\mathcal{S}$. Again, $\Delta\boldsymbol{\alpha}_\mathcal{S}$ and $\Delta\alpha_k$ satisfy

$$(\mathbf{H}_{\mathcal{S}\mathcal{S}} + \mathbf{I}_{|\mathcal{S}|}/\lambda)\Delta\boldsymbol{\alpha}_\mathcal{S} = -\mathbf{H}_{\mathcal{S}k}\Delta\alpha_k.$$

Since $\Delta\boldsymbol{\alpha}_\mathcal{S}$ changes linearly with $\Delta\alpha_k$, we can trace the variation of the margin with $\Delta\alpha_k$.

Update α_k. The update is now reduced to determine $\Delta\alpha_k$. Distinct from the incremental case, the invalid data vanishes and the algorithm terminates with $\Delta\alpha_k^\star = -\alpha_k$.

As a result, we can update α_k with $\Delta\alpha_k = -\eta\alpha_k$. Like the incremental case, each update is comprised of four steps when we keep watch on breakpoints. The whole procedure stops when $\eta = 1$.

4 ID-ODM

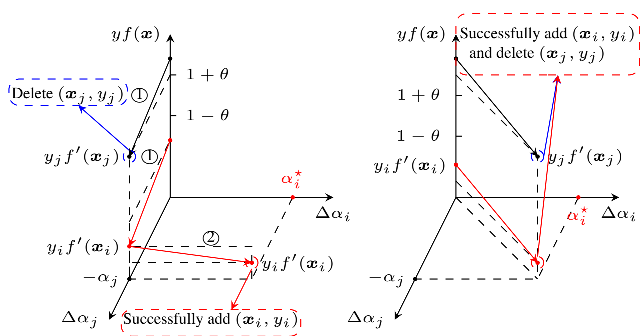
In this section, we first present the general ID-ODM for multiple instances varying and then analyze the time complexity and convergence.

4.1 Multiple Instances Varying

Without loss of generality, we assume that there are p instances $\{(\boldsymbol{x}_i, y_i)\}_{i\in\mathcal{E}}$ becoming available and q instances $\{(\boldsymbol{x}_j, y_j)\}_{j\in\mathcal{I}}$ turning invalid where

$$\mathcal{E} = \{m+1, m+2, \dots, m+p\}, \quad \mathcal{I} \subseteq \mathcal{S}.$$

We denote $\mathcal{A} = \mathcal{E} \cup \mathcal{I}$ to represent all varying instances for convenience. The most straightforward way is to view each single instance individually and invoke the method in Section 3.1 and 3.2 for p times and q times, respectively. Nevertheless, this scheme is computationally inefficient as shown in Figure 2(a).



(a) (b)

Figure 2: Different schemes to process multiple instances. (a) Update $\alpha_k, k \in \mathcal{A}$ individually. (b) Update $\boldsymbol{\alpha}_\mathcal{E}$ and $\boldsymbol{\alpha}_\mathcal{I}$ simultaneously.

One smarter way is to update $\boldsymbol{\alpha}_\mathcal{E}$ and $\boldsymbol{\alpha}_\mathcal{I}$ simultaneously as shown in Figure 2(b). Under this circumstance, multiple new instances satisfy the optimal conditions, and multiple invalid instances are removed at the same time. In the same manner as before, we first update $\mathcal{S} = \mathcal{S} \setminus \mathcal{I}$ and derive the relationship between $\boldsymbol{\alpha}_\mathcal{S}$ and $\boldsymbol{\alpha}_\mathcal{A}$, and then show how to update $\boldsymbol{\alpha}_\mathcal{E}$ and $\boldsymbol{\alpha}_\mathcal{I}$ simultaneously.

The relationship between $\boldsymbol{\alpha}_\mathcal{S}$ and $\boldsymbol{\alpha}_\mathcal{A}$. Different from the single instance case where $\Delta\alpha_k$ is a scalar, now $\Delta\boldsymbol{\alpha}_\mathcal{S}$ varies with the vector $\Delta\boldsymbol{\alpha}_\mathcal{A}$. The margin change of $(\boldsymbol{x}_i, y_i)$ is described as

$$\begin{aligned}\Delta\gamma_i &= \sum_{j\in\mathcal{S}} \Delta\alpha_j y_i y_j \kappa_{ij} + \sum_{k\in\mathcal{A}} \Delta\alpha_k y_i y_k \kappa_{ik} \\ &= \mathbf{H}_{i\mathcal{S}}\Delta\boldsymbol{\alpha}_\mathcal{S} + \mathbf{H}_{i\mathcal{A}}\Delta\boldsymbol{\alpha}_\mathcal{A} \end{aligned} \tag{13}$$

In order to make KKT conditions hold, $\Delta\boldsymbol{\alpha}_\mathcal{S}$, $\Delta\boldsymbol{\alpha}_\mathcal{E}$, and $\Delta\boldsymbol{\alpha}_\mathcal{I}$ should satisfy

$$(\mathbf{H}_{\mathcal{S}\mathcal{S}} + \mathbf{I}/\lambda)\Delta\boldsymbol{\alpha}_\mathcal{S} + \mathbf{H}_{\mathcal{S}\mathcal{A}}\Delta\boldsymbol{\alpha}_\mathcal{A} = \mathbf{0}.$$

Thus, $\Delta\boldsymbol{\alpha}_\mathcal{S}$ can be traced by $\Delta\boldsymbol{\alpha}_\mathcal{E}$ and $\Delta\boldsymbol{\alpha}_\mathcal{I}$ according to

$$\Delta\boldsymbol{\alpha}_\mathcal{S} = -\mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}\mathcal{A}}\Delta\boldsymbol{\alpha}_\mathcal{A} = [\mathbf{T}_{\mathcal{S}\mathcal{E}} \quad \mathbf{T}_{\mathcal{S}\mathcal{I}}] \begin{bmatrix}\Delta\boldsymbol{\alpha}_\mathcal{E} \\ \Delta\boldsymbol{\alpha}_\mathcal{I}\end{bmatrix} \tag{14}$$

where $\mathbf{T}_{\mathcal{S}\mathcal{E}} = -\mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}\mathcal{E}}$ and $\mathbf{T}_{\mathcal{S}\mathcal{I}} = -\mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}\mathcal{I}}$.

Update $\boldsymbol{\alpha}_\mathcal{A}$. The critical point to update the model lies in identifying $\Delta\boldsymbol{\alpha}_\mathcal{A}$. Intuitively, $\boldsymbol{\alpha}_\mathcal{I}$ is supposed to update along the opposite direction of $\boldsymbol{\alpha}_\mathcal{I}$ to remove the invalid data. In other words, $\Delta\boldsymbol{\alpha}_\mathcal{I} = -\eta\boldsymbol{\alpha}_\mathcal{I}$. As for $\boldsymbol{\alpha}_\mathcal{E}$, we determine the update direction $\Delta\boldsymbol{\alpha}_\mathcal{E}$ by keeping KKT conditions w.r.t the new instances. Since multiple instances arrive, Eqn. (10) can be recast in

$$\boldsymbol{d}_\mathcal{E} = \boldsymbol{\gamma}_\mathcal{E} + \mathbf{H}_{\mathcal{E}\mathcal{S}}\Delta\boldsymbol{\alpha}_\mathcal{S}^\star + \mathbf{H}_{\mathcal{E}\mathcal{A}}\Delta\boldsymbol{\alpha}_\mathcal{A}^\star + (\boldsymbol{\alpha}_\mathcal{E} + \Delta\boldsymbol{\alpha}_\mathcal{E}^\star)/\lambda,$$

where $\boldsymbol{d}$ is the vector with the k-th entry d_k defined in Eqn. (11). By denoting $\Delta\boldsymbol{\alpha}_\mathcal{S}$ via $\Delta\boldsymbol{\alpha}_\mathcal{E}$ and $\Delta\boldsymbol{\alpha}_\mathcal{I}$, we can determine $\Delta\boldsymbol{\alpha}_\mathcal{E}^\star$ by

$$\begin{aligned}\Delta\boldsymbol{\alpha}_\mathcal{E}^\star = &(\mathbf{Q}_\mathcal{E} + \mathbf{H}_{\mathcal{E}\mathcal{S}}\mathbf{T}_{\mathcal{S}\mathcal{E}})^{-1}(\boldsymbol{d}_\mathcal{E} - \boldsymbol{\gamma}_\mathcal{E} - \boldsymbol{\alpha}_\mathcal{E}/\lambda) \\ &- (\mathbf{Q}_\mathcal{E} + \mathbf{H}_{\mathcal{E}\mathcal{S}}\mathbf{T}_{\mathcal{S}\mathcal{E}})^{-1}(\mathbf{H}_{\mathcal{E}\mathcal{S}}\mathbf{T}_{\mathcal{S}\mathcal{I}} + \mathbf{H}_{\mathcal{E}\mathcal{I}})\Delta\boldsymbol{\alpha}_\mathcal{I}^\star.\end{aligned}$$

It seems that $\Delta\boldsymbol{\alpha}_\mathcal{E}^\star$ is relevant to $\Delta\boldsymbol{\alpha}_\mathcal{I}^\star$, but when no breakpoint is produced, $\Delta\boldsymbol{\alpha}_\mathcal{I}^\star = -\boldsymbol{\alpha}_\mathcal{I}$ is fixed. Consequently, we update $\boldsymbol{\alpha}_\mathcal{A}$ via

$$\Delta\boldsymbol{\alpha}_\mathcal{A} = \begin{bmatrix}\Delta\boldsymbol{\alpha}_\mathcal{E} \\ \Delta\boldsymbol{\alpha}_\mathcal{I}\end{bmatrix} = \eta \begin{bmatrix}\Delta\boldsymbol{\alpha}_\mathcal{E}^\star \\ -\boldsymbol{\alpha}_\mathcal{I}\end{bmatrix}$$

and track $\Delta\boldsymbol{\alpha}_\mathcal{S}$ and $\Delta f(\boldsymbol{x})$ by searching for the minimal η which will lead to a breakpoint or the termination of ID-ODM. Algorithm 1 summarizes the pseudo-code for ID-ODM.

4.2 Time Complexity

According to the Algorithm 1, the main time cost is caused by computing the inverse of the matrix $\mathbf{Q}_\mathcal{S}$. It is well accepted that the time complexity of computing the inverse of a n-order square matrix is $\mathcal{O}(n^3)$. If we calculate $\mathbf{Q}_\mathcal{S}^{-1}$ at each iteration, the time complexity is $\mathcal{O}(c(m+p-q)^3)$, where c is the number of breakpoints. Hence it is very time-consuming. However, by resorting to Sherman-Morrison-Woodbury formula [Golub and Van Loan, 1983], we can reduce the time cost to $\mathcal{O}(c(m+p-q)^2)$.

Suppose a new instance $(\boldsymbol{x}_c, y_c)$ is added to $\mathcal{S}$, let $\mathcal{S}' = \mathcal{S} \cup \{c\}$, then we have

$$\mathbf{Q}_{\mathcal{S}'} = \begin{bmatrix}\mathbf{Q}_\mathcal{S} & \mathbf{H}_{\mathcal{S}c} \\ \mathbf{H}_{c\mathcal{S}} & \mathbf{H}_{cc} + 1/\lambda\end{bmatrix} = \begin{bmatrix}\mathbf{Q}_\mathcal{S} & \mathbf{H}_{\mathcal{S}c} \\ \mathbf{H}_{c\mathcal{S}} & \kappa_{cc} + 1/\lambda\end{bmatrix}.$$

The Schur's complement [Zhang, 2006] of $\mathbf{Q}_\mathcal{S}$ is $\mathbf{S} = \kappa_{cc} + 1/\lambda - \mathbf{H}_{c\mathcal{S}}\mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}c}$. According to the Sherman-Morrison-Woodbury formula,

$$\begin{aligned}\mathbf{Q}_{\mathcal{S}'}^{-1} &= \begin{bmatrix}\mathbf{Q}_\mathcal{S}^{-1} + \mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}c}\mathbf{S}^{-1}\mathbf{H}_{c\mathcal{S}}\mathbf{Q}_\mathcal{S}^{-1} & -\mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}c}\mathbf{S}^{-1} \\ -\mathbf{S}^{-1}\mathbf{H}_{c\mathcal{S}}\mathbf{Q}_\mathcal{S}^{-1} & \mathbf{S}^{-1}\end{bmatrix} \\ &= \begin{bmatrix}\mathbf{Q}_\mathcal{S}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{bmatrix} + \begin{bmatrix}\boldsymbol{v} \\ 1\end{bmatrix}\mathbf{S}^{-1}\begin{bmatrix}\boldsymbol{v}^\top & 1\end{bmatrix},\end{aligned} \tag{15}$$

where $\boldsymbol{v} = -\mathbf{Q}_\mathcal{S}^{-1}\mathbf{H}_{\mathcal{S}c}$.

As for the decremental case, we can exchange the roles of $\mathcal{S}$ and $\mathcal{S}'$, and calculate $\mathbf{Q}_{\mathcal{S}'}^{-1}$ in the same manner.

Algorithm 1 ID-ODM

Input: The data set $\{(\boldsymbol{x}_i, y_i)\}_{i\in[m]}$, the hyperparameters $\{\lambda, \theta, \sigma\}$, the given model $\boldsymbol{\alpha}_{\mathcal{S}}$, the varying instances $\{(\boldsymbol{x}_i, y_i)\}_{i\in\mathcal{E}\cup\mathcal{I}}$.
Calculate the margin $\boldsymbol{\gamma}$, and the matrix $\mathbf{Q}^{-1}$, $\mathbf{H}$, $\mathbf{T}$.
for $k \in \mathcal{E}$ **do**
  if $\gamma_k \in [1-\theta, 1+\theta]$ **then**
    $\mathcal{C} = \mathcal{C} \cup \{k\}$, $\mathcal{E} = \mathcal{E} \setminus \{k\}$.
  end if
end for
for $k \in \mathcal{I}$ **do**
  if $k \in \mathcal{C}$ **then**
    $\mathcal{C} = \mathcal{C} \setminus \{k\}$, $\mathcal{I} = \mathcal{I} \setminus \{k\}$.
  end if
end for
$\mathcal{S} = \mathcal{S} \setminus \mathcal{I}$.
while true **do**
  Calculate the update direction $\Delta\boldsymbol{\alpha}_{\mathcal{A}}$.
  Calculate the minimal η leading to breakpoints.
  if $\eta < 1$ **then**
    Update the index sets $\mathcal{S}$ and $\mathcal{C}$.
    Update $\mathbf{Q}_{\mathcal{S}}^{-1}$, $\mathbf{T}$, $\boldsymbol{\gamma}$, $\boldsymbol{\alpha}_{\mathcal{S}}$ and $\boldsymbol{\alpha}_{\mathcal{A}}$.
  else
    Break.
  end if
end while
Output: The updated model $\boldsymbol{\alpha}_{\mathcal{S}}$.

4.3 Convergence

According to Eqn. (9), ID-ODM depends on the existence of $\mathbf{Q}_{\mathcal{S}}^{-1}$. Fortunately, for *quadratic*-type loss, $\mathbf{Q}$ is strictly diagonally dominant and positive definite, thus its inverse always exists.

When an instance $(\boldsymbol{x}_k, y_k)$ comes or leaves, α_k varies, the classification hyperplane evolves, and $\boldsymbol{\alpha}_{\mathcal{S}}$ monotonically increases or decreases [Laskov *et al.*, 2006]. To guarantee the convergence of Algorithm 1, we should further prove the monotonic update of $\boldsymbol{\alpha}_{\mathcal{S}}$ is strict, that is no instance migrates between index sets back and forth.

Theorem 1 (Immediate cycling will not occur)**.** *If a breakpoint $(\boldsymbol{x}_c, y_c)$ migrates from $\mathcal{C}$ to $\mathcal{S}$ or vice versa in round t, it will not come back in the next round.*

Proof. $\mathcal{C} \to \mathcal{S}$: Suppose $(\boldsymbol{x}_c, y_c)$ migrates from $\mathcal{C}$ to $\mathcal{L}$. Before it enters $\mathcal{L}$, we always have $\alpha_c^{(t)} = 0$. If it comes back from $\mathcal{L}$ to $\mathcal{C}$ in the next round, its margin will increase, i.e., $\Delta\gamma_c^{(t+1)} > 0$. Notice that $\Delta\gamma_i = -\Delta\alpha_i/\lambda$, $\forall i \in \mathcal{L}$ according to Eqn. (5), which implies $\Delta\alpha_c^{(t+1)} < 0$. Together with $\alpha_c^{(t)} = 0$, we have $\alpha_c^{(t+1)} < 0$. A contradiction occurs since $\alpha_i > 0$, $\forall i \in \mathcal{L}$.

The proof of migration from $\mathcal{C}$ to $\mathcal{R}$ is similar. If $(\boldsymbol{x}_c, y_c)$ comes back from $\mathcal{R}$ to $\mathcal{C}$ in the next round, its margin will decrease, and we have $\Delta\gamma_c^{(t+1)} < 0$ and $\Delta\alpha_c^{(t+1)} > 0$ by noticing that $\Delta\gamma_i = -\Delta\alpha_i/\lambda$, $\forall i \in \mathcal{R}$ according to Eqn. (6). Together with $\alpha_c^{(t)} = 0$, we have $\alpha_c^{(t+1)} > 0$, which contradicts with $\alpha_i < 0$, $\forall i \in \mathcal{R}$.

$\mathcal{S} \to \mathcal{C}$: It is equivalent to prove that the margin varies monotonically, i.e., $\Delta\gamma_c^{(t+1)}$ and $\Delta\gamma_c^{(t)}$ have the same sign.

Before a breakpoint turns out in round t, the index sets are $\mathcal{S}^{(t-1)}$ and $\mathcal{C}^{(t-1)}$. Obviously the migration of $(\boldsymbol{x}_c, y_c)$ from $\mathcal{S}^{(t-1)}$ to $\mathcal{C}^{(t)}$ can be viewed as the reverse process of migration from $\mathcal{C}^{(t)}$ to $\mathcal{S}^{(t-1)}$. Thus, we can calculate $\boldsymbol{t}^{(t)}$ as

$$
\begin{aligned}
\boldsymbol{t}^{(t)} &= -\mathbf{Q}_{\mathcal{S}^{(t-1)}}^{-1}\mathbf{H}_{\mathcal{S}^{(t-1)}k} \\
&= -\left(\begin{bmatrix}\mathbf{Q}_{\mathcal{S}^{(t)}}^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix} + \begin{bmatrix}\boldsymbol{v}\\ 1\end{bmatrix}\mathbf{S}^{-1}\begin{bmatrix}\boldsymbol{v}^\top & 1\end{bmatrix}\right)\mathbf{H}_{\mathcal{S}^{(t-1)}k} \qquad (16)\\
&= \begin{bmatrix}\boldsymbol{t}^{(t+1)}\\ 0\end{bmatrix} - \begin{bmatrix}\boldsymbol{v}\\ 1\end{bmatrix}\mathbf{S}^{-1}\left(\boldsymbol{v}^\top\mathbf{H}_{\mathcal{S}^{(t)}k} + \mathbf{H}_{ck}\right),
\end{aligned}
$$

where $\mathbf{S}$ is the Shur's complement of $\mathbf{Q}_{\mathcal{S}^{(t-1)}}$, and the first and last equality are according to the definition of $\boldsymbol{t}$, $\boldsymbol{v}$, and Eqn. (15), respectively. With Eqn. (16), we have

$$
\begin{aligned}
\Delta\alpha_c^{(t)} &= -\mathbf{S}^{-1}\left(\boldsymbol{v}^\top\mathbf{H}_{\mathcal{S}^{(t)}k} + \mathbf{H}_{ck}\right)\Delta\alpha_k^{(t)} \\
&= -\mathbf{S}^{-1}\left(-\mathbf{H}_{c\mathcal{S}^{(t)}}\mathbf{Q}_{\mathcal{S}^{(t)}}^{-1}\mathbf{H}_{\mathcal{S}^{(t)}k} + \mathbf{H}_{ck}\right)\Delta\alpha_k^{(t)}. \qquad (17)
\end{aligned}
$$

Then we can further calculate the variation of margin for the instance $(\boldsymbol{x}_c, y_c)$ in round $t+1$ as

$$
\begin{aligned}
\Delta\gamma_c^{(t+1)} &= \sum_{j\in\mathcal{S}^{(t)}} \Delta\alpha_j^{(t+1)} h_{cj} + \Delta\alpha_k^{(t+1)} h_{ck} \\
&= \mathbf{H}_{c\mathcal{S}^{(t)}}\Delta\alpha_{\mathcal{S}^{(t)}}^{(t+1)} + \mathbf{H}_{ck}\Delta\alpha_k^{(t+1)} \\
&= \left(-\mathbf{H}_{c\mathcal{S}^{(t)}}\mathbf{Q}_{\mathcal{S}^{(t)}}^{-1}\mathbf{H}_{\mathcal{S}^{(t)}k} + \mathbf{H}_{ck}\right)\Delta\alpha_k^{(t+1)},
\end{aligned}
$$

where the first and third equality are according to Eqn. (7) and Eqn. (9), respectively. With Eqn. (17), we have $\Delta\gamma_c^{(t+1)} = -\mathbf{S}\Delta\alpha_c^{(t)}\Delta\alpha_k^{(t+1)}/\Delta\alpha_k^{(t)}$.

Since the update of α_k is monotonic, $\Delta\alpha_k^{(t)}$ and $\Delta\alpha_k^{(t+1)}$ have the same sign. Notice that $\Delta\alpha_c^{(t)}$ has the opposite sign to $\Delta\gamma_c^{(t)}$, thus $\Delta\gamma_c^{(t+1)}$ has the same sign with $\Delta\gamma_c^{(t)}$ if $\mathbf{S} > 0$. Since $\mathbf{Q}_{\mathcal{S}}$ is a positive definite matrix, the corresponding Shur's complement $\mathbf{S} > 0$ [Ouellette, 1981], which concludes the proof. For multiple instance changes, the proof is similar. □

5 Experiments

In this section, we empirically analyze the effectiveness and efficiency of our proposed methods.

5.1 Setup

Data sets. We perform experiments on nine real-world data sets available on the LIBSVM website [1]. The features of each data set are normalized into $[0, 1]$. The characteristics of data sets are summarized in Table 2. We randomly divide all data sets into training and test sets with a ratio of 4:1. Besides, we randomly select 75% of the training data as historical data and the rest are varying data.

[1] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html

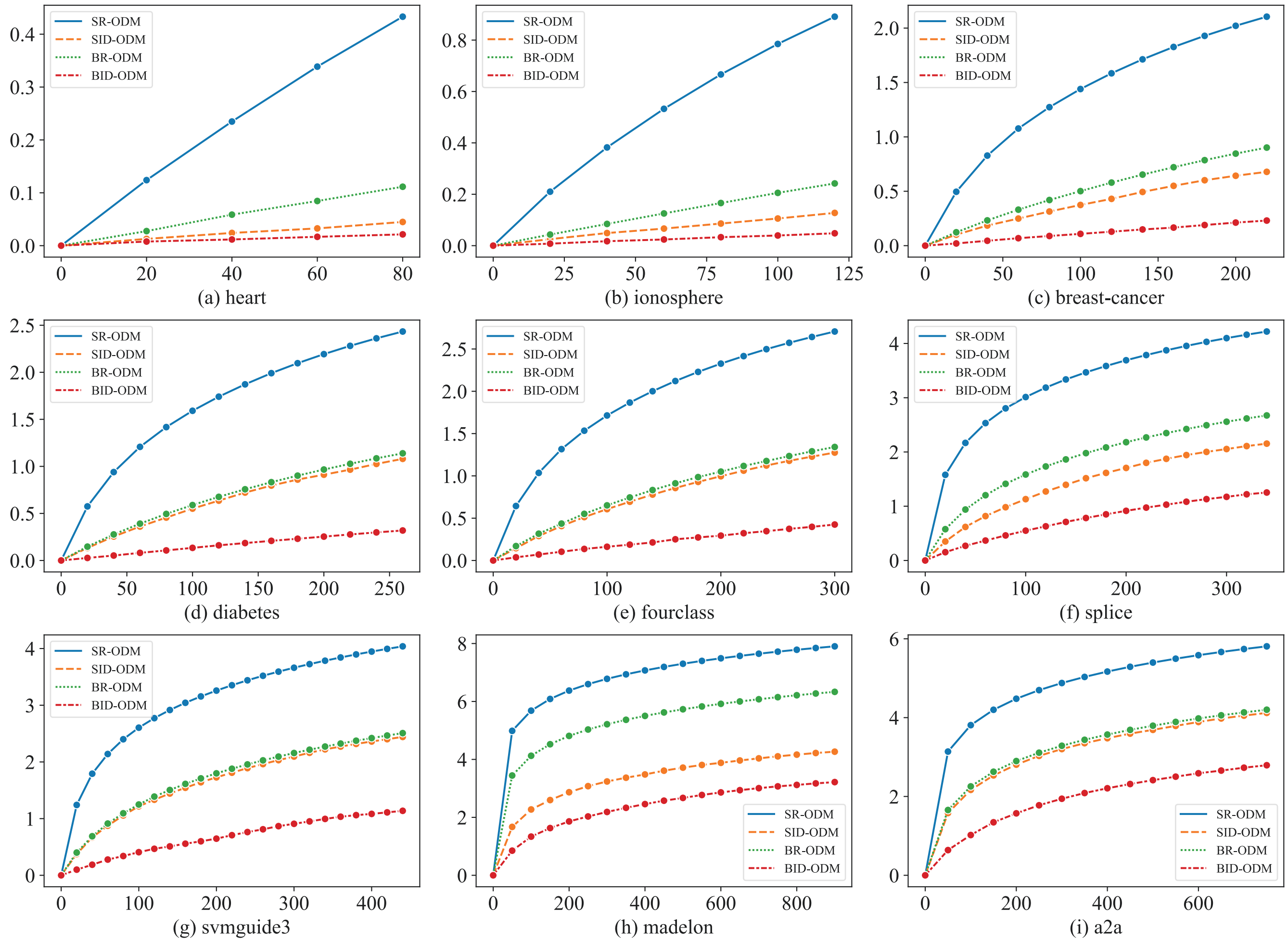


Figure 3: The cumulative running time of SR-ODM, BR-ODM, SID-ODM and BID-ODM on different data sets over 5 trials. The x-axis is the number of varying instances and the y-axis is the logarithm of cumulative running time in seconds.

ID	Data sets	#Instances	#Features
1	heart	270	13
2	ionosphere	351	34
3	breast	683	10
4	diabetes	768	8
5	fourclass	862	2
6	splice	1,000	60
7	svmguide3	1,243	21
8	madelon	2,000	500
9	a2a	2,265	123

Table 2: Characteristics of experimental data sets

Methods. To make an overall comparison, we implement *single incremental and decremental ODM (SID-ODM)* and *batch incremental and decremental ODM* (BID-ODM). We use historical data to train the initial ODM and all varying instances will be added to the model and then deleted. SID-ODM adds/deletes an instance at once while BID-ODM processes a batch of data. The batch size is set to 5. Apart from our proposed methods, we retrain the model for comparison, which is denoted as SR-ODM and BR-ODM corresponding to single and multiple varying instances, respectively. All methods are performed on a machine with the 11th Gen Intel(R) Core(TM) i5-11320H@3.20GHz CPUs and 16GB main memory. ODM is optimized by dual coordinate gradient descent.

Hyperparameters. The hyperparameters in our experiments contain three model hyperparameters $\{\lambda, \theta, \sigma\}$. λ and θ are tuned from $\{10^{-3}, \ldots, 10^{3}\}$ and $\{0.1, 0.2, \ldots, 0.9\}$, respectively by grid search. The kernel hyperparameter σ is fixed to $1/d$ where d is the number of features. All experiments are performed five times according to different data partitions.

Evaluation Measures. To verify the efficiency of our proposed methods, we compare the running time of each method. Besides, we also compare the accuracy on the test set to validate the effectiveness. We record the average accuracy as well as the standard deviation.

Data sets	Time (in seconds)				Speedup Ratio		
	SR-ODM	SID-ODM	BR-ODM	BID-ODM	SID/SR	BID/BR	BID/SID
heart	0.007±0.001	0.001±0.001	0.007±0.003	0.001±0.001	11.68	5.55	2.16
ionosphere	0.012±0.002	0.001±0.001	0.011±0.002	0.002±0.001	10.57	5.59	2.78
breast	0.033±0.005	0.004±0.003	0.033±0.005	0.006±0.004	7.50	5.50	3.61
diabetes	0.040±0.006	0.008±0.004	0.041±0.006	0.007±0.002	5.32	5.66	5.21
fourclass	0.047±0.007	0.009±0.004	0.047±0.007	0.009±0.005	5.42	5.33	4.88
splice	0.197±0.031	0.022±0.015	0.198±0.030	0.037±0.018	8.85	5.43	3.05
svmguide3	0.127±0.020	0.024±0.012	0.128±0.020	0.024±0.018	5.31	5.31	4.95
madelon	3.001±0.470	0.078±0.079	3.116±0.496	0.132±0.080	38.41	23.56	2.95
a2a	0.443±0.071	0.082±0.054	0.438±0.066	0.102±0.055	5.42	4.29	4.00

Table 3: The average running time of SR-ODM, BR-ODM, SID-ODM, and BID-ODM on each data set with the speedup ratio

5.2 Results

Efficiency Comparison. Figure 3 demonstrates the cumulative running time of conducting SID-ODM, BID-ODM, SR-ODM, and BR-ODM on each data set. It can be seen that both SID-ODM and BID-ODM cost much less time than retraining the model from scratch, which reveals the efficiency of our proposed method. Besides, it is worth noting that the running time of SR-ODM and BR-ODM varies significantly with the data size while SID-ODM and BID-ODM behave steadily, which benefits from the Sherman-Morrison-Woodbury formula.

Table 3 further shows the speedup ratio of our proposed methods compared to retraining the model. Since data sets have various characteristics, the boost of the performance differs slightly, but on all data sets, we can gain more than 4 times acceleration and even can reach $9.1\times$ speedup on average. Besides, there is no doubt that BID-ODM is more efficient than SID-ODM since it processes multiple instances at one time but with a closed-time cost to SID-ODM. The speedup ratio of BID-ODM compared to SID-ODM is also recorded in Table 3 to make an intuitive comparison.

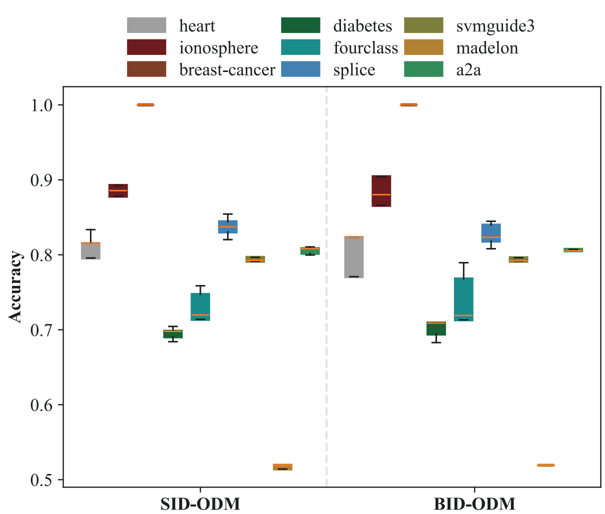


Figure 4: The accuracy of SID-ODM and BID-ODM on data sets

Generalization Performance Comparison. Figure 4 shows the accuracy over 5 trials with a box plot. We also plot the mean accuracy of retraining the model in the box. Intuitively, our proposed methods accelerate the update of ODM nearly without sacrificing accuracy, which validates the effectiveness of our proposed methods.

In order to further validate the performance of ID-ODM, we compare it with MID-SVM [Karasuyama and Takeuchi, 2009], which is another batch IDL method based on SVM. We incrementally add 100 instances to the training set and record the accuracy. As shown in Table 4, ID-ODM performs better than MID-SVM on most datasets, which benefits from optimizing margin distribution.

Data sets	ID-ODM	MID-SVM
heart	**81.5 ± 4.7**	81.1 ± 4.1
ionosphere	87.4± 6.6	**90.6 ± 3.8**
breast	**97.2 ± 0.9**	96.7 ± 1.2
diabetes	**76.9 ± 4.8**	76.6 ± 4.9
fourclass	**78.7 ± 2.8**	77.1 ± 1.9
splice	84.4 ± 3.1	**85.5 ± 3.6**
svmguide3	**80.5 ± 1.5**	79.9 ± 1.3
madelon	**58.0 ± 1.1**	56.7 ± 1.6
a2a	**83.0 ± 0.8**	**83.0 ± 0.8**
Avg. Acc.	81.1	80.8

Table 4: The accuracy of ID-ODM and MID-SVM

6 Conclusions

In this paper, we propose the ID-ODM to efficiently update ODM when new data is available or old data turns invalid. Specifically, we avoid updating the unbounded Lagrange multipliers in an infinite range by estimating the optimal value beforehand, thus iteratively updating the model. Moreover, ID-ODM is effective no matter how data varies, thus it is applicable to various real-world tasks. Besides, we provide some theoretical analysis on ID-ODM. Extensive experiments validate the effectiveness and efficiency of our method. In the future, we will extend our method to other models with *quadratic*-type loss.

Acknowledgments

This work was supported in part by the National Key R&D Program of China under Grant 2020AAA0108501, the National Natural Science Foundation of China under Grant 62006088, and the Key R&D Program of Hubei under Grant 2020BAA020.

References

[Boyd and Vandenberghe, 2004] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, United Kingdom, 2004.

[Cauwenberghs and Poggio, 2000] Gert Cauwenberghs and Tomaso A. Poggio. Incremental and decremental support vector machine learning. In *Advances in Neural Information Processing Systems*, pages 409–415, Denver, CO, 2000.

[Chen *et al.*, 2023] Haiyan Chen, Ying Yu, Yizhen Jia, and Bin Gu. Incremental learning for transductive support vector machine. *Pattern Recognition*, 133:10982, 2023.

[Cherubin *et al.*, 2021] Giovanni Cherubin, Konstantinos Chatzikokolakis, and Martin Jaggi. Exact optimization of conformal predictors via incremental and decremental learning. In *Proceedings of the 38th International Conference on Machine Learning*, pages 1836–1845, virtual, 2021.

[Cortes and Vapnik, 1995] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[Ertekin *et al.*, 2007] Seyda Ertekin, Jian Huang, Léon Bottou, and C. Lee Giles. Learning on the border: active learning in imbalanced data classification. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management*, pages 127–136, Lisbon, Portugal, 2007.

[Gâlmeanu and Andonie, 2022] Honorius Gâlmeanu and Razvan Andonie. Weighted incremental-decremental support vector machines for concept drift with shifting window. *Neural Networks*, 152(5):528–541, 2022.

[Gao and Zhou, 2013] Wei Gao and Zhi-Hua Zhou. On the doubt about margin explanation of boosting. *Artificial Intelligence*, 203:1–18, 2013.

[Golub and Van Loan, 1983] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, 1983.

[Gu *et al.*, 2012] Bin Gu, Jiandong Wang, Yuecheng Yu, Guansheng Zheng, Yu-Fan Huang, and Tao Xu. Accurate on-line ν-support vector learning. *Neural Networks*, 27:51–59, 2012.

[Gu *et al.*, 2015a] Bin Gu, Victor S. Sheng, Keng Yeow Tay, Walter Romano, and Shuo Li. Incremental support vector learning for ordinal regression. *IEEE Transactions Neural Networks Learning System*, 26(7):1403–1416, 2015.

[Gu *et al.*, 2015b] Bin Gu, Victor S. Sheng, Zhijie Wang, Derek Ho, Said Osman, and Shuo Li. Incremental learning for ν-support vector regression. *Neural Networks*, 67:140–150, 2015.

[Gu *et al.*, 2018] Bin Gu, Xiao-Tong Yuan, Songcan Chen, and Heng Huang. New incremental learning algorithm for semi-supervised support vector machine. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1475–1484, London, United Kingdom, 2018.

[Huang *et al.*, 2020] Sheng-Jun Huang, Guo-Xiang Li, Wen-Yu Huang, and Shao-Yuan Li. Incremental multi-label learning with active queries. *Journal of Computer Science and Technology*, 35(2):234–246, 2020.

[Jiang *et al.*, 2019] Hansi Jiang, Haoyu Wang, Wenhao Hu, Deovrat Kakde, and Arin Chaudhuri. Fast incremental SVDD learning algorithm with the gaussian kernel. In *Proceedings of 33rd AAAI Conference on Artificial Intelligence*, pages 3991–3998, Honolulu, HI, 2019.

[Karasuyama and Takeuchi, 2009] Masayuki Karasuyama and Ichiro Takeuchi. Multiple incremental decremental learning of support vector machines. In *Advances in Neural Information Processing Systems*, pages 907–915, Vancouver, Canada, 2009.

[Laskov *et al.*, 2006] Pavel Laskov, Christian Gehl, Stefan Krüger, and Klaus-Robert Müller. Incremental support vector learning: Analysis, implementation and applications. *Journal of Machine Learning Research*, 7(9):1909–1936, 2006.

[Laxhammar and Falkman, 2014] Rikard Laxhammar and Göran Falkman. Online learning and sequential anomaly detection in trajectories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1158–1173, 2014.

[Liang and Li, 2009] Zhizheng Liang and Youfu Li. Incremental support vector machine learning in the primal and applications. *Neurocomputing*, 72(10-12):2249–2258, 2009.

[Nguyen *et al.*, 2019] Thanh Tam Nguyen, Matthias Weidlich, Bolong Zheng, Hongzhi Yin, Quoc Viet Hung Nguyen, and Bela Stantic. From anomaly detection to rumour detection using data streams of social platforms. *Proceedings of the VLDB Endowment*, 12(9):1016–1029, 2019.

[Nguyen *et al.*, 2020] Quoc Phong Nguyen, Bryan Kian Hsiang Low, and Patrick Jaillet. Variational bayesian unlearning. In *Advances in Neural Information Processing Systems*, pages 16025–16036, virtual, 2020.

[Ouellette, 1981] Diane Valérie Ouellette. Schur complements and statistics. *Linear Algebra and its Applications*, 36:187–295, 1981.

[Rüping, 2001] Stefan Rüping. Incremental learning with support vector machines. In *Proceedings of the 1st IEEE International Conference on Data Mining*, pages 641–642, San Jose, CA, 2001.

[Schlimmer and Fisher, 1986] Jeffrey C. Schlimmer and Douglas H. Fisher. A case study of incremental concept induction. In *Proceedings of the 5th National Conference on Artificial Intelligence*, pages 496–501, Philadelphia, PA, 1986.

[Schölkopf *et al.*, 2000] Bernhard Schölkopf, Alexander J. Smola, Robert C. Williamson, and Peter L. Bartlett. New support vector algorithms. *Neural Computation*, 12(5):1207–1245, 2000.

[Sekhari *et al.*, 2021] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. In *Advances in Neural Information Processing Systems*, pages 18075–18086, Virtual, 2021.

[Syed *et al.*, 1999] Nadeem Ahmed Syed, Huan Liu, and Kah Kay Sung. Handling concept drifts in incremental learning with support vector machines. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 317–321, San Diego, CA, 1999.

[Tax and Duin, 2004] David M. J. Tax and Robert P. W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

[Wang and Vucetic, 2009] Zhuang Wang and Slobodan Vucetic. Fast online training of ramp loss support vector machines. In *Proceedings of the 9th IEEE International Conference on Data Mining*, pages 569–577, Miami, FL, 2009.

[Zhang and Zhou, 2019] Teng Zhang and Zhi-Hua Zhou. Optimal margin distribution machine. *IEEE Transactions on Knowledge and Data Engineering*, 32(6):1143–1156, 2019.

[Zhang, 2006] Fuzhen Zhang. *The Schur complement and its applications*. Springer Science and Business Media, Berlin, Germany, 2006.