Scalable Support Vector Machine for Semi-supervised Learning

Trung Le†, Khanh Nguyen†, Van Nguyen‡, Vu Nguyen†, Dinh Phung†
†Pattern Recognition and Data Analytics Centre
Deakin University, Australia
‡Faculty of Information Technology
HCMc University of Pedagogy, Hochiminh city, Vietnam

Abstract—Owing to the prevalence of unlabeled data, semi-supervised learning has recently drawn significant attention and has found applicable in many real-world applications. In this paper, we present the so-called Graph-based Semi-supervised Support Vector Machine (gS3VM), a method that leverages the excellent generalization ability of kernel-based method with the geometrical and distributed information carried in a spectral graph for semi-supervised learning purpose. The proposed gS3VM can be solved directly in the primal form using the Stochastic Gradient Descent method with the ideal convergence rate $O\left(\frac{1}{T}\right)$. Besides, our gS3VM allows the combinations of a wide spectrum of loss functions (e.g., Hinge, smooth Hinge, Logistic, L1, and ε-insensitive) and smoothness functions (i.e., $l_p(t) = |t|^p$ with $p \geq 1$). We note that the well-known Laplacian Support Vector Machine falls into the spectrum of gS3VM corresponding with the combination of the Hinge loss and the smoothness function $l_2(.)$. We further validate our proposed method on several benchmark datasets to demonstrate that gS3VM is appropriate for the large-scale datasets since it is optimal in memory used and yields superior classification accuracy whilst simultaneously achieving a significant computation speed-up in comparison with the state-of-the-art baselines.

Index Terms—Semi-supervised Learning, Kernel Method, Support Vector Machine, Spectral Graph, Stochastic Gradient Descent.

I. INTRODUCTION

Semi-supervised learning matters. Semi-supervised learning (SSL) aims at utilizing the intrinsic information carried in unlabeled data to enhance the generalization capacity of the learning algorithms. During the past decade, SSL has attracted significant attention and has found applicable in a variety of real-world problems including text categorization [11], image retrieval [2], bioinformatics [3], natural language processing [4] to name a few. While obtaining pre-defined labels is a labor-intensive and time-consuming process [5], ones have found that unlabeled data, when used in conjunction with a small amount of labeled data, can bring a remarkable improvement in classification accuracy [11].

A notable approach to semi-supervised learning paradigm is to employ spectral graph in order to represent the adjacent and distributive information carried in data. Graph-based methods are nonparametric, discriminative, and transductive in nature. The typical graph-based methods include min-cut [6, 7], harmonic function [8], graph random walk [9, 10], spectral graph transducer [11, 12], and manifold regularization [13].

Inspired from the pioneering work of [11], recent works have attempted to incorporate kernel methods such as Support Vector Machine (SVM) [14] with the semi-supervised learning paradigm. The underlying idea of this research line is to solve standard SVM problem while treating the unknown labels as optimization variables [5]. This leads to a non-convex optimization problem with a combinatorial explosion of label assignments. A wide spectrum of techniques have been proposed to solve this non-convex optimization problem, e.g., local combination search [1], gradient descent [15], continuation techniques [16], convex-concave procedures [17, 18], deterministic annealing [19, 20, 21], and semi-definite programming [22]. Although these works can somehow handle the combinatorial intractability, their common requirement to repeatedly retrain the model limits their applicability to real-world applications, hence lacking the ability to perform online learning for large-scale applications.

Conjoining the advantages of kernel method and the spectral graph theory, several existing works have tried to incorporate information carried in a spectral graph for building a better kernel function [23, 24, 25, 26, 27]. Basically, these methods employ the Laplacian matrix induced from the spectral graph to construct kernel functions which can capture the features of the ambient space. Manifold regularization framework [13] exploits the geometric information of the probability distribution that generates data and incorporates it as an additional regularization term. Two regularization terms are introduced to control the complexity of the classifier in the ambient space and the complexity induced from the geometric information of the distribution. However, the computational complexity for manifold regularization approach is cubic in the training size $n$ (i.e., $O(n^3)$). Hence other researches have been carried out to enhance the scalability of the manifold regularization framework [28, 29, 30]. Specifically, the work of [30] makes use of the preconditioned conjugate gradient to solve the optimization problem encountered in manifold regularization framework in the primal form. By that way, the computational complexity is reduced from $O(n^3)$ to $O(n^2)$. However, this approach is to actually solve the optimization problem in the first dual layer instead of the primal form, hence rendering the solution infeasible for online setting. In addition, the LapSVM in primal approach [30] requires to store the entire Hessian matrix of size $n \times n$ in the memory, resulting in a memory complexity of $O(n^2)$. Our evaluating experiments...
with LapSVM in primal show that it always consumes a huge amount of memory in its execution (cf. Tables VII and VIII). Recently, stochastic gradient descent (SGD) methods [31, 32, 33] have emerged as a promising framework to speed up the training process and enable the online learning paradigm. SGD possesses three key advantages: (1) fast; (2) ability to run in online mode; and (3) economic in memory usage. In this paper, we leverage three knowledge domains of kernel method, spectral graph theory and stochastic gradient descent to propose a novel approach to semi-supervised learning, termed as Graph-based Semi-supervised Support Vector Machine (gS3VM). Our gS3VM allows the combination of a wide spectrum of loss functions (cf. Section V) and smoothness functions $l_p(\cdot)$ where $p \geq 1$ (cf. Eq. (5)). We note that the well-known Laplacian Support Vector Machine (LapSVM) [13, 30] is the special case of gS3VM(s) when using Hinge loss and the smoothness function $l_2(\cdot)$. We then develop a new algorithm based on the SGD framework [33] to directly solve the optimization problem of gS3VM in its primal form with the ideal convergence rate $O\left(\frac{1}{T}\right)$. At each iteration, a labeled instance and an edge in the spectral graph are randomly sampled. As the result, the computational cost at each iteration is very economic and this makes the proposed method efficient to deal with large-scale datasets while maintaining comparable predictive performance.

To summarize, our contributions in this paper are as follows:

- We provide a novel view of jointly learning the kernel-based method with the spectral graph to propose gS3VM for semi-supervised learning purpose. Our proposed gS3VM enables the combination of a wide spectrum of convex loss functions and smoothness functions.
- We apply stochastic gradient descent (SGD) framework [33] to solve directly gS3VM in its primal form. Hence, gS3VM has all advantageous properties of SGD-based methods including fast computation, memory efficiency, and ability to run in online setting. Inheriting from the strength of SGD-based method, our gS3VM can perform online learning for large-scale applications. To our best of knowledge, the proposed gS3VM is the first semi-supervised learning method that can deal with the online learning context wherein data arrive continuously and sequentially.
- We provide a theoretical analysis to show that gS3VM has the ideal convergence rate $O\left(\frac{1}{T}\right)$ if using the loss function $l'(w; x, y)$ satisfying $\left\|l'(w; x, y)\right\|_\infty \leq A$ for some $A > 0$ and the smoothness function $l_p(\cdot) = |.|^p$ with $p \geq 1$. We verify that the necessary condition $\left\|l'(w; x, y)\right\|_\infty \leq A$ holds for a wide class of loss functions including Hinge, smooth Hinge, and Logistic for classification task and L1, $\varepsilon$-insensitive for regression task (cf. Section V).
- We validate our proposed method on several benchmark datasets. The experimental results empirically confirm the ideal convergence rate $O\left(\frac{1}{T}\right)$ of gS3VM and show that gS3VM is really scalable for large-scale datasets. In particular, it offers a comparable classification accuracy whilst achieving a significant computational speed-up in comparison with the state-of-the-art baselines.

II. RELATED WORK

We review the work in semi-supervised learning paradigm that are closely related to ours. Graph-based semi-supervised learning is an active research topic under semi-supervised learning paradigm. At its crux, graph-based semi-supervised methods define a graph where the vertices are labeled and unlabeled data of the training set and edges (may be weighted) reflect the similarity of data. Most of graph-based methods can be interpreted as estimating the prediction function $f$ such that: it should predict the labeled data as accurate as possible; and it should be smooth on the graph.

In [6, 7], semi-supervised learning problem is viewed as graph mincut problem. In the binary case, positive labels act as sources and negative labels act as sinks. The objective is to find a minimum set of edges whose removal blocks all flow from the sources to the sinks. Another way to infer the labels of unlabeled data is to compute the marginal probability of the discrete Markov random field. In [34], Markov Chain Monte Carlo sampling techniques is used to approximate this marginal probability. The work of [35] proposes to compute the marginal probabilities of the discrete Markov random field at any temperature with the Multi-canonical Monte Carlo method, which seems to be able to overcome the energy trap faced by the standard Metropolis or Swendsen-Wang method. The harmonic functions used in [8] is regarded as a continuous relaxation of the discrete Markov random field. It does relaxation on the value of the prediction function and makes use of the quadratic loss with infinite weight so that the labeled data are clamped. The works of [23, 24, 25, 26, 27] utilize the Laplacian matrix induced from the spectral graph to form kernel functions which can capture the features of the ambient space.

Yet another successful approach in semi-supervised learning paradigm is the kernel-based approach. The kernel-based semi-supervised methods are primarily driven by the idea to solve a standard SVM problem while treating the unknown labels as optimization variables [5]. This leads to a non-convex optimization problem with a combination explosion of label assignments. Many methods have been proposed to solve this optimization problem, for example local combinatorial search [11], gradient descent [15], continuation techniques [16], convex-concave procedures [17, 18], deterministic annealing [19, 20, 21], and semi-definite programming [22]. However, the requirement of retraining the whole dataset over and over preludes the applications of these kernel-based semi-supervised methods to the real-world datasets.

Some recent works on semi-supervised learning have primarily concentrated on the improvements of its safeness and classification accuracy. Li et al. [2015] assumes that the low-density assumption of S3VM [1]. Wang et al. [2015]...
extends \[13\] to propose semi-supervised discrimination-aware manifold regularization framework which considers the discrimination of all available instances in learning of manifold regularization. Tan et al. [2014] proposes using the $p$-norm as a regularization quantity in manifold regularization framework to perform dimension reduction in the context of semi-supervised learning.

The closest work to ours is manifold regularization framework \[13\] and its extensions \[28, 29, 30\]. However, the original work of manifold regularization \[13\] requires to invert a matrix of size $n$ by $n$ which costs cubically and hence is not scalable. Addressing this issue, Tsang et al. [29] scales up manifold regularization framework by adding $\varepsilon$-insensitive loss into the energy function, i.e., replacing $\sum w_{ij} (f(x_i) - f(x_j))^2$ by $\sum w_{ij} (|f(x_i) - f(x_j)|^2_\infty)$, where $|z|_\infty = \max \{|z| - \varepsilon, 0\}$. The intuition is that most pairwise differences $|f(x_i) - f(x_j)|$ are very small. By ignoring differences smaller than $\varepsilon$, the solution becomes sparser. LapSVM (in primal) \[30\] employs the preconditioned conjugate gradient to solve the optimization problem of manifold regularization in the primal form. This allows the computational complexity to be scaled up from $O(n^3)$ to $O(n^2)$. However, the optimization problem in \[30\] is indeed solved in the first dual layer rather than in the primal form. In addition, we empirically find that the LapSVM in primal \[30\] is unsatisfactory in terms of memory complexity. In particular, this method always consumes a huge amount of memory for computation (cf. Tables VII and VIII).

III. SPECTRAL-GRAPH-BASED SEMI-SUPERVISED LEARNING

A. Spectral Graph

Spectral graph is an useful tool to capture the geometrical and distributive information carried in the data. It is usually an undirected graph whose vertices are the data instances. In the context of semi-supervised learning, we are given a training set $X = X_l \cup X_u$ where $X_l = \{(x_i, y_i)\}_{i=1}^l$ is labeled data and $X_u = \{x_i\}_{i=1}^{l+u}$ is unlabeled data. We can start with constructing the spectral graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the vertex set $\mathcal{V}$ includes all labeled and unlabeled instances (i.e., $\mathcal{V} = \{x_i\}_{i=1}^{l+u}$). An edge $e_{ij} = x_i x_j \in \mathcal{E}$ between two vertices $x_i, x_j$ represents the similarity of the two instances. Let $\mu_{ij}$ be the weight of this $e_{ij}$. The principle is that if $\mu_{ij}$ is large, then $y_i, y_j$ are expected to receive the same label. The set of edges $\mathcal{G}$ and its weights can be built using the following ways:

- **Fully connected graph:** every pair of vertices $x_i, x_j$ is connected by an edge. The edge weight decreases when the distance $||x_i - x_j||$ increases. The Gaussian kernel weight function widely used is given by
  \[
  \mu_{ij} = e^{-\frac{||x_i - x_j||^2}{\sigma^2}}
  \]
  where $\sigma$ is known as the bandwidth parameter and controls how quickly the weight decreases.

- **k-NN:** each vertex $x_i$ determines its $k$ nearest neighbors (k-NN) and makes an edge with each of its $k$-NN. The Gaussian kernel weight function can be used for the edge weight. Empirically, k-NN graphs with small $k$ tend to perform well.

- **$\varepsilon$-NN:** we connect $x_i$ and $x_j$ if $||x_i - x_j|| \leq \varepsilon$. Again the Gaussian kernel weight function can be used to weight the connected edges. In practice, $\varepsilon$-NN graphs are easier to construct than $k$-NN graphs.

It is noteworthy that when constructing the spectral graph, we avoid connecting the edge of two labeled instances since we do not need to propagate the label between them. Figure 1 demonstrates an example of spectral graph constructed on 3D dataset using $k$-NN with $k = 5$.

B. Label Propagation

After building the spectral graph, a semi-supervised learning problem is formulated as assigning label to the unlabeled vertices. We need the mechanism to rationally propagate labels from the labeled vertices to the unlabeled ones. The idea is that if $\mu_{ij}$ is large, then the two labels $y_i, y_j$ are expected to be the same.

To assign labels to the unlabeled instances, it is desirable to learn a mapping function $f : \mathcal{X} \rightarrow \mathcal{Y}$ where $\mathcal{X}$ and $\mathcal{Y}$ are domains of data and label, respectively, such that

- $f(x_i)$ is as close as to its label $y_i$ as possible for all labeled instances $x_i$ ($1 \leq i \leq l$).
- $f$ should be smooth on the whole graph $\mathcal{G}$, i.e., if $x_i$ is very close to $x_j$ (i.e., $x_i, x_j$ are very similar or $\mu_{ij}$ is large), the discrepancy between $f_i$ and $f_j$ (i.e., $|f_i - f_j|$) is small.

Therefore, we arrive at the following optimization problem

\[
\min_f \left( \sum_{i=1}^l (f_i - y_i)^2 + \sum_{(i,j) \in \mathcal{E}} \mu_{ij} (f_i - f_j)^2 \right)
\]

where by convention we define $\sum_{i=1}^0 = 0$ and $f_i = f(x_i)$. The optimization problem in Eq. (1) peaks its minimum as the first term is exactly 0 and the second term is as smallest
as possible. It is therefore rewritten as follows

$$
\min_f \left( \sum_{(i,j) \in E} \mu_{ij} (f_i - f_j)^2 \right)
$$

To extend the representation ability of the prediction function $f$, we relax the discrete function $f$ to be real-valued. The drawback of the relaxation is that in the solution, $f(x)$ is now real-valued and hence does not directly correspond to a label. This can however be addressed by thresholding $f(x)$ at zero to produce discrete label predictions, i.e., if $f(x) \geq 0$, predict $y = 1$; and if $f(x) < 0$, predict $y = -1$.

IV. GRAPH-BASED SEMI-SUPERVISED SUPPORT VECTOR MACHINE (gS3VM)

In this section, we present our proposed Graph-based Semi-supervised Support Vector Machine (gS3VM). We start this section with introducing the optimization problem of gS3VM. We then describe SGD-based solution for gS3VM. Finally, we present the convergence analysis of gS3VM.

A. gS3VM Optimization Problem

Let $\Phi : \mathcal{X} \rightarrow \mathcal{H}$ be a transformation from the input space $\mathcal{X}$ to a Reproducing Hilbert Kernel Space (RHKS) $\mathcal{H}$. We use the function $f(x) = w^T \Phi (x) - \rho = \sum_{i=1}^{l} \alpha_i K(x_i, x) - \rho$, where $w = \sum_{i=1}^{l} \alpha_i \Phi(x_i)$ and $K(., .)$ is kernel function, to predict label. Inspired from the optimization problem in Eq. (2), the following optimization problem is proposed

$$
\min_w \left( \frac{1}{2} \|w\|^2 + \frac{C}{t} \sum_{i=1}^{l} \xi_i + \frac{C'}{|E|} \sum_{(i,j) \in E} \mu_{ij} (f_i - f_j)^2 \right)
$$

s.t. : $\forall_{i=1}^{l} : y_i (w^T \Phi(x_i) - \rho) \geq 1 - \xi_i$

$$
\forall_{i=1}^{l} : \xi_i \geq 0
$$

where $f_i = w^T \Phi(x_i) - \rho$.

In the optimization problem in Eq. (3), we minimize $\frac{1}{2} \|w\|^2$ to maximize the margin for motivating the generalization capacity. At the same time, we minimize $\sum_{(i,j) \in E} \mu_{ij} (f_i - f_j)^2$ to make the prediction function smoother on the spectral graph.

We rewrite the optimization problem in Eq. (3) in the primal form as follows

$$
\min_w \left( \frac{1}{2} \|w\|^2 + \frac{C}{t} \sum_{i=1}^{l} l(w; z_i) + \frac{C'}{|E|} \sum_{(i,j) \in E} \mu_{ij} l_p (w^T \Phi_{ij}) \right)
$$

where $z_i = (x_i, y_i)$, $l(w; x, y) = \max \{0, 1 - y w^T \Phi (x)\}$, $\Phi_{ij} = \Phi(x_i) - \Phi(x_j)$, $l_p(t) = |t|^p$ with $t \in \mathbb{R}$ and $p \geq 1$.

In the optimization problem in Eq. (4), the minimization of $\sum_{i=1}^{l} l(w; x_i, y_i)$ encourages the fitness of gS3VM on the labeled portion while the minimization of $\sum_{(i,j) \in E} \mu_{ij} l_p (w^T \Phi_{ij})$ guarantees the smoothness of gS3VM on the spectral graph. Naturally, we can extend the optimization of gS3VM by replacing the Hinge loss by any loss functions (e.g., Logistic, L1) and $l_2(\cdot)$ by $l_p(\cdot)$ with $p \geq 1$. We achieve the following optimization problem

$$
\min_w \left( \frac{1}{2} \|w\|^2 + \frac{C}{t} \sum_{i=1}^{l} l(w; z_i) + \frac{C'}{|E|} \sum_{(i,j) \in E} \mu_{ij} l_p (w^T \Phi_{ij}) \right)
$$

where $l(w; x, y)$ is any convex loss function, and $l_p(t) = |t|^p$ with $p \geq 1$.

It is noteworthy that Laplacian Support Vector Machine (LapSVM) [13, 30] is the special case of gS3VM using the Hinge loss with the smoothness function $l_2(\cdot)$.

B. Stochastic Gradient Descent Algorithm for gS3VM

We employ the SGD framework proposed in [35] to solve the optimization problem in Eq. (5) in the primal form. Let us denote the objective function as

$$
\mathcal{J}(w) \triangleq \frac{\|w\|^2}{2} + C l(w; x_i, y_i) + C' \mu_{uv} l_p (w^T \Phi_{uv})
$$

At the iteration $t$, we do the following:

- Uniformly sample a labeled instance $x_i$ ($1 \leq i \leq l$) from the labeled portion $X_l$ and an edge $(u, v)$ from the set of edges $E$.
- Define the instantaneous objective function

$$
\mathcal{J}_t (w_t) = \frac{\|w\|^2}{2} + C l(w; x_i, y_i) + C' \mu_{uv} l_p (w^T \Phi_{uv})
$$

- Define the stochastic gradient $g_t$

$$
g_t = \mathcal{J}_t'(w_t) = w_t + C l'(w_t; x_i, y_i) + C' \mu_{uv} l_p'(w_t^T \Phi_{uv})
$$

where $l'(w; x, y)$ specifies the derivative or sub-gradient w.r.t $w$ to $w$.

- Update $w_{t+1}$ where the learning rate $\eta_t = \frac{2}{t+1}$

$$
w_{t+1} = w_t - \eta_t g_t
$$

$$
w_{t+1} = \frac{t - 1}{t + 1} w_t - \frac{2C}{t + 1} l'(w_t; x_i, y_i)
$$

$$
= \frac{2C' \mu_{uv}}{t + 1} l_p'(w_t^T \Phi_{uv})
$$

- Update $w_{t+1}$ as follows

$$
w_{t+1} = \frac{t - 1}{t + 1} w_t + \frac{2}{t + 1} w_{t+1}
$$

We note that the derivative $l_p'(w^T \Phi) \text{ w.r.t } w$ can be computed as

$$
l_p'(w^T \Phi) = p \text{sign}(w^T \Phi) |w^T \Phi|^{p-1} \Phi
$$
Algorithm 1 Algorithm for gS3VM.

Input: \( C, C', p, K (\ldots) \)
1: \( w_1 = 0 \)
2: \( \bar{w}_1 = 0 \)
3: for \( t = 1 \) to \( T \) do
4: \( \) Uniformly sample \( i_t \) from \( \{1, 2, \ldots, l\} \) and \((u_t, v_t)\) from the set of edges \( E \)
5: \( )\) Update \( w_{t+1} \)
\[ w_{t+1} = \frac{t-1}{t+1} w_t - \frac{2C}{t+1} \Phi (w_t; x_i, y_i) \]
\[ - \frac{2C}{t+1} \mu_{u_t,v_t} i^*_t (w_t^T \Phi (u_t, v_t)) \]
6: \( \) Update \( \bar{w}_{t+1} \)
\[ \bar{w}_{t+1} = \frac{t-1}{t+1} \bar{w}_t + \frac{2}{t+1} w_{t+1} \]
7: end for

Output: \( w_{T+1} \)

The pseudocode of gS3VM is presented Algorithm 1. We note that we store \( w_t \) and \( \bar{w}_t \) as \( w_t = \sum_i \alpha_i \Phi (x_i) \) and \( \bar{w}_t = \sum_i \beta_i \Phi (x_i) \). In line 6 of Algorithm 1, the update of \( w_{t+1} \) involves the coefficients of \( \Phi (x_i), \Phi (x_{u_t}), \) and \( \Phi (x_{v_t}) \). In line 4 of Algorithm 1 we need to sample the edge \((u_t, v_t)\) from the set of edges \( E \) and compute the edge weight \( \mu_{u_t,v_t} \). It is noteworthy in gS3VM we use the fully connected spectral graph to maximize the freedom of label propagation and avoid the additional computation incurred in other kind of spectral graph (e.g., k-NN or \( \varepsilon \)-NN spectral graph). In addition, the edge weight \( \mu_{u_t,v_t} \) can be computed on the fly when necessary.

C. Convergence Analysis

In what follows, we present the convergence analysis for gS3VM. In particular, assuming that the loss function satisfies the condition: \( \|i^* (w; x, y)\| \leq A, \forall x, y \), we prove that our gS3VM achieves the ideal convergence rate \( O \left( \frac{1}{T} \right) \) with \( 1 \leq p < 2 \) and \( p \geq 2 \) under some condition of the parameters (cf. Theorem 5). We present the theoretical results and the rigorous proofs are given in Appendix A. Without loss of generality, we assume that the feature map \( \Phi (x) \) is bounded in the feature space, i.e., \( \|\Phi (x)\| = K (x, x)^{1/2} \leq R, \forall x \). We denote the optimal solution by \( w^* \), i.e., \( w^* = \arg \min_w J (w) \).

The following lemma shows the formula for \( \bar{w}_t \) from its recursive formula.

Lemma 1. We have the following statement
\[ \bar{w}_t = \frac{2}{t (t-1)} \sum_{i=1}^{t-1} i w_i \]

Lemma 2 offers the foundation to establish an upper bound for \( \|w_t\| \) in the next lemma.

Lemma 2. Let us consider the function \( f (x; a, b, p) = ax^{p-1} - x + b \) where \( x \geq 0 \) and \( p \geq 1 \), \( a, b > 0 \). The following statements guarantee

i) If \( p < 2 \) then \( f (M; a, b, p) \leq 0 \) where \( M = \max \left(1, \frac{b^2}{a^{1/p}} \right) \).
ii) If \( p = 2 \) and \( a < 1 \) then \( f (M; a, b, p) \leq 0 \) where \( M = \frac{b}{1-a} \).
iii) If \( p > 2 \) and \( ab^{p-2} \leq \frac{(p-2)p-2}{(p-1)^p} \) then \( f (M; a, b, p) \leq 0 \) where \( M = \left( \frac{1}{(p-1)a} \right)^{\frac{1}{p-2}} \).

Lemma 3 establishes an upper bound on \( \|w_t\| \) which is used to define the bound in the next lemma.

Lemma 3. We have the following statement
\[ \|w_t\| \leq M, \forall t \]

where \( M \) is defined as
\[ M = \begin{cases} \max \left(1, \frac{(a+b)^{1/p}}{p^{1/p}} \right) & \text{if } p < 2 \\ \left( \frac{1}{(p-1)a} \right)^{\frac{1}{p-2}} & \text{if } p > 2, \frac{ab^{p-2}}{(p-1)^p} \leq \left( \frac{p-2)p-2}{(p-1)^p} \right) \end{cases} \]

with \( a = C' (2R)^p \) and \( b = CA \).

Lemma 4 establishes an upper bound on \( \|g_t\| \) which is used in the main theorems.

Lemma 4. We have the following statement
\[ \|g_t\| \leq G, \forall t \]

where \( G = M + CA + C' (2R)^p \) \( pM^{p-1} \).

The following theorem shows the convergence rate \( O \left( \frac{1}{T} \right) \) of gS3VM.

Theorem 5. Considering the running of Algorithm 1 the following statements hold
i) \( \mathbb{E} [J (w_{T+1})] - J (w^*) \leq \frac{2G}{T}, \forall T \)
ii) \( \mathbb{E} \left[ \|w_{T+1} - w^*\|^2 \right] \leq \frac{4G^2}{T}, \forall T \)

if \( 1 \leq p < 2 \) or \( p = 2 \), \( a < 1 \) or \( p > 2 \), \( ab^{p-2} \leq \frac{(p-2)p-2}{(p-1)^p} \)

where \( a = C' (2R)^p \) and \( b = CA \).

Theorem 5 states the regret in the form of expectation. We go further to prove that for all \( T \geq T_0 = \left[ \frac{2G}{C^2} \right] \), with a high confidence level, \( J (w_{T+1}) \) approaches the optimal value \( J (w^*) \) within an \( \varepsilon \)-precision in Theorem 6.

Theorem 6. With the probability at least \( 1 - \delta, \) for all \( T \geq T_0 = \left[ \frac{2G}{C^2} \right] \), \( J (w_{T+1}) \) approximates the optimal value \( J (w^*) \) within an \( \varepsilon \)-precision, i.e., \( J (w_{T+1}) \leq J (w^*) + \varepsilon \)

if \( 1 \leq p < 2 \) or \( p = 2 \), \( a < 1 \) or \( p > 2 \), \( ab^{p-2} \leq \frac{(p-2)p-2}{(p-1)^p} \)

where \( a = C' (2R)^p \) and \( b = CA \).

V. Suitability of Loss Function and Kernel Function

In this section, we present the suitability of the loss functions and kernel functions that can be used for gS3VM. We verify that most of the well-known loss functions satisfying the necessary condition: \( \|i^* (w; x, y)\| \leq A \) for an appropriate positive number \( A \).
exponential kernel function which is given by
\[ \Phi(x) \]
that encloses all spheres within radius \( R \). To allow the ability to flexibly control the minimal sphere

\[ R = \min \{ a, \frac{1}{\sigma_f} \} \]

Therefore, by choosing \( A = R \) we have

\[ \| l(x) \| \leq \| A \| = A \]

**Smooth Hinge loss**

\[
l(w; x, y) = \begin{cases} 0 & \text{if } y > 1 \\ 1 - y w^T \Phi(x) & \text{if } y < 1 - \tau \\ \frac{1}{2\tau}(1 - y)^2 & \text{otherwise} \end{cases}
\]

\[
l'(w; x, y) = -\mathbb{I}_{y < 1 - \tau} y \Phi(x) + \tau^{-1} \mathbb{I}_{y \leq 1} (y - 1) y \Phi(x)
\]

where \( o = w^T \Phi(x) \). Therefore, by choosing \( A = R \) we have

\[ \| l'(w; x, y) \| \leq \| \Phi(x) \| \leq R = A \]

**Logistic loss**

\[
l(w; x, y) = \log(1 + \exp(-y w^T \Phi(x)))
\]

\[
l'(w; x, y) = -y \exp(-y w^T \Phi(x)) \frac{\Phi(x)}{\exp(-y w^T \Phi(x)) + 1}
\]

Therefore, by choosing \( A = R \) we have

\[ \| l'(w; x, y) \| < \| \Phi(x) \| \leq R = A \]

**L1 loss**

\[
l(w; x, y) = |y - w^T \Phi(x)|
\]

\[
l'(w; x, y) = \text{sign}(w^T \Phi(x) - y) \Phi(x)
\]

Therefore, by choosing \( A = R \) we have

\[ \| l'(w; x, y) \| \leq \| \Phi(x) \| \leq R = A \]

**\( \varepsilon \)-insensitive loss**

\[
l(w; x, y) = \max \{ 0, |y - w^T \Phi(x)| - \varepsilon \}
\]

\[
l'(w; x, y) = \mathbb{I}_{|y - w^T \Phi(x)| > \varepsilon} \text{sign}(w^T \Phi(x) - y) \Phi(x)
\]

Therefore, by choosing \( A = R \) we have

\[ \| l'(w; x, y) \| \leq \| \Phi(x) \| \leq R = A \]

Here \( \mathbb{I}_S \) is the indicator function which is equal to 1 if the statement \( S \) is true and 0 if otherwise.

It can be observed that the positive constant \( A \) coincides the radius \( R \) (i.e., \( A = R \)) for the aforementioned loss functions.

To allow the ability to flexibly control the minimal sphere that encloses all \( \Phi(x) \)'s, we propose to use the squared exponential kernel function which is given by

\[ K(x, x') = \sigma_f^2 e^{-\frac{|x - x'|^2}{2\sigma_f^2}} \] (6)

where \( \sigma_f \) is the length-scale parameter and \( \sigma_f \) is the output variance parameter.

It appears that using the kernel in Eq. (6), we have the following equality

\[ \| \Phi(x) \| = K(x, x)^{1/2} = \sigma_f \leq R \]

Recall that if \( p = 2 \) or \( p > 2 \), Algorithm [I] converges to the optimal solution with the ideal convergence rate \( O\left(\frac{1}{p}\right) \) under specific conditions. In particular, with \( p = 2 \) the corresponding condition is \( a < 1 \) and with \( p > 2 \) the corresponding condition is \( ab^{p-2} \leq \left(\frac{p-2}{p-1}\right)^{p-1} \) where \( a = C'/(2R)^p \) and \( b = CA \).

Using the squared exponential kernel function in Eq. (6), we can adjust the output variance parameter \( \sigma_f \) to make the convergent condition valid. More specifically, we consider two cases:

- \( p = 2 \): the convergent condition is derived as follows

\[ a < 1 \]

\[ C'/(2R)^p < 1 \]

\[ R < \frac{1}{2} \left(\frac{pC'}{2}\right)^{-1/p} \]

We can simply choose the output variance parameter \( \sigma_f \) as

\[ \sigma_f = R = \frac{1}{2} \left(\frac{pC'}{2}\right)^{-1/p} \]

where \( \rho > 0 \) is a very small number.

- \( p > 2 \): the convergent condition is derived as follows

\[ ab^{p-2} \leq \left(\frac{p-2}{p-1}\right)^{p-1} \]

\[ C'/(2R)^p < 1 \]

\[ R^{p-2} \leq \left(\frac{p-2}{2pC'\rho} - 1\right)^{\frac{1}{p-2}} \]

We can simply choose the output variance parameter \( \sigma_f \) as

\[ \sigma_f = R = \left(\frac{p-2}{2pC'\rho} - 1\right)^{\frac{1}{p-2}} \]

In case we wish to use the popular radial basic function (RBF) kernel which is given by

\[ K(x, x') = e^{-\frac{|x - x'|^2}{2\sigma_f^2}} \]

we can control the second trade-off parameter \( C' \) so as to ensure the ideal convergence rate \( O\left(\frac{1}{p}\right) \) with \( p \geq 2 \). More specifically, we also consider two cases:
\* \* \* p = 2: the convergent condition is derived as follows
\[ a < 1 \]
\[ C' < \frac{1}{(2R)^p} p = \frac{1}{8} \]  
(7)  
\* \* \* \* \* p > 2: the convergent condition is derived as follows
\[ a b^{p-2} \leq \frac{(p-2)^{p-2}}{(p-1)^{p-1}} \]
\[ C' (2R)^p (CA)^{p-2} \leq \frac{(p-2)^{p-2}}{(p-1)^{p-1}} \]
\[ C' (2R)^p (CR)^{p-2} \leq \frac{(p-2)^{p-2}}{(p-1)^{p-1}} \]
\[ C' \leq \frac{1}{2 p C^{p-2} p (p-1)^{p-1}} \]  
(8)  

VI. Experiments

We conduct the extensive experiments to investigate the influence of the parameters and factors to the model behavior and to compare our proposed gS3VM with the state-of-the-art baselines on the benchmark datasets. In particular, we design three kinds of experiments to analyze the influence of factors, e.g., the loss function \( l \), the smoothness function \( l_p \), and the percentage of unlabeled portion to the model behavior. In the first experiment (cf. Section VI-C1), we empirically prove the theoretical convergence rate \( O \left( \frac{1}{T} \right) \) for all combinations of loss function (Hinge, Logistic) and smooth function \( (p = 1, 2, 3) \) and we also investigate how the number of iterations affects the classification accuracy.

In the second experiment (cf. Section VI-C2), we study the influence of various combinations of the loss function and the smoothness function to the quality of prediction and the training time when the percentage of unlabeled portion is either 80% or 90%. In the third experiment (cf. Section VI-C3), we examine the proposed method under the semi-supervised setting where the proportion of unlabeled data is varied from 50% to 90%. Finally, we compare our proposed gS3VM with the state-of-the-art baselines on the benchmark datasets.

A. Experimental Datasets

We conduct the experiments on 13 benchmark dataset\(^2\) for semi-supervised learning. The statistics of the experimental datasets are given in Table 1.

| Dataset          | Size  | Dimension |
|------------------|-------|-----------|
| G50C             | 551   | 50        |
| COL20            | 145   | 1,014     |
| USPST            | 601   | 256       |
| AUSTRALIAN       | 690   | 14        |
| A1A              | 1,605 | 123       |
| MUSHROOMS        | 8,124 | 112       |
| SVMGUIDE3        | 1,243 | 21        |
| SVMGUIDE1        | 3,089 | 4         |
| W5A              | 9,888 | 300       |
| W5A              | 9,749 | 300       |
| CORKS            | 59,535| 8         |
| UCRN            | 49,990| 22        |
| COVTYPE          | 581,012| 54        |

Table 1  

The statistics of experimental datasets.

All compared methods are run on a Windows computer with the configuration of CPU Xeon 3.47 GHz and 96GB RAM. All codes of the baselines are achieved from the corresponding authors.

C. Impact of Incident Factors to The Learning Behavior

1) Simulation Study on Convergence Rate: In this simulation study, we want to empirically examine the convergence rate of gS3VM with various combinations of loss function (Hinge, Logistic) and smooth function \( (p = 1, 2, 3) \). We select two datasets G50C and USPST. For each dataset, we compute the quantity \( \Delta J^T = (J(w_{T+1}) - J(w^*)) * T \) and measure the accuracy when the number of iterations \( T \) is varied. We repeated each experiment five times to record the necessary quantities and their standard deviations. As observed from Figures 2 and 3, \( \Delta J^T \) tends to decrease and when \( T \) is sufficiently large, this quantity is lower-bounded by a constant.

Hence, we can empirically conclude the convergence rate \( O \left( \frac{1}{T} \right) \) of gS3VM. This conclusion also matches with the theoretical analysis developed in Section IV-C. It is noteworthy that in this simulation study we use the RBF kernel and with \( p = 2 \) and \( p = 3 \) the second trade-off parameter \( C' \) is selected using Eqs. (7, 8) so as to theoretically guarantee the ideal convergence rate \( O \left( \frac{1}{T} \right) \) of our gS3VM.

2) Simulation Study on Influence of Loss Function and Smoothness Function to The Learning Performance: In this simulation study, we answer the question how the variation in loss function and smoothness function affects the learning of the original LapSVM \(^{13}\) from \( O \left( n^3 \right) \) to \( O \left( kn^2 \right) \) using the preconditioned conjugate gradient and an early stopping strategy.

\* \* \* \* \* CCCP \(^{13}\): An approach is proposed to solve the non-convex optimization problem occurred in the kernel semi-supervised approach using convex-concave procedures.

\* \* \* Self-KNN: Self-training is one of the most classical technique using in semi-supervised classification. Self-KNN employs \( k \)-NN method as a core classifier for self training.

\* \* \* SVM: Support Vector Machine which is implemented using LIBSVM solver \(^{38}\) and trained with fully label setting. We use fully labeled SVM as a milestone to judge how good the semi-supervised methods are.

\^2Most of the experimental datasets can be conveniently downloaded from the URL https://www.csie.ntu.edu.tw/˜cjlin/libsvmtools/datasets/.

\^13Most of the experimental datasets can be conveniently downloaded from the URL https://www.csie.ntu.edu.tw/˜cjlin/libsvmtools/datasets/.
Figure 2. The variation of the accuracy and the quantity $\Delta JT = (J(w_{T+1}) - J(w^*)) \times T$ on the dataset G50C w.r.t. the number of iterations $T$. Hinge and Logistic losses are combined with $p = 1, 2, 3$. When $T$ increases, the accuracy gradually improves and the quantity $\Delta JT$ decreases to a constant.

Figure 3. The variation of the accuracy and the quantity $\Delta JT = (J(w_{T+1}) - J(w^*)) \times T$ on the dataset USPST w.r.t. the number of iterations $T$. Hinge and Logistic losses are combined with $p = 1, 2, 3$. When $T$ increases, the accuracy gradually improves and the quantity $\Delta JT$ gradually decreases to a constant.
Figure 4. Left: The classification accuracy when fraction of unlabeled data is varied. Right: The F1 score when fraction of unlabeled data is varied.

Figure 5. Left: Accuracy on the experimental datasets when hiding 80% label of data. Right: Accuracy on the experimental datasets when hiding 90% label of data.

Figure 6. Left: The F1 score on the experimental datasets when hiding 80% label of data. Right: The F1 score on the experimental datasets when hiding 90% label of data.
Figure 7. Left: The training time on the experimental datasets when hiding 80% label of data. Right: The training time on the experimental datasets when hiding 90% label of data.

### Table II

**The classification accuracy (%) and training time (Time (s)) of the proposed method when 80% of data are hidden label.**

| Dataset   | Accuracy (%) | Time (s) |
|-----------|--------------|----------|
|           | Hinge loss   | Logistic loss |
|           | p = 1 | p = 2 | p = 3 | p = 1 | p = 2 | p = 3 | p = 1 | p = 2 | p = 3 |
| A1A       | 80.69       | 81.00    | 79.54  | 80.37  | 80.17  | 80.37  | 0.078  | 0.073  | 0.073  |
| AUSTRALIAN| 91.30       | 91.30    | 90.58  | 91.30  | 90.10  | 90.58  | 0.026  | 0.021  | 0.021  |
| COIL20    | 100         | 100      | 100    | 100    | 100    | 100    | 0.010  | 0.010  | 0.015  |
| G50C      | 92.73       | 95.45    | 95.45  | 95.45  | 95.45  | 95.45  | 0.026  | 0.015  | 0.015  |
| SVM2      | 89.55       | 88.06    | 89.55  | 89.55  | 80.60  | 76.12  | 0.010  | 0.010  | 0.015  |
| SVM3      | 81.85       | 80.24    | 80.24  | 79.84  | 81.05  | 79.30  | 0.125  | 0.015  | 0.015  |
| USPS1     | 99.17       | 99.17    | 99.17  | 99.17  | 99.17  | 99.17  | 0.032  | 0.031  | 0.037  |
| COD-RNA  | 87.25       | 87.43    | 86.32  | 87.05  | 87.04  | 87.26  | 1.432  | 0.995  | 0.891  |
| COVTYPE   | 84.15       | 83.89    | 79.84  | 87.05  | 87.04  | 87.26  | 1.590  | 1.755  | 1.760  |
| IJCNN1    | 93.16       | 93.12    | 93.07  | 92.70  | 93.14  | 92.70  | 1.594  | 0.672  | 0.665  |
| W5A       | 97.57       | 97.59    | 97.52  | 97.61  | 97.69  | 97.39  | 0.041  | 0.031  | 0.047  |
| W8A       | 97.90       | 97.42    | 97.34  | 97.41  | 97.40  | 97.22  | 1.140  | 0.073  | 0.073  |
| MUSHROOM  | 99.96       | 99.94    | 99.98  | 100    | 99.98  | 99.98  | 0.042  | 0.031  | 0.036  |

### Table III

**The classification precision (%) and F1 (%) of the proposed method when 80% of data are hidden label.**

| Dataset   | Precision (%) | F1 score (%) |
|-----------|---------------|--------------|
|           | Hinge loss    | Logistic loss | Hinge loss    | Logistic loss |
|           | p = 1 | p = 2 | p = 3 | p = 1 | p = 2 | p = 3 | p = 1 | p = 2 | p = 3 |
| A1A       | 67.35       | 71.43      | 71.81  | 66.00  | 89.48  | 64.81  | 51.56  | 49.59  | 40.12  |
| AUSTRALIAN| 88.89       | 88.89      | 88.71  | 91.53  | 88.65  | 90.00  | 90.32  | 90.32  | 89.43  |
| COIL20    | 100         | 100        | 100    | 100    | 100    | 100    | 100    | 100    | 100    |
| G50C      | 88.52       | 96.30      | 96.30  | 96.30  | 96.30  | 96.30  | 93.10  | 95.41  | 95.41  |
| SVM2      | 90.00       | 100        | 100    | 100    | 100    | 67.86  | 83.72  | 78.95  | 80.25  |
| SVM3      | 79.17       | 85.71      | 81.25  | 83.70  | 92.86  | 75.64  | 45.78  | 32.88  | 34.67  |
| USPS1     | 100         | 100        | 100    | 100    | 100    | 90.10  | 99.08  | 99.08  | 99.08  |
| COD-RNA  | 79.55       | 78.94      | 77.64  | 78.95  | 79.03  | 78.48  | 81.29  | 81.84  | 80.16  |
| COVTYPE   | 82.40       | 82.14      | 77.22  | 76.06  | 76.92  | 76.83  | 89.40  | 88.95  | 86.65  |
| IJCNN1    | 99.55       | 91.62      | 89.51  | 90.79  | 91.37  | 94.03  | 48.78  | 47.45  | 47.61  |
| W5A       | 85.10       | 88.24      | 74.79  | 78.33  | 89.74  | 90.48  | 28.70  | 30.25  | 29.04  |
| W8A       | 88.97       | 75.57      | 76.14  | 82.62  | 80.13  | 61.93  | 48.75  | 30.82  | 24.58  |
| MUSHROOM  | 99.92       | 99.88      | 99.96  | 100    | 99.96  | 100    | 99.96  | 99.94  | 99.98  |
performance on the real datasets. We experiment on the real datasets given in Table I with different combinations of loss function (Hinge, Logistic) and smoothness function ($p = 1, 2, 3$). Each experiment is performed five times and the average accuracy, F1 score, precision, and training time corresponding to 80% or 90% of hidden label are reported in Tables V, VI, VIII, and IX. We observe that the Hinge loss is slightly better than Logistic one and the combination of Hinge loss and the smoothness $l_1$ is slightly better than other combinations. It is noteworthy that in this simulation study we use the RBF kernel and with $p = 2$ and $p = 3$ the second trade-off parameter $C'$ is selected using Eqs. (7) and (8) so as to theoretically guarantee the ideal convergence rate $O(1/K)$ of our gS3VM.

### Simulation Study on Influence on Percentage of Hidden Label to The Learning Performance

In this simulation study, we address the question how variation in percentage of hidden label influences the learning performance. We also experiment on the real datasets given in Table II with the various percentages of hidden label varied in the grid

| Dataset | Accuracy (%) | Time (s) |
|---------|--------------|----------|
| A1A     | 83.18        | 0.078    |
| AUSTRALIAN | 90.58    | 0.016    |
| COIL    | 100.00      | 0.015    |
| G50C    | 97.27        | 0.020    |
| SVM2    | 82.09        | 0.010    |
| SVM3    | 82.26        | 0.094    |
| USPST   | 100.00      | 0.037    |
| COD-RNA | 87.37        | 0.974    |
| COVTYPE | 85.22        | 1.588    |
| IJCNN1  | 92.84        | 0.641    |
| W5A     | 97.64        | 0.036    |
| W8A     | 98.00        | 0.677    |
| MUSHROOM | 99.92      | 0.172    |

### Table IV
The classification accuracy (%) and training time (Time) (second) of the proposed method when 90% of data are hidden label.

| Dataset | Hinge loss | Logistic loss |
|---------|------------|--------------|
| A1A     | 76.60      | 57.14        |
| AUSTRALIAN | 94.44    | 88.70        |
| COIL    | 100.00     | 100.00       |
| G50C    | 96.43      | 97.30        |
| SVM2    | 82.35      | 70.00        |
| SVM3    | 82.61      | 46.34        |
| USPST   | 100.00     | 100.00       |
| COD-RNA | 80.96      | 81.09        |
| COVTYPE | 85.18      | 89.50        |
| IJCNN1  | 88.20      | 45.03        |
| W5A     | 100.00     | 28.56        |
| W8A     | 87.62      | 52.95        |
| MUSHROOM | 99.92      | 99.92        |

### Table V
The classification precision (%) and F1 (%) of the proposed method when 90% of data are hidden label.

| Dataset | 50% | 60% | 70% | 80% | 90% |
|---------|-----|-----|-----|-----|-----|
| G50C    | 98.21 | 95.45 | 95.02 | 94.29 | 93.82 |
| COIL    | 99.50 | 99.67 | 99.50 | 99.40 | 99.44 |
| USPST   | 87.18 | 86.67 | 87.07 | 86.38 | 86.06 |
| AUSTRALIAN | 83.75 | 83.08 | 82.89 | 83.13 | 83.10 |
| IJCNN1  | 87.35 | 77.57 | 77.83 | 78.20 | 77.36 |
| W5A     | 91.128 | 88.40 | 88.11 | 87.29 | 87.92 |
| W8A     | 97.29 | 97.29 | 97.27 | 97.25 | 97.29 |
| COD-RNA | 97.39 | 97.42 | 97.36 | 97.40 | 97.36 |
| COVTYPE | 88.59 | 87.44 | 87.20 | 87.22 | 88.52 |

### Table VI
The classification accuracy (%), F1 score (%) when the fraction of unlabeled data is varied in 50%, 60%, 70%, 80%, and 90%.

---

Hinge loss and the smoothness is slightly better than Logistic one and the combination of loss function (Hinge, Logistic) and smoothness function as to theoretically guarantee the ideal convergence rate $O(1/K)$ of our gS3VM.

3) **Simulation Study on Influence on Percentage of Hidden Label to The Learning Performance**

In this simulation study, we address the question how variation in percentage of hidden label influences the learning performance. We also experiment on the real datasets given in Table II with the various percentages of hidden label varied in the grid.
| Dataset    | Method         | Accuracy | Precision | Recall | F1     | Time Used | Memory (MB of RAM) |
|------------|----------------|----------|-----------|--------|--------|-----------|--------------------|
| G50C       | gUS3VM         | 95.45    | 95.41     | 0.015  |        | 2.2       |                    |
|            | LapSVM         | 96.20    | 92.67     | 0.016  |        | 5.9       |                    |
|            | CCCP           | 98.18    | 98.18     | 0.141  |        | 1.6       |                    |
|            | Self-KNN       | 71.99    | 77.29     | 6.682  | 8.5    | 8.5       |                    |
|            | SVM            | 96.18    | 95.80     | 0.106  |        | 2.7       |                    |
| COIL20     | gUS3VM         | 100.00   | 100.00    | 0.010  |        | 2.7       |                    |
|            | LapSVM         | 100.00   | 100.00    | 0.039  |        | 11.0      |                    |
|            | CCCP           | 98.10    | 88.00     | 1.078  | 3.3    | 3.3       |                    |
|            | Self-KNN       | 93.18    | 89.76     | 7.968  |        | 3.3       |                    |
|            | SVM            | 100.00   | 100.00    | 0.090  |        | 2.9       |                    |
| USPST      | gUS3VM         | 99.17    | 99.08     | 0.031  |        | 3.9       |                    |
|            | LapSVM         | 99.20    | 99.09     | 0.282  |        | 9.3       |                    |
|            | CCCP           | 99.58    | 99.10     | 0.610  |        | 3.6       |                    |
|            | Self-KNN       | 99.82    | 99.80     | 35.406 |        | 3.3       |                    |
|            | SVM            | 99.80    | 99.81     | 0.490  |        | 5.0       |                    |
| AUSTRALIAN | gUS3VM         | 91.30    | 90.32     | 0.021  |        | 3.9       |                    |
|            | LapSVM         | 85.90    | 85.72     | 0.939  |        | 1.7       |                    |
|            | CCCP           | 81.88    | 81.20     | 0.002  |        | 3.6       |                    |
|            | Self-KNN       | 84.31    | 82.73     | 4.974  |        | 10.9      |                    |
|            | SVM            | 87.64    | 84.94     | 0.027  |        | 2.3       |                    |
| A1A        | gUS3VM         | 81.00    | 51.56     | 0.073  |        | 13.3      |                    |
|            | LapSVM         | 80.10    | 58.35     | 0.203  |        | 3.1       |                    |
|            | CCCP           | 79.75    | 46.28     | 0.953  |        | 0.7       |                    |
|            | Self-KNN       | 77.71    | 50.00     | 32.474 |        | 26.5      |                    |
|            | SVM            | 83.12    | 60.44     | 0.340  |        | 15.3      |                    |
| MUSHROOMS  | gUS3VM         | 99.39    | 99.34     | 0.031  |        | 48.9      |                    |
|            | LapSVM         | 98.80    | 98.89     | 5.260  |        | 336.6     |                    |
|            | CCCP           | 98.00    | 98.00     | 28.078 |        | 105.6     |                    |
|            | Self-KNN       | 82.92    | 82.07     | 550.590|        | 37.1      |                    |
|            | SVM            | 100.00   | 100.00    | 632.360|        | 105.6     |                    |
| SVMGUIDE3  | gUS3VM         | 80.24    | 33.88     | 0.015  |        | 1.5       |                    |
|            | LapSVM         | 75.80    | 31.93     | 0.330  |        | 2.9       |                    |
|            | CCCP           | 81.45    | 43.90     | 1.421  |        | 0.7       |                    |
|            | Self-KNN       | 88.24    | 47.84     | 18.368 |        | 10.4      |                    |
|            | SVM            | 83.67    | 36.71     | 0.230  |        | 6.4       |                    |
| W5A        | gUS3VM         | 97.69    | 33.82     | 0.041  |        | 102.5     |                    |
|            | LapSVM         | 97.00    | 40.01     | 1.181  |        | 81.1      |                    |
|            | CCCP           | 98.33    | 66.67     | 146.281|        | 251.2     |                    |
|            | Self-KNN       | 77.50    | 4.57     | 1,777.910|        | 84.2      |                    |
|            | SVM            | 98.49    | 67.25     | 48.060 |        | 105.4     |                    |
| W8A        | gUS3VM         | 97.42    | 48.75     | 0.013  |        | 110.0     |                    |
|            | LapSVM         | 97.40    | 35.94     | 26.145 |        | 17,550.0  |                    |
|            | CCCP           | 97.10    | 4.62     | 1,380.160|        | 276.7     |                    |
|            | Self-KNN       | 73.87    | 4.25     | 38,480.940|        | 387.5     |                    |
|            | SVM            | 98.82    | 77.42     | 64,650 |        | 115.7     |                    |
| COD-RNA    | gUS3VM         | 87.43    | 81.84     | 0.993  |        | 109.1     |                    |
|            | LapSVM         | 85.70    | 80.65     | 13.142 |        | 12,652.0  |                    |
|            | CCCP           | 88.48    | 82.84     | 3,900.430|        | 278.5     |                    |
|            | Self-KNN       | 61.43    | 43.16     | 1,370.120|        | 649.0     |                    |
|            | SVM            | 92.04    | 88.14     | 7,223.175|        | 117.2     |                    |
| IJCNN1     | gUS3VM         | 93.12    | 47.45     | 0.672  |        | 110.7     |                    |
|            | LapSVM         | 95.30    | 76.60     | 15.397 |        | 17,035.0  |                    |
|            | CCCP           | 93.09    | 34.39     | 6,718.450|        | 274.3     |                    |
|            | Self-KNN       | 92.93    | 48.28     | 2,988.740|        | 335.7     |                    |
|            | SVM            | 98.84    | 93.16     | 301.480|        | 117.1     |                    |
| COVTYPE    | gUS3VM         | 83.89    | 88.95     | 1.222  |        | 119.5     |                    |
|            | LapSVM         | 81.80    | 87.75     | 19.753 |        | 69,498.0  |                    |
|            | CCCP           | 85.91    | 89.96     | 5,958.070|        | 291.9     |                    |
|            | Self-KNN       | 64.30    | 32.02     | 19.731 |        | 892.0     |                    |
|            | SVM            | 90.06    | 93.16     | 3,535.547|        | 130.2     |                    |

Table VII

The classification accuracy (%), training time (Time) (second), F1 score, and used memory (MB) of the competitive methods when 80% of data are hid label.
| Dataset    | Method  | Accuracy | F1     | Time | Memory (MB of RAM) |
|------------|---------|----------|--------|------|--------------------|
|            | gUS3VM  | 96.36    | 96.43  | 0.015| 2.2                |
| G50C       | LapSVM  | 94.50    | 94.67  | 0.019| 5.9                |
|            | CCCP    | 94.55    | 96.43  | 0.509| 1.6                |
|            | Self-KNN| 63.58    | 67.30  | 3.104| 8.5                |
|            | SVM     | 96.18    | 95.80  | 0.106| 2.7                |
|            | LapSVM  | 100.00   | 100.00 | 0.005| 2.7                |
| COIL20     | LapSVM  | 100.00   | 100.00 | 0.016| 11.0               |
|            | CCCP    | 100.00   | 100.00 | 0.366| 1.9                |
|            | Self-KNN| 98.67    | 98.64  | 1.529| 3.3                |
|            | SVM     | 100.00   | 100.00 | 0.090| 2.9                |
|            | gUS3VM  | 100.00   | 100.00 | 0.031| 3.9                |
| USPST      | LapSVM  | 99.60    | 99.09  | 0.038| 9.3                |
|            | CCCP    | 100.00   | 100.00 | 2.170| 13.4               |
|            | Self-KNN| 99.57    | 99.53  | 16.690|17.3               |
|            | SVM     | 100.00   | 100.00 | 0.490| 5.0                |
|            | gUS3VM  | 90.10    | 88.77  | 0.026| 2.5                |
| AUSTRALIAN | LapSVM  | 86.20    | 85.71  | 0.032| 1.7                |
|            | CCCP    | 89.85    | 87.81  | 0.031| 0.6                |
|            | Self-KNN| 83.76    | 81.89  | 2.305| 10.9               |
|            | SVM     | 87.64    | 84.94  | 0.027| 2.3                |
|            | gUS3VM  | 83.49    | 59.96  | 0.078| 2.5                |
| A1A        | LapSVM  | 81.60    | 56.31  | 0.049| 3.1                |
|            | CCCP    | 82.37    | 56.92  | 0.047| 0.7                |
|            | Self-KNN| 77.63    | 51.69  | 30.724|26.5               |
|            | SVM     | 83.12    | 60.44  | 0.340| 15.3               |
|            | gUS3VM  | 99.96    | 99.09  | 0.033| 48.9               |
| MUSHROOMS  | LapSVM  | 97.50    | 97.70  | 0.334| 336.6              |
|            | CCCP    | 99.96    | 100.00 | 8.820| 177.1              |
|            | Self-KNN| 83.97    | 82.07  | 6625.850|37.1              |
|            | SVM     | 100.00   | 100.00 | 632.360|105.6             |
|            | gUS3VM  | 81.05    | 40.51  | 0.015| 1.5                |
| SVMGUIDE3  | LapSVM  | 77.90    | 26.16  | 0.028| 2.9                |
|            | CCCP    | 83.37    | 34.67  | 0.054| 0.7                |
|            | Self-KNN| 91.28    | 41.31  | 19.210|10.4               |
|            | SVM     | 83.67    | 56.71  | 0.230| 6.4                |
|            | gUS3VM  | 97.61    | 30.35  | 0.032| 102.5              |
| W5A        | LapSVM  | 97.50    | 28.62  | 0.521| 811.3              |
|            | CCCP    | 97.39    | 66.67  | 7.410| 251.2              |
|            | Self-KNN| 65.34    | 41.16  | 1,001.300|84.15             |
|            | SVM     | 98.49    | 67.25  | 48.060|105.4              |
|            | gUS3VM  | 98.00    | 52.95  | 0.677| 110                |
| W8A        | LapSVM  | 97.32    | 22.23  | 9.150| 17,750.0           |
|            | CCCP    | 97.18    | 7.17   | 379.060|276.7             |
|            | Self-KNN| 71.06    | 53.00  | 27,502.260|387.5         |
|            | SVM     | 98.82    | 77.42  | 64.650|115.7              |
|            | gUS3VM  | 87.51    | 81.05  | 1.115| 110                |
| COD-RNA    | LapSVM  | 86.10    | 79.19  | 11.420|12,652.0           |
|            | CCCP    | 89.74    | 82.70  | 326.720|278.5             |
|            | Self-KNN| 63.26    | 45.73  | 27,568.110|387.5          |
|            | SVM     | 92.04    | 88.14  | 7,223.175|117.2            |
|            | gUS3VM  | 92.62    | 41.93  | 0.729| 110                |
| IJCNN1     | LapSVM  | 80.90    | 48.72  | 8.080| 17,015.0           |
|            | CCCP    | 93.29    | 0.56   | 302.810|274.3             |
|            | Self-KNN| 91.97    | 39.51  | 12,739.030|335.7          |
|            | SVM     | 98.84    | 93.93  | 301.480|117.1             |
|            | gUS3VM  | 85.29    | 89.63  | 1.510| 119.5              |
| COVTYPE    | LapSVM  | 80.20    | 86.35  | 34.020|69,498.0           |
|            | CCCP    | 85.75    | 89.77  | 1,275.230|26.0             |
|            | Self-KNN| 70.02    | 69.81  | 47,875.090|892.0           |
|            | SVM     | 90.06    | 93.16  | 3,535.547|130.2             |

Table VIII

The classification accuracy (%), training time (Time) (second), F1 score, and used memory (MB) of the competitive methods when 90% of data are hid label.
We observe that when the percentage of hidden label increases, the classification accuracy and the F1 score tend to decrease across the datasets except for two datasets COIL20 and MUSHROOMS which remain fairy stable (cf. Table VI and Figure 4 (left and right)). This observation is reasonable since when increasing the percentage of hidden label, we decrease the amount of information label provided to the classifier and hence make the label propagation more challenging.

D. Experimental Results on The Benchmark Datasets

In this experiment, we compare our proposed method with the baselines, including LapSVM, CCCP, Self-KNN and SVM as described in VI-B. Based on the observation from the experiment in VI-C2, we use the combination of Hinge loss and the smoothness function $l_1(.)$. Besides offering the best predictive performance, this combination also encourages the sparsity in the output solution.

The standard RBF kernel given by $K(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}}$ is used in the experiments. With LapSVM, we use the parameter settings proposed in [30], wherein the parameters $\gamma_A$ and $\gamma_B$ are searched in the grid $\{10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}, 1, 10, 100\}$. In all experiments with the LapSVM, we make use of the preconditoned conjugate gradient version, which seems more suitable for the LapSVM optimization problem [30]. With CCCP-TSVM, we use the setting CCCP-TSVM $\hat{\gamma} = l_{C_{U3SVM}}$. Only two parameters need to be tuned are the trade-off $\gamma$ and the width of kernel $\gamma$. Akin to our proposed gS3VM, the trade-off parameters $C' = C$ is tuned in the grid $\{2^{-5}, 2^{-3}, \ldots, 2^{3}, 2^{5}\}$ and the width of kernel $\gamma$ is varied in the grid $\{2^{-5}, 2^{-3}, \ldots, 2^{3}, 2^{5}\}$. In our proposed gS3VM, the bandwidth $\sigma_s$ of Gaussian kernel weight function, which involves in computing the weights of the spectral graph, is set so as to $\sigma_s = \sigma_l$. We run cross-validation with 5 folds. The optimal parameter set is selected according to the highest classification accuracy. We set the number of iterations $T$ in gS3VM to $0.2 \times (l + u)$ for the large-scale datasets such as MUSHROOMS, W5A, W8A, COD-RNA, COVTYPE and IJCNN1 and to $l + u$ for the remaining datasets. Each experiment is carried out five times to compute the average of the reported measures.

We measure the accuracy, F1 score, training time, and memory amount used in training when the percentage of hidden label is 80% and 90%. These measures are reported in Table VII and VIII. To improve the readability, in these two tables we emphasize the best method (not count the full-labeled SVM) for each measure using boldface, italicizing, or underlining. Regarding the classification accuracy and the F1 score, it can be seen that gS3VM are comparable with LapSVM and CCCP while being much better than Self-KNN. Particularly, CCCP seems to be the best method on 80% unlabeled dataset (cf. Figure 5 (left) and Figure 6 (left)) while gS3VM a little outperforms others on 90% unlabeled datasets (cf. Figure 5 (right) and Figure 6 (right)). Comparing with the full-labeled SVM, except for three datasets IJCNN1, COD-RNA, and COVTYPE, gS3VM produces the comparable classification accuracies. Remarkably for the computational time, gS3VM outperforms the baselines by a wide margin especially for the large-scale datasets. On the large-scale datasets W8A, COD-RNA, IJCNN1, and COVTYPE, gS3VM is significantly tens of times faster than LapSVM, the second fastest method (cf. Figure 7 (left) and Figure 7 (right)). We also examine the memory consumption in training for each method. It can be observed that gS3VM is also economic in terms of memory amount used in training especially for the large-scale datasets (cf. Figure 8). In contrast, LapSVM always consumes a huge amount of memory during its training.

![Figure 8. Memory used in training on the experimental datasets when hiding 80% or 90% label of data.](image)

VII. Conclusion

In this paper, we present the novel framework for semi-supervised learning, called Graph-based Semi-supervised Support Vector Machine (gS3VM). Our framework connects three domains of kernel method, spectral graph, and stochastic gradient descent at the right place. Our gS3VM can be solved directly in the primal form with the ideal convergence rate $O\left(\frac{1}{T}\right)$ and naturally inherits all strengths of a SGD-based method. We validate and compare gS3VM with other state-of-the-art methods under semi-supervised learning context on several benchmark datasets. The experimental results demonstrate that our proposed gS3VM offers comparable classification accuracy and is economic in memory amount used whilst achieving a significant speed-up comparing with the state-of-the-art baselines. Moreover, our approach is the first semi-supervised model offering the online setting that is essential in real-world application.
REFERENCES

[1] T. Joachims, “Transductive inference for text classification using support vector machines,” in International Conference on Machine Learning (ICML), Bled, Slovenia, 1999, pp. 200–209.

[2] L. Wang, K. L. Chan, and Z. Zhang, “Bootstrapping svm active learning by incorporating unlabelled images for image retrieval.” IEEE Computer Society, 2003, pp. 629–634.

[3] N. Kasabov, D. Zhang, and P. Pang, “Incremental learning in autonomous systems: evolving connectionist systems for on-line image and speech recognition,” in Advanced Robotics and its Social Impacts, 2005. IEEE Workshop on, june 2005, pp. 120 – 125.

[4] C. Goutte, H. Déjean, E. Gaussier, N. Canciedda, and J.-M. Renders, “Combining labelled and unlabelled data: A case study on fisher kernels and transductive inference for biological entity recognition,” in Proceedings of the 6th Conference on Natural Language Learning - Volume 20, ser. COLING-02, 2002, pp. 1–7.

[5] O. Chapelle, V. Sindhwani, and S. Keerthi, “Optimization techniques for semi-supervised support vector machines,” Journal of Machine Learning Research, vol. 9, pp. 203–233, Jun. 2008.

[6] A. Blum and S. Chawla, “Learning from labeled and unlabeled data using graph mincuts,” in Proceedings of the Eighteenth International Conference on Machine Learning, ser. ICML ’01, 2001, pp. 19–26.

[7] A. Blum, J. D. Lafferty, M. R. Rwebangira, and R. Reddy, “Semi-supervised learning using randomized mincuts.” in ICML, vol. 69, 2004.

[8] X. Zhu, Z. Ghahramani, and J. D. Lafferty, “Semisupervised learning using gaussian fields and harmonic functions.” in IN ICMIL, 2003, pp. 912–919.

[9] M. Szummer and T. Jaakkola, “Partially labeled classification with markov random walks,” in Advances in Neural Information Processing Systems. MIT Press, 2002, pp. 945–952.

[10] A. Azran, “The rendezvous algorithm: Multiclass semi-supervised learning with markov random walks,” in Proceedings of the 24th International Conference on Machine Learning, ser. ICML ’07, 2007, pp. 49–56.

[11] T. Joachims, “Transductive learning via spectral graph partitioning,” in In ICML, 2003, pp. 290–297.

[12] P. Duong, V. Nguyen, M. Dinh, T. Le, D. Tran, and W. Ma, “Graph-based semi-supervised support vector data description for novelty detection,” in 2015 International Joint Conference on Neural Networks (IJCNN), July 2015, pp. 1–6.

[13] M. Belkin, P. Niyogi, and V. Sindhwani, “Manifold regularization: A geometric framework for learning from labeled and unlabeled examples,” J. Mach. Learn. Res., vol. 7, pp. 2399–2434, Dec. 2006.

[14] C. Cortes and V. Vapnik, “Support-vector networks,” in Machine Learning, 1995, pp. 273–297.

[15] O. Chapelle and A. Zien. (2005) Semi-Supervised Classification by Low Density Separation.

[16] O. Chapelle, M. Chi, and A. Zien, “A continuation method for semi-supervised svms,” in Proceedings of the 23rd international conference on Machine learning, ser. ICML ’06. ACM, 2006, pp. 185–192.

[17] G. Fung and O. Mangasarian, “Semi-supervised support vector machines for unlabeled data classification,” Optimization Methods and Software, vol. 15, pp. 29–44, 2001.

[18] R. Collobert, F. Sinz, J. Weston, L. Bottou, and T. Joachims, “Large scale transductive svms,” Journal of Machine Learning Research, 2006.

[19] V. Sindhwani, S. Keerthi, and O. Chapelle, “Deterministic annealing for semi-supervised kernel machines,” in Proceedings of the 23rd international conference on Machine learning, ser. ICML ’06, 2006, pp. 841–848.

[20] T. Le, D. Tran, T. Tran, K. Nguyen, and W. Ma, “Fuzzy entropy semi-supervised support vector data description,” in 2013 International Joint Conference on Neural Networks (IJCNN), Aug 2013, pp. 1–5.

[21] V. Nguyen, T. Le, T. Pham, M. Dinh, and T. H. Le, “Kernel-based semi-supervised learning for novelty detection,” in 2014 International Joint Conference on Neural Networks (IJCNN), July 2014, pp. 4129–4136.

[22] T. De Bie and N. Cristianini, “Semi-supervised learning using semi-definite programming,” in Semi-supervised Learning, Cambridge, MA, 2006.

[23] O. Chapelle, J. Weston, and B. Schölkopf, “Cluster kernels for semi-supervised learning,” in Advances in Neural Information Processing Systems 15, S. Becker, S. Thrun, and K. Obermayer, Eds. MIT Press, 2003, pp. 601–608.

[24] C. Kemp, T. L. Griffiths, S. Stromsten, and J. B. Tenenbaum, “Semi-supervised learning with trees,” in Advances in Neural Information Processing Systems 16, S. Thrun, L. Saul, and B. Schölkopf, Eds., 2004, pp. 257–264.

[25] R. I. Kondor and J. D. Lafferty, “Diffusion kernels on graphs and other discrete input spaces.” Morgan Kaufmann Publishers Inc., 2002, pp. 315–322.

[26] A. J. Smola and I. R. Kondor, “Kernels and regularization on graphs.” in Proceedings of the Annual Conference on Computational Learning Theory, 2003.

[27] X. Zhu, J. S. Kandola, Z. Ghahramani, and J. D. Lafferty, “Nonparametric transforms of graph kernels for semi-supervised learning,” 2004.

[28] V. Sindhwani and P. Niyogi, “Linear manifold regularization for large scale semi-supervised learning,” in Proc. of the 22nd ICML Workshop on Learning with Partially Classified Training Data, 2005.

[29] I. W. Tsang and J. T. Kwok, “Large-scale sparsified manifold regularization.” MIT Press, 2006, pp. 1401–1408.

[30] S. Melacci and M. Belkin, “Laplacian support vector machines trained in the primal,” J. Mach. Learn. Res., vol. 12, Jul. 2011.

[31] S. Shalev-shwartz and Y. Singer, “Logarithmic regret algorithms for strongly convex repeated games,” in The Hebrew University, 2007.

[32] S. Kakade and Shalev-Shwartz, “Mind the duality gap:
Logarithmic regret algorithms for online optimization,” in *NIPS*, 2008.

[33] S. Lacoste-Julien, M. W. Schmidt, and F. Bach, “A simpler approach to obtaining an o(1/t) convergence rate for the projected stochastic subgradient method,” *CoRR*, 2012.

[34] X. Zhu and Z. Ghahramani, “Towards semi-supervised classification with markov random fields,” 2002.

[35] G. Getz, N. Shental, and E. Domany, “Semi-supervised learning – a statistical physics approach,” *CoRR*, 2006.

[36] F. Wu, W. Wang, Y. Yang, Y. Zhuang, and F. Nie, “Classification by semi-supervised discriminative regularization,” *Neurocomputing*, vol. 73, pp. 1641 – 1651, 2010.

[37] S. Shalev-Shwartz and T. Zhang, “Stochastic dual coordinate ascent methods for regularized loss,” *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 567–599, 2013.

[38] C.-C. Chang and C.-J. Lin, “Libsvm: A library for support vector machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27:27, May 2011.
Proof of Lemma 1

We prove that

\[ t (t - 1) \mathbf{w}_t = 2 \sum_{i=2}^{t} (i - 1) \mathbf{w}_{i-1} = \frac{2}{t-1} \sum_{i=1}^{t-1} i \mathbf{w}_i \]

Proof of Lemma 2

We prove three cases as follows

i) In this case, we have

\[ f (M; a, b, p) = aM^{p-1} - M + b = M^{p-1} (a - M^{2-p}) + b \]

Since \( M = \max \left( 1, (a + b)^{\frac{1}{2-p}} \right) \), we have \( M^{p-1} > 1 \) and \( M^{2-p} \geq a > b \). Hence, we gain

\[ f (M; a, b, p) < a - M^{2-p} + b \leq 0 \]

ii) With \( p = 2 \) and \( a < 1 \), we have \( M = \frac{b}{1-a} > 0 \) and

\[ f (M; p, a, b) = (a - 1) M + b \]

iii) In this case, we have

\[ f (M; a, b, p) = \frac{a}{(p-1)^{\frac{1}{p-2}} a^{\frac{1}{p-2}}} - \frac{1}{(p-1)^{\frac{1}{p-2}} a^{\frac{1}{p-2}}} + b \]

\[ = a - a (p-1)^{\frac{1}{p-2}} a^{\frac{1}{p-2}} \]

\[ = a \left( 2 - p + a \frac{1}{p-2} b (p-1)^{\frac{1}{p-2}} \right) \]

\[ = \frac{(p-1)^{\frac{1}{p-2}} a^{\frac{1}{p-2}}}{(p-1)^{\frac{1}{p-2}} a^{\frac{1}{p-2}}} \]

\[ = \frac{a \left( (ab)^{p-2} (p-1)^{p-1} \right)^{\frac{1}{p-2}}}{(p-1)^{\frac{1}{p-2}} a^{\frac{1}{p-2}}} \]

\[ \leq 0 \]

Proof of Lemma 3

We prove that \( \| \mathbf{w}_t \| \leq M \) for all \( t \) by induction. Assume the hypothesis holds with \( t \), we verify it for \( t + 1 \). We start with

\[ \mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t g_t \]

\[ = \frac{t-1}{t+1} \mathbf{w}_t - \frac{2C}{t+1} \left( \mathbf{w}_t; x_{i_t}, y_{i_t} \right) \]

\[ - \frac{2C' \mu_{u,v_t}}{t+1} \left( \mathbf{w}_t^T \Phi_{u,v_t} \right) \]

We have

\[ \| \mathbf{w}_{t+1} \| \leq \frac{t-1}{t+1} \| \mathbf{w}_t \| + \frac{2C}{t+1} \left\| \mathbf{w}_t \left( x_{i_t}, y_{i_t} \right) \right\| \]

\[ + \frac{2C' \mu_{u,v_t}}{t+1} \left\| \mathbf{w}_t^T \Phi_{u,v_t} \right\| \]

\[ \leq \frac{t-1}{t+1} \| \mathbf{w}_t \| + \frac{2CA}{t+1} + \frac{2C' \mu_{u,v_t}}{t+1} \left( \mathbf{w}_t^T \Phi_{u,v_t} \right) \]

Recall that

\[ l_p' \left( \mathbf{w}_t^T \Phi_{u,v_t} \right) = p \text{sign} \left( \mathbf{w}_t^T \Phi_{u,v_t} \right) \left| \mathbf{w}_t^T \Phi_{u,v_t} \right|^{p-1} \Phi_{u,v_t} \]

Hence, we gain

\[ \left\| l_p' \left( \mathbf{w}_t^T \Phi_{u,v_t} \right) \right\| \leq 2Rp \left| \mathbf{w}_t^T \Phi_{u,v_t} \right|^{p-1} \left\| \mathbf{w}_t \right\|^{p-1} \]

\[ \leq (2R)^p \left\| \mathbf{w}_t \right\|^{p-1} \]

Here we note that \( \| \Phi_{u,v_t} \| = \| \Phi (x_{u,v_t}) \| \leq 2R \).

Therefore, we gain the following inequality

\[ \| \mathbf{w}_{t+1} \| \leq \frac{t-1}{t+1} \| \mathbf{w}_t \| + \frac{2CA}{t+1} + \frac{2C' \mu_{u,v_t}}{t+1} \left( \mathbf{w}_t^T \Phi_{u,v_t} \right) \]

\[ \leq \frac{t-1}{t+1} \| \mathbf{w}_t \| + \frac{2b}{t+1} + \frac{2a}{t+1} \left\| \mathbf{w}_t \right\|^{p-1} \]

where we denote \( a = C' (2R)^p \) and \( b = CA \).

Recall that we define \( M \) as

\[ M = \begin{cases} \max \left( 1, (a + b)^{\frac{1}{2-p}} \right) & \text{if } p < 2 \\ \left( \frac{1}{(p-1)a} \right)^{\frac{2}{p-2}} & \text{if } p > 2, ab^{p-2} \leq \left( \frac{p-2}{p-1} \right)^{p-2} \end{cases} \]

Referring to Lemma 3 we possess \( f (M; p, a, b) \leq 0 \). We then have

\[ \| \mathbf{w}_{t+1} \| \leq \frac{t-1}{t+1} M + \frac{2b}{t+1} + \frac{2a M^{p-1}}{t+1} \]

\[ \leq M + \frac{2a M^{p-1}}{t+1} \]

\[ \leq M + \frac{f (M; p, a, b)}{t+1} \leq M \]

Proof of Lemma 4

To bound \( \| g_t \| \), we derive as

\[ \| g_t \| \leq \| \mathbf{w}_t \| + C \left( \| l_p' \left( \mathbf{w}_t; x_{i_t}, y_{i_t} \right) \| + C' \mu_{u,v_t} \left\| l_p' \left( \mathbf{w}_t^T \Phi_{u,v_t} \right) \right\| \right) \]

\[ \leq M + CA + C' (2R)^p \| \mathbf{w}_t \|^{p-1} \]

\[ \leq M + CA + C' (2R)^p M^{p-1} = G \]

Proof of Theorem 5

We have

\[ \| \mathbf{w}_{t+1} - \mathbf{w}^* \|^2 = \| \mathbf{w}_t - \eta_t g_t - \mathbf{w}^* \|^2 \]

\[ = \| \mathbf{w}_t - \mathbf{w}^* \|^2 + \eta_t^2 \| g_t \|^2 - 2 \eta_t \mathbf{w}^*^T (\mathbf{w}_t - \mathbf{w}^*) \]

Taking the conditional expectation w.r.t \( \mathbf{w}_t \), we gain

\[ \mathbb{E} \left[ \| \mathbf{w}_{t+1} - \mathbf{w}^* \|^2 \right] = \mathbb{E} \left[ \| \mathbf{w}_t - \mathbf{w}^* \|^2 \right] + \eta_t^2 \mathbb{E} \left[ \| g_t \|^2 \right] \]

\[ - 2 \eta_t \mathbb{E} \left[ g_t^T (\mathbf{w}_t - \mathbf{w}^*) \right] \]

\[ = \mathbb{E} \left[ \| \mathbf{w}_t - \mathbf{w}^* \|^2 \right] + \eta_t^2 \mathbb{E} \left[ \| g_t \|^2 \right] - 2 \eta_t \mathbb{E} \left[ (\mathbf{w}_t - \mathbf{w}^*)^T \mathbf{J}^* (\mathbf{w}_t) \right] \]

\[ \leq \mathbb{E} \left[ \| \mathbf{w}_t - \mathbf{w}^* \|^2 \right] + \eta_t^2 \mathbb{E} \left[ \| g_t \|^2 \right] + 2 \eta_t \mathbb{E} \left[ (\mathbf{w}_t - \mathbf{w}^*)^T \mathbf{J}^* (\mathbf{w}_t) \right] \]

\[ + 2 \eta_t \left( \mathbf{J}^* (\mathbf{w}^*) - \mathbf{J}^* (\mathbf{w}_t) - \frac{1}{2} \| \mathbf{w}_t - \mathbf{w}^* \|^2 \right) \]
Taking the expectation the above equation again, we yield
\[
\mathbb{E} [\|w_{t+1} - w^*\|^2] \leq \mathbb{E} [\|w_t - w^*\|^2] + \eta_t^2 \mathbb{E} [\|g_t\|^2] + 2 \eta_t \left( J(w^*) - \mathbb{E} [J(w_t)] - \frac{1}{2} \mathbb{E} [\|w_t - w^*\|^2] \right) \leq (1 - \eta_t) \mathbb{E} [\|w_t - w^*\|^2] + G \eta_t^2 + 2 \eta_t (J(w^*) - \mathbb{E} [J(w_t)])
\]
\[
\mathbb{E} [J(w_t)] - J(w^*) \leq \left( \frac{1}{2} \eta_t - \frac{1}{2} \right) \mathbb{E} [\|w_t - w^*\|^2] - \frac{1}{2} \mathbb{E} [\|w_t - w^*\|^2] + \frac{G \eta_t}{2}
\]

Using the learning rate \( \eta_t = \frac{2}{t+T} \), we gain
\[
\mathbb{E} [J(w_t)] - J(w^*) \leq \frac{t - 1}{4} \mathbb{E} [\|w_t - w^*\|^2] - \frac{t + 1}{4} \mathbb{E} [\|w_{t+1} - w^*\|^2] + \frac{G}{t+1}
\]
\[
\mathbb{E} [tJ(w_t)] - tJ(w^*) \leq \frac{(t - 1) t}{4} \mathbb{E} [\|w_t - w^*\|^2] - \frac{t(t + 1)}{4} \mathbb{E} [\|w_{t+1} - w^*\|^2] + \frac{G t}{t+1}
\]

Taking sum when \( t \) runs from 1 to \( T \), we achieve
\[
\mathbb{E} \left[ \sum_{t=1}^{T} tJ(w_t) \right] - \frac{T(T+1)}{2} J(w^*) \leq \frac{2G}{T} \]
\[
\mathbb{E} \left[ \frac{2}{T(T+1)} \sum_{t=1}^{T} tJ(w_t) \right] - J(w^*) \leq \frac{2G}{T} \]
\[
\mathbb{E} \left[ J \left( \frac{2}{T(T+1)} \sum_{t=1}^{T} tw_t \right) - J(w^*) \right] \leq \frac{2G}{T} \text{ (10)}
\]

thanks to the convexity of \( J(.) \)
\[
\mathbb{E} [J(\bar{w}_{T+1})] - J(w^*) \leq \frac{2G}{T}
\]

where \( \bar{w}_{T+1} = \frac{1}{T(T+1)} \sum_{t=1}^{T} tw_t \).

Furthermore, from the strong convexity of \( J(.) \) and \( w^* \) is a minimizer, we have
\[
J(\bar{w}_{T+1}) - J(w^*) \geq \left< J'(w^*), \bar{w}_{T+1} - w^* \right> + \frac{1}{2} \| \bar{w}_{T+1} - w^* \|^2 \geq \frac{1}{2} \| \bar{w}_{T+1} - w^* \|^2
\]
\[
\mathbb{E} [\| \bar{w}_{T+1} - w^* \|^2] \leq \frac{4G}{T} \text{ (11)}
\]

Proof of Theorem 6

Let us denote the random variable \( Z_{T+1} = J(\bar{w}_{T+1}) - J(w^*) \geq 0 \). According to Markov inequality, we have
\[
P [Z_{T+1} \geq \varepsilon] \leq \frac{\mathbb{E} [Z_{T+1}]}{\varepsilon} = \frac{\mathbb{E} [J(\bar{w}_{T+1}) - J(w^*)]}{\varepsilon} \leq \frac{2G}{T\varepsilon}
\]
\[
P [Z_{T+1} < \varepsilon] \geq 1 - \frac{2G}{T\varepsilon}
\]

By choosing \( T_0 = \lceil \frac{2L}{\varepsilon^2} \rceil \), for all \( T \geq T_0 \), we have
\[
P [Z_{T+1} < \varepsilon] \geq 1 - \frac{2G}{T\varepsilon} \geq 1 - \delta
\]