Sequential Alternating Proximal Method for Scalable Sparse Structural SVMs

P. Balamurugan
Computer Science and Automation
Indian Institute of Science
Bangalore, India.
Email: balamurugan@csa.iisc.ernet.in

Shirish Shevade
Computer Science and Automation
Indian Institute of Science
Bangalore, India.
Email: shirish@csa.iisc.ernet.in

T. Ravindra Babu
E-Com Research Lab
Education and Research
Infosys Ltd., India.
Email: Ravindrababu_T@infosys.com

***Abstract*—Structural Support Vector Machines (SSVMs) have recently gained wide prominence in classifying structured and complex objects like parse-trees, image segments and Part-of-Speech (POS) tags. Typical learning algorithms used in training SSVMs result in model parameters which are vectors residing in a large-dimensional feature space. Such a high-dimensional model parameter vector contains many non-zero components which often lead to slow prediction and storage issues. Hence there is a need for sparse parameter vectors which contain a very small number of non-zero components. L1-regularizer and elastic net regularizer have been traditionally used to get sparse model parameters. Though L1-regularized structural SVMs have been studied in the past, the use of elastic net regularizer for structural SVMs has not been explored yet. In this work, we formulate the elastic net SSVM and propose a sequential alternating proximal algorithm to solve the dual formulation. We compare the proposed method with existing methods for L1-regularized Structural SVMs. Experiments on large-scale benchmark datasets show that the proposed dual elastic net SSVM trained using the sequential alternating proximal algorithm scales well and results in highly sparse model parameters while achieving a comparable generalization performance. Hence the proposed sequential alternating proximal algorithm is a competitive method to achieve sparse model parameters and a comparable generalization performance when elastic net regularized Structural SVMs are used on very large datasets.**

***Keywords*-Structural SVMs, Alternating Proximal method.**

I. Introduction

Structured classification is the task of classifying structured output objects like parse-trees, image segments, protein structures from corresponding inputs like sentences, images and amino acid sequences. Such structured output objects are composed of various components which, in general, are not independent of each other. Each component of the object interacts with one or various other components of the same object in a complex manner. This nature of interaction among the various components of the structured outputs distinguishes these objects from distinct outputs which are used in binary and multi-class classification.

Structured classification in a supervised setting comprises of the learning stage where a suitable parametric model is learnt using a learning algorithm and the prediction stage where the prediction of outputs is done using the learnt parameters. In this work, we consider a structured input-output space pair $(\mathscr{X}, \mathscr{Y})$ where the structured objects $(\mathbf{x}, \mathbf{y})$ reside. For sequence learning application, which is a well-known example of structured classification, each object $\mathbf{x} \in \mathscr{X}$ is assumed to be made of T parts $\mathbf{x} = (x^1, x^2, \cdots, x^T)$ and the associated structured output $\mathbf{y} \in \mathscr{Y}$ has corresponding parts $\mathbf{y} = (y^1, y^2, \cdots, y^T)$. A crucial component of structured classification is the feature vector $f(\mathbf{x}, \mathbf{y})$ which relates an input $\mathbf{x}$ with a structured output $\mathbf{y}$. The feature vector is a map $f : \mathscr{X} \times \mathscr{Y} \rightarrow \mathbb{R}^d$ which transforms an input-output pair $(\mathbf{x}, \mathbf{y})$ into a d-dimensional vector. In practical applications, d is of the order of millions and designing a suitable $f(\mathbf{x}, \mathbf{y})$ for the particular application in hand plays an important role during the learning stage [16].

Structured output learning involves learning a discriminant function $h : \mathscr{X} \times \mathscr{Y} \rightarrow \mathbb{R}$ which is often parametrized with a suitable d-dimensional vector $\mathbf{w}$ that resides in the same space as the feature vector. The discriminant function h generally takes a linear form given by

$$h(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \mathbf{w}^T f(\mathbf{x}, \mathbf{y}) \tag{I.1}$$

The prediction stage then uses this discriminant rule to classify an unseen object $\hat{\mathbf{x}}$ as

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \; h(\hat{\mathbf{x}}, \mathbf{y}; \mathbf{w}) \tag{I.2}$$

We note that the prediction by (I.2) involves a search among all possible outputs $\mathbf{y} \in \mathscr{Y}$.

Support Vector Machines have been widely used for various machine-learning tasks [17]. Structural Support Vector Machines (SSVMs) [15][16] have become a very popular technique in structured classification. With the advent of very fast and efficient learning algorithms for structured output learning like the sequential dual method [2] and the cutting-plane method [9], SSVMs are a practical choice for many applications like POS-tagging, dependency parsing, etc., SSVMs solve a convex Quadratic Program (QP) which optimizes an L2-regularized parameter vector and consists of an exponential number of constraints corresponding to the number of possible structured outputs $\mathbf{y} \in \mathscr{Y}$. The solution of such a QP results in a d-dimensional model parameter vector $\mathbf{w}$. The high-dimensional parameter vector $\mathbf{w}$ usually consists of non-zero values for many or all dimensions. This

is primarily due to the presence of the L2-regularizer term in the QP. This often leads to slow prediction and storage concerns. Hence, there is a growing need for finding sparse parameter vectors which contain only a very small number of non-zero components. This would not only help in a compact parameter vector which helps allay storage issues and helps in faster prediction but would also lead to feature-selection for structured classification in a straight-forward manner.

One possible approach to obtaining a sparse model parameter vector $\mathbf{w}$ is the use of L1-regularizer instead of the L2-regularizer used in standard Structural SVM formulation. L1-regularizer in structural SVMs results in a Linear Program (LP) instead of a QP; but it still contains an exponential number of constraints. A variety of methods to solve the resultant LP have been given in [22] and [20]. However, the experimental results have been reported only on small or synthetic datasets. Hence, the scalability of L1-regularized SSVMs to large datasets with millions of features and the effect of L1-regularizer on model sparsity for such datasets remain to be explored.

A different approach to obtain sparse model parameters is to use a weighted combination of L1 and L2 regularizers which is called the elastic net regularizer. Elastic net regularizer was introduced in [23] for regression problems and in [18] for binary classification problems. In both regression and classification, elastic net regularizer has been shown to obtain sparse model parameters. In structured classification setting, elastic net regularizer has been used for Conditional Random Fields (CRFs) in [10]. However the use of elastic net regularizer has not yet been explored for Structural SVMs. Moreover, the usage of elastic net regularizer has been studied frequently in its primal formulation. The dual formulations of elastic net regularizer have rarely been used; see [8] and [21] where they have been used in the context of multiple-kernel learning and in the analysis of linearized Bregman methods. However, the dual elastic net formulations used in [8] and [21] consist of only a small number of variables. Hence there is a need for developing efficient algorithms for solving the dual elastic net SSVM formulation which involves an exponential number of variables.

A. Contributions

In this work, we formulate the primal problem of elastic net regularized structural SVMs and derive its dual. The use of alternating proximal algorithm is explored when the dual problem has exponentially large number of variables. We adapt the basic alternating proximal scheme to solve the dual elastic net structural SVM. We devise a sequential alternating proximal method which works by sequentially visiting each training example and solving simpler problems restricted to a small subset of variables associated with that example. We implemented and experimented our method on various benchmark large-scale sequence learning datasets for which the existing L1 regularized methods do not scale well. Our results demonstrate that the proposed sequential alternating proximal method scales very well and the use of elastic net regularizer results in highly sparse model parameters. The proposed method also achieves a comparable generalization performance with the resultant sparse model. We also conducted experiments to study contributions of L1 and L2 regularizers on model sparsity. Through these various experiments, the proposed sequential alternating proximal method to solve the dual elastic net structural SVM formulation is shown to be a competitive scalable method to achieve very sparse models.

The rest of the paper is organized as follows. The next section describes various formulations of Structural SVMs and discusses relevant algorithms to solve these formulations. The use of elastic net regularizer in various contexts is also discussed. In Section III, we introduce the elastic net regularized Structural SVMs and derive its dual. We illustrate the sequential alternating proximal method to solve the dual elastic net SSVM formulation in Section IV. The details of empirical results on scalability, sparsity and generalization performance, achieved by the proposed method on various datasets, are presented in Section V. Section VI concludes the paper.

II. Structural SVMs

Structural SVMs [15][16] learn from training data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ of structured inputs and outputs, by finding the solution to a convex Quadratic Program (QP), which is (**OP1**):

$$\begin{aligned} \min_{\mathbf{w}, \xi_i \geq 0} \quad & \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \xi_i \geq l_i(\mathbf{y}) - \mathbf{w}^T \Delta f_i(\mathbf{y}) \; \forall \; i, \; \forall \; \mathbf{y} \in \mathscr{Y} \end{aligned} \tag{II.1}$$

where $C > 0$ is a regularization coefficient that trades regularization of $\mathbf{w}$ against the sum of slack variables ξ_i, which quantify the loss encountered in misclassification. The vector $\Delta f_i(\mathbf{y}) = f(\mathbf{x}_i, \mathbf{y}_i) - f(\mathbf{x}_i, \mathbf{y})$ can be called the difference feature vector. The term $l_i(\mathbf{y})$ is a loss function which is decomposable over various parts of the structured output. For sequence-labeling applications, $l_i(\mathbf{y})$ is the decomposable Hamming-loss function

$$l_i(\mathbf{y}) = \sum_{t=1}^{T} I(y_i^t \neq y^t) \tag{II.2}$$

where $I(p) = 1$ if p is true and 0 otherwise. We note that OP1 contains a quadratic regularizer involving the model parameters $\mathbf{w}$. The quadratic regularizer does not necessarily lead to a sparse model, *i.e.*, most of the components of the $\mathbf{w}$ vector are typically non-zero. This makes the inference task in (I.2) to be slow, particularly when the dimension d of $\mathbf{w}$ is very large.

An equivalent single-slack formulation of problem OP1 given in [9] is (**OP1-1-slack**):

$$\min_{\mathbf{w},\xi\geq 0} \quad \frac{1}{2}\|\mathbf{w}\|_2^2 + C\xi$$
$$\text{s.t.} \quad \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} \Delta f_i(\bar{\mathbf{y}}_i) \geq \frac{1}{n}\sum_{i=1}^{n} l_i(\bar{\mathbf{y}}_i) - \xi,$$
$$\forall\, (\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_n) \in \mathscr{Y}^n \qquad \text{(II.3)}$$

A cutting-plane method to solve OP1-1-slack proposed in [9], has been observed to be fast on various large datasets. A sequential dual algorithm proposed in [2] directly works on OP1 and solves a scaled version of dual problem of OP1, which is (**OP2**):

$$\min_{\boldsymbol{\alpha}} \quad \frac{C}{2}\| \sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y})\Delta f_i(\mathbf{y})\|^2 - \sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) l_i(\mathbf{y})$$
$$\text{s.t.} \quad \sum_{\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) = 1 \;\forall\, i, \quad \boldsymbol{\alpha}_i(\mathbf{y}) \geq 0 \;\forall\, i, \mathbf{y} \qquad \text{(II.4)}$$

The sequential dual algorithm sequentially traverses through each example i and solves a subproblem involving the dual variables $\boldsymbol{\alpha}_i(\mathbf{y})$ associated with the i-th example. It has been empirically observed to be very fast on large datasets. The primal and dual variables of OP1 and OP2 are related as

$$\mathbf{w} = C\sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y})\Delta f_i(\mathbf{y}) \qquad \text{(II.5)}$$

Solving OP1-1-slack or OP2 usually results in non-sparse parameter vector $\mathbf{w}$.

The use of L1 regularizer is well-known to obtain sparse model parameters in SVM-based methods [4]. Zhu et al. [22] studied the use of L1-regularizer for structural SVMs. They replace the quadratic term $\frac{1}{2}\|\mathbf{w}\|_2^2$ in OP1 with $\|\mathbf{w}\|_1$, which can be formulated as an LP (**OP3**):

$$\min_{\mathbf{w},\xi_i\geq 0} \quad \|\mathbf{w}\|_1 + C\sum_i \xi_i$$
$$\text{s.t.} \quad \xi_i \geq l_i(\mathbf{y}) - \mathbf{w}^T\Delta f_i(\mathbf{y}) \;\forall\, i, \;\forall\, \mathbf{y} \in \mathscr{Y} \qquad \text{(II.6)}$$

The L1-norm regularizer is empirically found to give sparse model parameters. A simple cutting-plane method is proposed in [22] to solve OP3. However, solving the resultant sub-LP requires an off-the-shelf LP solver. Experimental results reported indicate that off-the-shelf LP solvers are much slower and do not scale well even for medium-sized datasets. To overcome the dependence on such solvers, other methods like projected sub-gradient and EM-style algorithm are given in [22]. The projected sub-gradient method solves an equivalent formulation of OP3, given as :

$$\min_{\mathbf{w}} \quad \frac{1}{n}\sum_{i=1}^{n} \max_{\mathbf{y}\in\mathscr{Y}} \left\{ l_i(\mathbf{y}) - \mathbf{w}^T\Delta f_i(\mathbf{y})\right\}$$
$$\text{s.t.} \quad \|\mathbf{w}\|_1 \leq \epsilon \qquad \text{(II.7)}$$

We note that the formulation (II.7) however contains a norm-constraint $\|\mathbf{w}\|_1 \leq \epsilon$ for some positive ϵ. The projected sub-gradient method sequentially visits each example i and updates $\mathbf{w}$ by finding a sub-gradient of the objective term in (II.7) at the i-th example. The update step used in the projected sub-gradient method is similar in spirit to stochastic gradient descent update [3]. After this update step at the i-th example, a projection step is carried out which projects the updated parameters to the L1-norm ball $\|\mathbf{w}\|_1 \leq \epsilon$. The positive parameter ϵ gives a control on the sparsity of the model parameters. Note that the projection step after each example, as given in the online version of the projected sub-gradient algorithm in [22], is computationally expensive [7]. Another algorithm proposed in [22] is the EM-style algorithm which solves an equivalent adaptive-net formulation of OP3. However, the results reported in [22] are for small synthetic datasets and the scalability of the proposed methods to very large datasets is yet to be explored.

Wang and Shawe-Taylor [20] impose a positivity constraint $\mathbf{w} \geq \mathbf{0}$ in the problem OP3. With this additional positivity constraint, the Lagrange function for the problem OP3 can be given as (**OP4**):

$$\min_{\mathbf{w}\geq 0,\xi_i\geq 0} \max_{\boldsymbol{\alpha}_i(\mathbf{y})\geq 0} \quad \mathbf{1}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i +$$
$$\sum_{i=1}^{n}\sum_{\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y})\left[l_i(\mathbf{y}) - \mathbf{w}^T\Delta f_i(\mathbf{y}) - \xi_i\right] \qquad \text{(II.8)}$$

In [20], Wang and Shawe-Taylor considered the simple 0/1-loss: $l_i(\mathbf{y}) = 0$, if $\mathbf{y} = \mathbf{y}_i$ and 1 otherwise, and designed a column-generation method to add constraints to a working set $\mathscr{W}$. The extra-gradient method was used to find a saddle point of the Lagrange function OP4 with respect to the constraints in $\mathscr{W}$. The method gave a comparable generalization performance within first 20 iterations on medium sized datasets. However, the results mentioned in [20] do not give details about the sparsity of the model obtained. The scalability of the proposed extra-gradient method to large datasets with very large feature dimensions has also not yet been explored.

Apart from the L1 regularizer, the use of weighted combination of L1 and L2 regularizers $\|\mathbf{w}\|_1$ and $\|\mathbf{w}\|_2^2$, also results in sparse models. Such a combination is called the elastic net regularizer [23]. Elastic net regularizer has been used in the context of binary classification [18] and regression [23] in its primal form. Recently, Lavergne et al. [10] applied elastic net regularizer to structured output learning using CRFs and gave a number of algorithms to solve the primal problem. Dual elastic net regularizer for multiple-kernel learning has been explored in [8]. However, the use of elastic net regularizer for Structural SVMs has not yet been studied to the best of our knowledge.

In this work, we formulate the dual elastic net regularized Structural SVM and propose an efficient algorithm to solve

the dual problem.

III. Elastic Net Regularized Structural SVMs

We introduce the primal elastic net regularized SSVM problem (ENSSVM) as the following optimization problem (**OP5**):

$$\begin{aligned} \min_{\mathbf{w},\xi_i \geq 0} \quad & \rho_1 \|\mathbf{w}\|_1 + \frac{\rho_2}{2}\|\mathbf{w}\|_2^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \xi_i \geq l_i(\mathbf{y}) - \mathbf{w}^T \Delta f_i(\mathbf{y}) \; \forall \; i, \; \forall \; \mathbf{y} \in \mathscr{Y} \end{aligned} \tag{III.1}$$

We note that the primal problem consists of the combination of both the L1-regularizer term $\|\mathbf{w}\|_1$ weighted by a factor ρ_1 and the squared L2-regularizer term $\|\mathbf{w}\|_2^2$ weighted by ρ_2. By putting appropriate non-negative weights ρ_1 and ρ_2, it is possible to recover both OP1 and OP3 from OP5. Hence OP5 can be considered as a generalization of OP1 and OP3.

The dual problem of OP5 can be formulated as a convex QP (**OP6**):

$$\begin{aligned} \min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & \frac{1}{2\rho_2}\| \sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) \Delta f_i(\mathbf{y}) - \boldsymbol{\beta}\|^2 - \sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) l_i(\mathbf{y}) \\ & \text{s.t.} \; \sum_{\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) = C \; \forall \; i, \;\; \boldsymbol{\alpha}_i(\mathbf{y}) \geq 0 \; \forall \; i, \mathbf{y}, \\ & \qquad\qquad \boldsymbol{\beta} \in [-\rho_1, \rho_1]^d \end{aligned}$$

We give the derivation of the dual problem below.

A. Derivation of Dual Problem OP6

We derive the dual problem as follows. To handle the non-differentiable term $\|\mathbf{w}\|_1$, we write an equivalent formulation of the primal problem OP5 as (**P5**):

$$\begin{aligned} \min_{\mathbf{w},\mathbf{u},\mathbf{v},\xi_i \geq 0} \quad & \rho_1 \mathbf{1}^T(\mathbf{u}+\mathbf{v}) + \frac{\rho_2}{2}\|\mathbf{w}\|_2^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \mathbf{w} = \mathbf{u} - \mathbf{v} \; , \; \mathbf{u} \geq 0 \; , \; \mathbf{v} \geq 0, \\ & \xi_i \geq l_i(\mathbf{y}) - \mathbf{w}^T \Delta f_i(\mathbf{y}) \; \forall \; i, \; \forall \; \mathbf{y} \in \mathscr{Y} \end{aligned}$$

The problem P5 is a convex QP with linear constraints. Therefore, the first-order KKT conditions are necessary and sufficient at optimality. From the KKT conditions of P5, we obtain

$$\mathbf{w} \;\; = \;\; \frac{1}{\rho_2}\left[\sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) \Delta f_i(\mathbf{y}) - \boldsymbol{\beta}\right], \rho_2 \neq 0, \tag{III.2}$$

$$\boldsymbol{\beta} \in [-\rho_1, \rho_1]^d \tag{III.3}$$

and

$$\sum_{\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) = C \;\; \forall \; i. \tag{III.4}$$

Using these conditions, the dual problem of P5 can be written as in OP6. Note that if $\rho_2 = 0$, the resulting dual is a large-scale LP. By scaling $\boldsymbol{\alpha}_i(\mathbf{y}) \approx C\boldsymbol{\alpha}_i(\mathbf{y})$ in OP6, we obtain a modified normalization constraint which restricts the dual variables $\boldsymbol{\alpha}_i(\mathbf{y})$ for each example i to belong to a unit simplex. Thus we have a scaled version of OP6 given by (**OP7**):

$$\begin{aligned} \min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & \frac{C}{2\rho_2}\left\| \sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) \Delta f_i(\mathbf{y}) - \frac{\boldsymbol{\beta}}{C}\right\|^2 - \sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) l_i(\mathbf{y}) \\ & \text{s.t.} \; \sum_{\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) = 1 \; \forall \; i, \;\; \boldsymbol{\alpha}_i(\mathbf{y}) \geq 0 \; \forall \; i, \mathbf{y}, \\ & \qquad\qquad \boldsymbol{\beta} \in [-\rho_1, \rho_1]^d \end{aligned} \tag{III.5}$$

This problem formulation is also referred to as the Dual ENSSVM in this paper. The optimal dual objective value of the scaled version OP7 is (-$\frac{1}{C}$) times the optimal primal objective value obtained by OP5. The primal and dual variables of the problems OP5 and OP7 are related as

$$\mathbf{w} = \frac{1}{\rho_2}\left[C\sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) \Delta f_i(\mathbf{y}) - \boldsymbol{\beta}\right]. \tag{III.6}$$

$\boldsymbol{\beta}$ is a d-dimensional vector, with each of its components taking values in $[-\rho_1, \rho_1]$. The $\boldsymbol{\beta}$ vector aids in achieving sparsity of the primal parameter vector $\mathbf{w}$ as described in the next section.

We note the close resemblance of dual problem OP7 with the problem OP2, which is the dual of Structural SVM. Indeed, for a fixed $\boldsymbol{\beta}$, the optimality of $\boldsymbol{\alpha}_i(\mathbf{y})$ in OP7 can be checked by the quantity

$$\zeta_i = \max_{\mathbf{y}} \; g_i(\mathbf{y}) - \min_{\mathbf{y}:\boldsymbol{\alpha}_i(\mathbf{y})>0} g_i(\mathbf{y}) \tag{III.7}$$

where

$$g_i(\mathbf{y}) = l_i(\mathbf{y}) - \mathbf{w}^T \Delta f_i(\mathbf{y}). \tag{III.8}$$

Hence, at optimality we would have

$$\zeta_i = 0 \;\; \forall \; i. \tag{III.9}$$

Note that the optimality conditions given by (III.9) are very similar to the optimality conditions for OP2 [2]. Hence applying a sequential optimization method, similar to that in [2], to solve OP7 can be tried. However a straight-forward application of such a scheme might not help because of the presence of another variable $\boldsymbol{\beta}$. To overcome this difficulty, we propose a sequential alternating proximal algorithm to solve the Dual ENSSVM. As we see later, the optimality condition (III.9) can be used as a possible termination criterion for the proposed sequential alternating proximal algorithm.

IV. A Sequential Alternating Proximal Method to solve Dual ENSSVM

In this section, we describe a sequential alternating proximal method to solve Dual ENSSVM. We first illustrate the basic alternating proximal method given in [1]. We consider the convex problem of the following type:

$$\min_{u\in\mathscr{U},v\in\mathscr{V}} \; H(u)+G(v)+\frac{\sigma}{2}P(u,v) \tag{IV.1}$$

where we make the following assumptions ($\mathscr{A}_1$):

- $\mathscr{U}$ and $\mathscr{V}$ are real (possibly infinite dimensional) Hilbert spaces;
- $H:\mathscr{U}\to\mathbb{R}\cup\{+\infty\}, G:\mathscr{V}\to\mathbb{R}\cup\{+\infty\}$ are closed convex proper functions on $\mathscr{U}$ and $\mathscr{V}$ respectively;
- $P(u,v):\mathscr{U}\times\mathscr{V}\to\mathbb{R}^+$ is a continuous quadratic function of the form $P(u,v)=\|Au-Bv\|^2$ and A,B are linear continuous operators;
- $\sigma>0$.

The alternating proximal algorithm (with costs-to-move) given in Attouch et al. [1] solves (IV.1) by starting from an initial point (u^0,v^0) and generating iterates $\{u^k,v^k\}_{k=0}^{\infty}$ by the following alternating procedure. First, u^{k+1} is found by solving

$$u^{k+1}=\arg\min_{\eta}\left\{H(\eta)+\frac{\sigma}{2}P(\eta,v^k)+\frac{\mu}{2}\|\eta-u^k\|^2\right\} \tag{IV.2}$$

Then the algorithm alternates to find v^{k+1} by solving

$$v^{k+1}=\arg\min_{\gamma}\left\{G(\gamma)+\frac{\sigma}{2}P(u^{k+1},\gamma)+\frac{\nu}{2}\|\gamma-v^k\|^2\right\} \tag{IV.3}$$

The cost-to-move terms $\|\eta-u^k\|^2$ and $\|\gamma-v^k\|^2$ prevent huge oscillation of the iterates from their previous values. The penalty terms μ and ν are assumed to be positive. Then the alternating procedure of solving (IV.2) and (IV.3) is guaranteed to weakly converge to an optimal solution [1].

We now show that the dual problem OP7 can be written in a form as given in (IV.1) and hence the alternating proximal method can be directly applied to solve the dual problem OP7. We first introduce the indicator-function θ_S of a non-empty closed convex set S as

$$\theta_S(z)=\begin{cases}0 & \text{if } z\in S\\ +\infty & \text{otherwise}\end{cases} \tag{IV.4}$$

If the outputs can be enumerated as $\mathbf{y}^1,\mathbf{y}^2,\cdots,\mathbf{y}^m$ such that m denotes the size $|\mathscr{Y}|$ of the output space $\mathscr{Y}$, then we can write the vector of dual variables as $\boldsymbol{\alpha}=[\alpha_i(\mathbf{y}^j)]_{i=1,j=1}^{n,m}$. The variables associated with a single example can then be denoted as $\boldsymbol{\alpha}_i=[\alpha_i(\mathbf{y}^j)]_{j=1}^{m}$. We can similarly define $\mathbf{g}_i=[g_i(\mathbf{y}^j)]_{j=1}^{m}$ for an example i where $g_i(\mathbf{y}^j)$ is given by (III.8). Now, we consider the following non-empty convex sets

$$S_\alpha=\left\{\boldsymbol{\alpha}:\sum_{j=1}^{m}\boldsymbol{\alpha}_i(\mathbf{y}^j)=1\;\forall\, i,\;\; \boldsymbol{\alpha}_i(\mathbf{y}^j)\geq 0\;\forall\, i,j\right\} \tag{IV.5}$$

and

$$S_\beta=\left\{\boldsymbol{\beta}:\boldsymbol{\beta}\in[-\rho_1,\rho_1]^d\right\}. \tag{IV.6}$$

We write the linear term $\sum_{i,\mathbf{y}}\boldsymbol{\alpha}_i(\mathbf{y})l_i(\mathbf{y})$ as $\boldsymbol{\alpha}^T l$ and the quadratic term $\frac{C}{2\rho_2}\left\|\sum_{i,\mathbf{y}}\boldsymbol{\alpha}_i(\mathbf{y})\Delta f_i(\mathbf{y})-\frac{\boldsymbol{\beta}}{C}\right\|^2$ as $\frac{\sigma}{2}P(\boldsymbol{\alpha},\boldsymbol{\beta})$, where $\sigma=\frac{C}{\rho_2}$. Then, we can write the problem OP7 as (**OP8**):

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \; H(\boldsymbol{\alpha})+G(\boldsymbol{\beta})+\frac{\sigma}{2}P(\boldsymbol{\alpha},\boldsymbol{\beta}) \tag{IV.7}$$

where

$$H(\boldsymbol{\alpha}) \;=\; \theta_{S_\alpha}(\boldsymbol{\alpha})-\boldsymbol{\alpha}^T l \tag{IV.8}$$

and

$$G(\boldsymbol{\beta}) \;=\; \theta_{S_\beta}(\boldsymbol{\beta}). \tag{IV.9}$$

The alternating proximal algorithm (with costs to move) proposed by Attouch et al. [1] to solve OP8 is as follows. We start with an initial point $(\boldsymbol{\alpha}^0,\boldsymbol{\beta}^0)$ and optimize OP8 by alternating between the variables $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. At the $(k+1)$-th iteration, we obtain $(\boldsymbol{\alpha}^{k+1},\boldsymbol{\beta}^{k+1})$ as

$$\boldsymbol{\alpha}^{k+1}=\arg\min_{\eta}\left\{H(\eta)+\frac{\sigma}{2}P(\eta,\boldsymbol{\beta}^k)+\frac{\mu}{2}\|\eta-\boldsymbol{\alpha}^k\|^2\right\} \tag{IV.10}$$

and

$$\boldsymbol{\beta}^{k+1}=\arg\min_{\gamma}\left\{G(\gamma)+\frac{\sigma}{2}P(\boldsymbol{\alpha}^{k+1},\gamma)+\frac{\nu}{2}\|\gamma-\boldsymbol{\beta}^k\|^2\right\} \tag{IV.11}$$

We denote (IV.10) as the $\boldsymbol{\alpha}$-problem and (IV.11) as the $\boldsymbol{\beta}$-problem. Hence the alternating proximal algorithm alternates between solving the $\boldsymbol{\alpha}$-problem and the $\boldsymbol{\beta}$-problem and imposes a cost μ on moving the iterate $\boldsymbol{\alpha}^k$ to $\boldsymbol{\alpha}^{k+1}$ and ν on moving the iterate $\boldsymbol{\beta}^k$ to $\boldsymbol{\beta}^{k+1}$. In [1], these cost-to-move terms $\frac{\mu}{2}\|\boldsymbol{\alpha}^{k+1}-\boldsymbol{\alpha}^k\|^2$ and $\frac{\nu}{2}\|\boldsymbol{\beta}^{k+1}-\boldsymbol{\beta}^k\|^2$ play a crucial role in the convergence. We describe the overall alternating proximal algorithm to solve OP8 in Algorithm 1. Note that the initialization of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ variables is done in Step 3 and Step 4 of Algorithm 1 so as to satisfy the feasibility conditions in OP7. Since the $\boldsymbol{\alpha}_i(\mathbf{y})$ variables can be considered to form a distribution for each example i, their initialization is done in such a way that, for an example $(\mathbf{x}_i,\mathbf{y}_i)$, the entire mass is initially associated with the actual output $\mathbf{y}_i$.

Algorithm 1 *An Alternating Proximal algorithm to solve OP8*

1: Input $S = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, C
2: $\mathbf{w} = \mathbf{0}$
3: Initialize $\boldsymbol{\alpha}^0$ as $\boldsymbol{\alpha}_i^0(\mathbf{y}_i) = 1 \ \forall i$, $\boldsymbol{\alpha}_i^0(\mathbf{y}) = 0 \ \forall i, \ \forall \mathbf{y} \in \mathscr{Y} \setminus \{\mathbf{y}_i\}$
4: $\boldsymbol{\beta}^0 = \mathbf{0}$
5: **for** $k = 0, 1, 2, \ldots$ **do**
6: Solve $\boldsymbol{\alpha}$-problem in (IV.10) to get $\boldsymbol{\alpha}^{k+1}$
7: Solve $\boldsymbol{\beta}$-problem in (IV.11) to get $\boldsymbol{\beta}^{k+1}$
8: Update $\mathbf{w}$ through (III.6)
9: **end for**

Under the assumptions of $\mathscr{A}_1$ and assuming that the objective function $H(u) + G(v) + \frac{\sigma}{2}P(u, v)$ in (IV.1) has at least one minimum point, and with $\mu, \nu > 0$, the alternating procedure described in (IV.2) and (IV.3) weakly converges to an optimum point (u^∞, v^∞) [1]. We note that the objective function in OP8 satisfies these assumptions and hence Algorithm 1 to solve OP8 weakly converges to an optimal solution $(\boldsymbol{\alpha}^\infty, \boldsymbol{\beta}^\infty)$ of OP8. We state the theorem on convergence of the algorithm without proof [1].

Theorem 1: The sequence $(\boldsymbol{\alpha}^k, \boldsymbol{\beta}^k)$ generated by the Alternating Proximal Algorithm (Algorithm 1) weakly converges to a minimum point $(\boldsymbol{\alpha}^\infty, \boldsymbol{\beta}^\infty)$ of OP8. Moreover, $\|\boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k\|^2 \to 0$ and $\|\boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\|^2 \to 0$, as $k \to \infty$.

Let us now describe how to solve the $\boldsymbol{\alpha}$-problem and the $\boldsymbol{\beta}$-problem. We use the values $\mu = 1$ and $\nu = 1$ for illustration purpose. Consider the case where all $\boldsymbol{\alpha}_i^k(\mathbf{y})$ variables are available to us. Then the $\boldsymbol{\alpha}$-problem can be solved using a plain SMO-type algorithm [13]. But ENSSVMs consist of an exponential number of $\boldsymbol{\alpha}_i^k(\mathbf{y})$ variables and hence we follow a sequential approach similar to that used in [2] to solve the $\boldsymbol{\alpha}$-problem. We describe the procedure next.

A. Solving the $\boldsymbol{\alpha}$-problem

Note that, for a fixed $\boldsymbol{\beta}$, the $\boldsymbol{\alpha}$-problem in (IV.10) is of the form

$$\min_{\boldsymbol{\alpha}} \ \frac{C}{2\rho_2}\left\|\sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y})\Delta f_i(\mathbf{y}) - \frac{\boldsymbol{\beta}^k}{C}\right\|^2 - \sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) l_i(\mathbf{y}) + \frac{1}{2}\|\boldsymbol{\alpha} - \boldsymbol{\alpha}^k\|^2$$
$$\text{s.t.} \sum_{\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y}) = 1 \ \forall \ i, \ \ \boldsymbol{\alpha}_i(\mathbf{y}) \geq 0 \ \forall \ i, \ \forall \mathbf{y} \tag{IV.12}$$

We now discuss two different ways of solving the $\boldsymbol{\alpha}$-problem. We can solve (IV.12) over all examples i=$1, 2, \cdots, n$, which we call the batch approach. Alternatively, we can sequentially traverse through the examples and solve for the variables $\boldsymbol{\alpha}_i(\mathbf{y})$ associated with a particular example i, by fixing all other variables $\boldsymbol{\alpha}_j(\mathbf{y}), \ j \neq i$. This is called the sequential approach. In the batch approach, the $\boldsymbol{\beta}$-update in step 7 and $\mathbf{w}$-update in step 8 of Algorithm 1 would be performed after processing all the examples, whereas in the sequential approach, these updates are carried out after every example. In our experiments, we observed that the sequential approach gives a useful model parameter $\mathbf{w}$ within a couple of passes over the examples while the batch approach fails to exhibit such a behaviour. We now describe the details of the sequential approach used to solve the $\boldsymbol{\alpha}$-problem. For a particular example i, we let $\boldsymbol{\alpha}_i^{k+1}(\mathbf{y}) = \boldsymbol{\alpha}_i^k(\mathbf{y}) + \boldsymbol{\delta\alpha_i}(\mathbf{y})$, where $\boldsymbol{\delta\alpha_i}(\mathbf{y})$ represents the change in $\boldsymbol{\alpha}_i^k(\mathbf{y})$ variables. We use the vector notation $\boldsymbol{\alpha}_i^{k+1} = \boldsymbol{\alpha}_i^k + \boldsymbol{\delta\alpha_i}$ to denote the same. Then, from (IV.12), we arrive at the following subproblem (**$\boldsymbol{\alpha}$-SUB**):

$$\min_{\boldsymbol{\delta\alpha_i}} \ \frac{C}{2\rho_2}\boldsymbol{\delta\alpha_i}^T(Q + D)\boldsymbol{\delta\alpha_i} - \mathbf{g}_i{}^T\boldsymbol{\delta\alpha_i}$$
$$\text{s.t.} \ \ \boldsymbol{\delta\alpha_i}^T\mathbf{1} = \mathbf{0}, \ \ \boldsymbol{\delta\alpha_i} \geq -\boldsymbol{\alpha}_i^k \tag{IV.13}$$

where D is a diagonal matrix with entries $\frac{\rho_2}{C}$ and the (p, q)-th entry in the matrix Q is given by $\Delta f_i(\mathbf{y}^p)^T\Delta f_i(\mathbf{y}^q)$. $\boldsymbol{\alpha}$-SUB is still over all possible $\mathbf{y} \in \mathscr{Y}$. However, the optimality conditions in (III.7) and (III.9) indicate that we can maintain a small subset V_i of outputs $\mathbf{y}$ for each example, such that $V_i = \{\mathbf{y} : \boldsymbol{\alpha}_i(\mathbf{y}) > 0\}$, which is very similar to that used in [2]. With the availability of such a set V_i, $\boldsymbol{\alpha}$-SUB can now be solved over a small set of variables $\boldsymbol{\alpha}_i^{k+1}(\mathbf{y}), \ \forall \mathbf{y} \in V_i$ using an SMO-type algorithm. After solving for the variables $\boldsymbol{\alpha}_i^{k+1}(\mathbf{y})$ associated with a particular example i, we alternate to solve the $\boldsymbol{\beta}$-problem given in (IV.11).

B. Solving the $\boldsymbol{\beta}$-problem

Given the current set of $\boldsymbol{\alpha}_i^{k+1}(\mathbf{y})$ variables, solving for the $\boldsymbol{\beta}$ variable is simpler. In the sequential approach, we update the $\boldsymbol{\beta}$ variable after solving $\boldsymbol{\alpha}$-problem for every example i. Therefore, we denote the update to $\boldsymbol{\beta}$ variable for i-th example in $(k+1)$-th iteration as $\boldsymbol{\beta}_i^{k+1}$. Hence by (IV.11), $\boldsymbol{\beta}_i^{k+1}$ is obtained using

$$\Pi_{[-\rho_1, \rho_1]^d}\left\{\frac{C}{C\rho_2 + 1}\left[\rho_2\boldsymbol{\beta}_{i-1}^{k+1} + \sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i^{k+1}(\mathbf{y})\Delta f_i(\mathbf{y})\right]\right\} \tag{IV.14}$$

where $\Pi_S\{z\}$ denotes the projection operator, projecting the value z to the set S. In particular, the projection in (IV.14) is done component-wise using $\Pi_{[-\rho_1, \rho_1]}\{z\} = \max(-\rho_1, \min(z, \rho_1))$. Note that such a simple projection step can be efficiently carried out after each example, as opposed to the expensive projection to a L1-ball, given in [22].

After solving the $\boldsymbol{\beta}$-problem, we update $\mathbf{w}$ through (III.6) and proceed to the next example $(i + 1)$ to solve the $\boldsymbol{\alpha}$-problem restricted to the variables $\boldsymbol{\alpha}_{i+1}^{k+1}(\mathbf{y})$. Thus, we

obtain a sequential alternating proximal algorithm which visits each training example sequentially and solves the $\boldsymbol{\alpha}$-problem in (IV.10) and alternates to solve the $\boldsymbol{\beta}$-problem in (IV.11) and then updates the primal variable $\mathbf{w}$ through (III.6). While updating $\mathbf{w}$ by (III.6), we observed that for appropriate values of μ and ν, many components of the term $C\sum_{i,\mathbf{y}} \boldsymbol{\alpha}_i(\mathbf{y})\Delta f_i(\mathbf{y})$ lie in the range $[-\rho_1, \rho_1]$ and the $\boldsymbol{\beta}$ value also equals the value of the former and hence forces many components of $\mathbf{w}$ to become zero. Thus the $\boldsymbol{\beta}$ variable implicitly helps in achieving sparse models.

The $\mathbf{w}$-update step is important for maintaining the set V_i for each example. To construct V_i, we use the following approach. Whenever we visit an example i, we find a violating output $\hat{\mathbf{y}}_i$ with the current primal variable $\mathbf{w}$ using

$$\hat{\mathbf{y}}_i \quad = \quad \arg\max_{\mathbf{y}} \; g_i(\mathbf{y}) \tag{IV.15}$$

where $g_i(\mathbf{y})$ is as defined in (III.8). If $\hat{\mathbf{y}}_i$ is not present in V_i, we add it to V_i. The set V_i can be considered as the active set of outputs $\mathbf{y}$, for which the corresponding $\boldsymbol{\alpha}_i(\mathbf{y})$ variables are non-zero. Finding $\hat{\mathbf{y}}_i$ also helps in checking the optimality of $\boldsymbol{\alpha}_i(\mathbf{y})$ variables. Note that ζ_i in (III.7) can be written as

$$\zeta_i \quad = \quad g_i(\hat{\mathbf{y}}_i) - \min_{\mathbf{y}\in V_i} \; g_i(\mathbf{y}). \tag{IV.16}$$

We can now check the optimality of $\boldsymbol{\alpha}_i(\mathbf{y})$ variables using (III.9). For numerical reasons, we check

$$\zeta_i < \vartheta \;\; \forall \; i \tag{IV.17}$$

for some small positive ϑ. We also like to point out that the $\arg\max$ computation in (IV.15) is a computationally intensive task and requires special purpose algorithms like Viterbi for sequence labeling applications. We describe the entire alternating procedure in Algorithm 2. Note that the algorithm is very easy to implement. Though Algorithm 2 does not contain any terminating condition in its present form, it is easy to see that the optimality condition (IV.17) can be used to terminate the algorithm. When (IV.17) holds for all the examples, no more optimization of $\boldsymbol{\alpha}_i(\mathbf{y})$ is carried out and hence the $\boldsymbol{\beta}$ variable also turns out to be optimal. We demonstrate the effectiveness of this algorithm on different real-world datasets in the next section.

Algorithm 2 *Sequential Alternating Proximal (SAP) algorithm to solve OP8*

1: Input $S = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, C
2: $\mathbf{w} = \mathbf{0}, V_i = \{\mathbf{y}_i\},\ i = 1, 2, \cdots, n$
3: Initialize $\boldsymbol{\alpha}^0$ as $\boldsymbol{\alpha}_i^0(\mathbf{y}_i) = 1\ \forall i\ ,\ \boldsymbol{\alpha}_i^0(\mathbf{y}) = 0\ \forall i,\ \forall \mathbf{y} \in \mathscr{Y} \setminus \{\mathbf{y}_i\}$
4: $\boldsymbol{\beta}^0 = \mathbf{0}$
5: **for** $k = 0, 1, 2, \ldots$ **do**
6: **for** $i = 1, 2, \ldots, n$ **do**
7: Find violator $\hat{\mathbf{y}}_i$ using (IV.15)
8: **if** $\hat{\mathbf{y}}_i \notin V_i$ **then**
9: $V_i = V_i \cup \{\hat{\mathbf{y}}_i\}$
10: $\boldsymbol{\alpha}_i^k(\hat{\mathbf{y}}_i) = 0$
11: **end if**
12: Solve $\boldsymbol{\alpha}$-SUB with respect to V_i to get $\boldsymbol{\alpha}_i^{k+1}(\mathbf{y})$
13: Obtain $\boldsymbol{\beta}^{k+1}$ using (IV.14)
14: Update $\mathbf{w}$ through (III.6)
15: **end for**
16: **end for**

V. Experiments

In this section, we give details about experiments conducted using the proposed sequential alternating proximal (SAP) algorithm for dual ENSSVM. We consider the problem of sequence labeling, which is a well-known structured prediction problem. We used five benchmark sequence labeling datasets for our experiments. They are OCR [12], Part-of-Speech Tagging (POS) [12], CoNLL2000 [14], WSJ-POS [11] and Brown [5]. The dataset characteristics are listed in Table I. OCR dataset is further divided into 10 different partitions, each containing similar proportion of train and test sizes.

The feature vector $f(\mathbf{x}, \mathbf{y})$ for these datasets is constructed using the procedure given in [16]. Finding the violator sequence in (IV.15) is done using Viterbi algorithm. All programs were implemented in C language. The experiments were run on a dual-CPU quad-core 2.4GHz Intel Xeon Processor with a 16GB shared main memory running under Linux.

Though ρ_1 and ρ_2 parameters can take any positive value, we assume $\rho_1 + \rho_2 = 1$ in our experiments. The range $\rho_1 \in (0, 1)$ helps us to understand the results in terms of the number of non-zero parameters selected and to compare the generalization performance of the model with the selected features. In our experiments, we set μ and ν to small positive values.

1) Comparison of the SAP algorithm for Dual ENSSVM with L1-SSVM: We implemented L1 regularized SSVM using the projected sub-gradient algorithm given in [22]. The L1-SSVM solved by the projected sub-gradient algorithm contains a bound constraint of the form $\|\mathbf{w}\|_1 \leq \epsilon$ for some positive ϵ. The ϵ value controls the sparsity of model parameters. Large values of ϵ give rise to many non-zero model parameters. We implemented the projection of the model parameters onto $\|\mathbf{w}\|_1 \leq \epsilon$ using the procedure given in [6]. Note that if the projection step is carried out after each example as given in [22], it slows down the training drastically and hence we perform the projection step only after passing through all examples, in contrast to the sequential $\boldsymbol{\beta}$-update done after every example in the SAP algorithm (Algorithm 2-Step 13).

We experimented using our dual SAP algorithm and the projected sub-gradient method for L1-SSVM on OCR data

Table I
DATA SET SUMMARY. n AND n_{test} DENOTE THE SIZES OF THE TRAINING AND TEST DATA RESPECTIVELY, r IS THE INPUT DIMENSION, l DENOTES THE NUMBER OF CLASSES AND d IS THE FEATURE VECTOR DIMENSION

Data set	n	n_{test}	r	l	d
OCR	6877	55310	128	26	4004
POS	7200	1681	404990	42	17011344
WSJPOS	35531	1681	446180	42	18741324
CoNLL	8936	2012	1679679	22	36953422
Brown	48242	9098	290843	185	53840180

partition and POS and CoNLL datasets. We compared their performance in terms of the number of non-zero features (#NZ), test accuracy and time. For our SAP algorithm, we fixed the values of C to be 1 for OCR and 0.1 for POS and CoNLL. We set $\rho_1 + \rho_2 = 1$ with $\rho_1 = 0.7$ for OCR and $\rho_1 = 0.9$ for POS and CoNLL datasets. We did this so that the L1-term $\|\mathbf{w}\|_1$ gets the maximum weight and aids in selecting a small number of features. For the projected sub-gradient method, we used cross-validation to choose the value of ϵ.

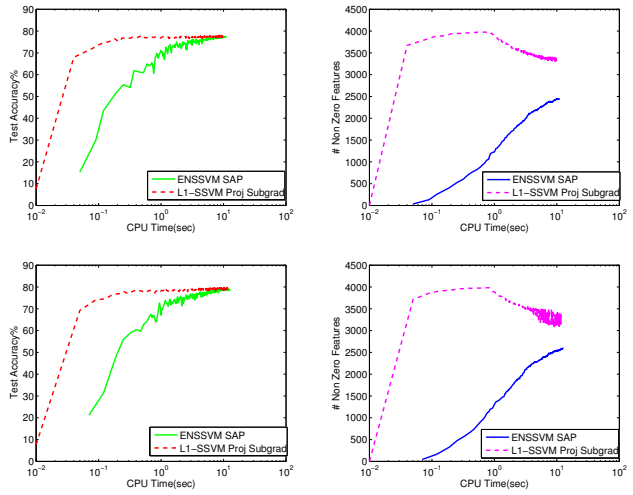

Figure 1. **Comparison of** *Sequential Alternating Proximal method for Dual ENSSVM* **and** *Projected sub-gradient method for L1-SSVM.* **The rows correspond to the OCR data partitions OCR0 and OCR6 respectively.**

We give the plots for test set accuracy and the number of non-zero features selected by both the methods for some OCR partitions in Figure 1. It is evident from the plots that the SAP algorithm for Dual ENSSVM is able to generalize well with a smaller number of features when compared to the projected sub-gradient method for L1-SSVM. The sparsity plots of Figure 1 indicate that the SAP method displays a very good control on the number of non-zero model parameters selected in the initial iterations whereas such a behavior is not seen in the projected sub-gradient method. We also see from the plots that nearly all the features are selected by the projected sub-gradient algorithm during the initial stages of learning, which does not happen if the SAP algorithm is used. We give the results for POS and CoNLL datasets in Table II. These results indicate that, in spite of the projection step being carried out after a complete pass through all the examples, L1-SSVM is too slow on large datasets and fails to achieve highly sparse models which give a comparable generalization performance when compared to those obtained by Dual ENSSVM.

Table II
COMPARISON OF ENSSVM AND L1-SSVM ON LARGE DATASETS. #NZ DENOTES THE NUMBER OF NON-ZERO FEATURES SELECTED.

Data set	Algorithm	#NZ	Accuracy %	Time
POS	ENSSVM	3089	94.17	≈6 hours
	L1-SSVM	3924	94.14	>3 days
CoNLL	ENSSVM	2313	95.2	≈22 hours
	L1-SSVM	9398	95.2	>4 days

We also implemented the extra-gradient method for L1 regularized SSVM given in [20] by adapting the implementation available at [19]. However the extra-gradient method was observed to be very slow on small datasets like OCR. Hence we do not report those results here.

2) Performance of the SAP algorithm on large datasets: One notable problem that we encountered when we tried L1-SSVM on very large datasets was that various methods used to solve L1-SSVM were not scaling well with the size of the data. For example, on the Brown dataset, the projected sub-gradient method for L1-SSVM took more than 10 days to achieve a sparse model with a reasonable generalization performance. The extra-gradient method was also not scalable on very large datasets. However the proposed SAP algorithm was found to be scalable even on very large datasets. We provide the plots in Figure 2 indicating the performance of the SAP algorithm on very large datasets. For these experiments we considered $\rho_1 = 0.9$, $\rho_2 = 0.1$ and $C = 1$. Since our algorithm is sequential in nature, we get a reasonable generalization performance in the first couple of iterations itself with highly sparse models. This is obviously seen for all the datasets in Figure 2. As the plots reveal, though many features are required for the best test accuracy at the final stages of the algorithm, the SAP algorithm achieves a comparable performance in the first few iterations with a very small number of features. Hence our algorithm gives the user a wide control over the number of non-zero model parameters to be selected without significant degradation in the generalization performance.

Table III
PERFORMANCE OF SAP ON LARGE DATASETS. #NZ AND % NZ DENOTE RESPECTIVELY, THE NUMBER AND PERCENTAGE OF NON-ZERO FEATURES SELECTED.

Data set	#NZ	% NZ	Accuracy %
POS	3365	0.02	94.5
WSJPOS	8906	0.05	96
CoNLL	2981	0.01	95.4
Brown	17407	0.032	96.7

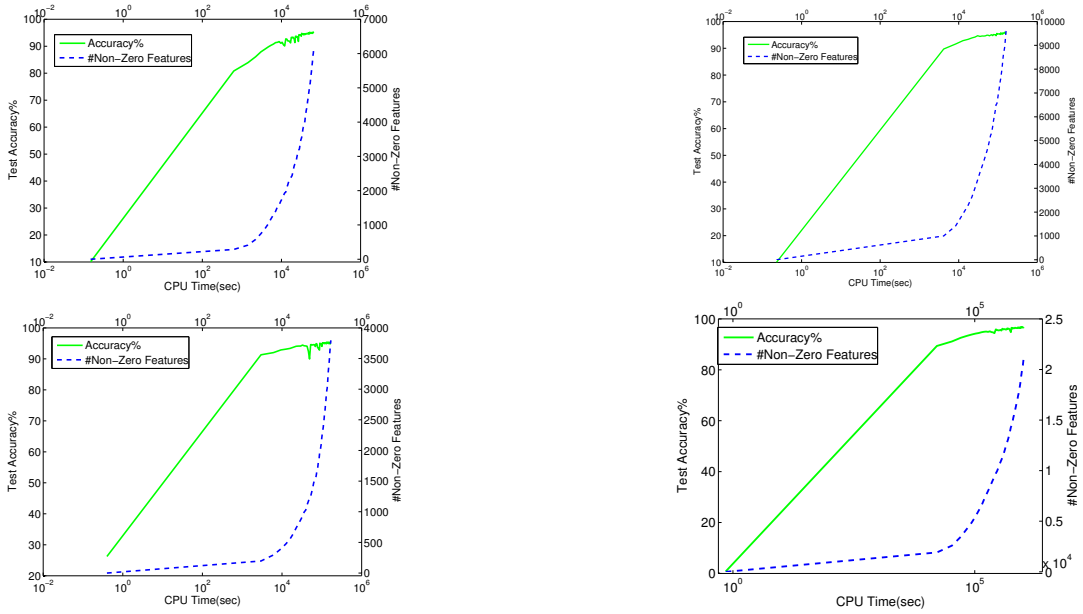

Figure 2. **Performance of** *Sequential Alternating Proximal method for Dual ENSSVM* **on large datasets with C=1. The plots correspond to the data sets** *Row 1*: **POS and WSJPOS**, *Row 2*: **CoNLL and Brown.**

Since our method is able to generalize well within the first few iterations of the learning phase, we calculate the maximum test accuracy within the first 50 iterations, the corresponding number of non-zero features selected and the percentage of non-zero features with respect to the total number of features to achieve the maximum test set accuracy. We give these details for all the large datasets in Table III. These details also show that SAP for dual ENSSVM achieves a comparable test set generalization with a very small number of features for all large datasets within the first few iterations.

3) Effect of ρ_1 and ρ_2 parameters on the SAP algorithm for Dual ENSSVM: Having discussed the efficacy of the Sequential Alternating Proximal algorithm in selecting a very small number of features for the dual ENSSVM, we now discuss the effect of parameters ρ_1 and ρ_2 which appear as coefficients of the L1 and L2 regularizer terms in OP5. The regularization coefficient C was fixed to be 0.1 for these experiments. We performed the experiments on POS and CoNLL datasets and present the results in Figure 3. In the plots, we represent the test accuracy and the corresponding number of non-zero features obtained for different values of ρ_1. The parameter ρ_1 was set to different values from the set $\{0.9, 0.5, 0.3, 0.1\}$. From the plots, we notice that as we decrease $\rho 1$ value, the number of non-zero model parameters increases. However, this has a very little effect on the test set generalization performance. Hence a very small number of model parameters selected with the choices $\rho 1 = 0.9$ and 0.5 are sufficient to achieve a comparable generalization performance.

4) Effect of regularization coefficient C on the SAP algorithm for Dual ENSSVM: Other than ρ_1 and ρ_2, the dual ENSSVM problem contains the regularization coefficient C which is also tunable. We performed experiments with different values of C on the CoNLL and POS datasets. The values ρ_1 and ρ_2 were fixed to be 0.9 and 0.1 respectively for these experiments. We performed experiments for the first 100 iterations and give details of sparsity required for achieving the best test set generalization performance in the first 100 iterations in Table IV. It is clear from Table IV that the generalization performance achieved using various C values is comparable.

Table IV
EFFECT OF C ON VARIOUS DATASETS. #NZ AND % NZ DENOTE RESPECTIVELY, THE NUMBER AND PERCENTAGE OF NON-ZERO FEATURES SELECTED.

		#NZ	% NZ	Accuracy %
POS	$C = 0.1$	3967	0.02	94.57
	$C = 1$	5104	0.03	95.21
	$C = 10$	5110	0.03	95.14
CoNLL	$C = 0.1$	2313	0.006	95.23
	$C = 1$	2981	0.008	95.41
	$C = 10$	4107	0.01	95.53

VI. CONCLUSION

In this work, we formulated the elastic net regularized Structural SVM (ENSSVM) and proposed a sequential alternating proximal (SAP) algorithm to solve the dual problem. The SAP algorithm works by sequentially visiting each training example and by solving a restricted sub-problem associated with that example. The sub-problem is made up of

Figure 3. **Effect of various ρ_1 values on *SAP* method for Dual ENSSVM** *Left*: **ρ_1 variation on POS dataset**, *Right*: **ρ_1 variation on CoNLL dataset**.

two different sets of variables and the SAP algorithm solves the sub-problem by alternating between the two sets of variables. Experiments on large-scale benchmark datasets show that the proposed SAP algorithm for dual ENSSVM scales well and achieves a comparable generalization performance with very sparse models, compared to existing algorithms for L1-regularized Structural SVM. The sequential nature of the dual SAP algorithm helps in achieving state-of-the-art generalization performance in the first few iterations itself. Thus the proposed SAP algorithm for dual ENSSVM formulation is a powerful alternative to design a scalable sparse structured output classifier.

ACKNOWLEDGMENT

The work of the first author was partially supported by the grant from Infosys Ltd., India.

REFERENCES

[1] H. Attouch, J. Bolte, P. Redont and A. Soubeyran. Alternating proximal algorithms for weakly coupled convex minimization problems. Applications to dynamical games and PDE's. Journal of Convex Analysis 15 (3), 485-506. 2008.

[2] P. Balamurugan, S. Shevade, S. Sundararajan and S. S. Keerthi. A Sequential Dual Method for Structural SVMs. SDM, 2011.

[3] L. Bottou. Large-Scale Machine Learning with Stochastic Gradient Descent. COMPSTAT, 2010.

[4] P. S. Bradley and O. L. Mangasarian. Feature Selection via concave minimization and support vector machines. ICML, 1998.

[5] Brown Corpus. Available at `http://nltk.googlecode.com/svn/trunk/nltk_data/index.xml`

[6] J. Duchi. Matlab code to project onto L1 ball. Available at `http://www.cs.berkeley.edu/~jduchi/projects/DuchiShSiCh08/ProjectOntoL1Ball.m`

[7] J. Duchi, S. Shalev-Shwartz, Y. Singer, T. Chandra. Efficient Projections onto the l_1-Ball for Learning in High Dimensions. ICML, 2008.

[8] Z. Hussain and J. Shawe-Taylor. Metric learning analysis. PinView FP7-216529 Project Deliverable Report D3.3, 2011.

[9] T. Joachims, T. Finley and C-N. J. Yu. Cutting-Plane training of Structural SVMs. Machine Learning, 77(1):27-59, 2009.

[10] T. Lavergne, O. Cappé and F. Yvon. Practical Very Large CRFs. Proceedings of the ACL, pages 504-513, 2010.

[11] M. Marcus, B. Santorini and M. A. Marcinkiewicz. Building a Large Annotated Corpus of English. Computational Linguistics, 1993.

[12] N. Nguyen and Y. Guo. Comparisons of Sequence-labeling Algorithms and Extensions. ICML, 2007.

[13] J. Platt. Fast Training of SVMs using Sequential Minimal Optimization. Advances in Kernel Methods-Support Vector Learning, 1999.

[14] E. F. T. K. Sang and S. Buchholz. Introduction to the CoNLL-2000 shared task: Chunking. CoNLL, 2000.

[15] B. Taskar, C. Guestrin and D. Koller. Maximum-margin Markov networks. NIPS, 2003.

[16] I. Tsochantaridis, T. Joachims, T. Hoffmann and Y. Altun. Large Margin Methods for Structured and Inter-dependent Output Variables. JMLR, 6:1453:1484, 2005.

[17] L. Wang (Ed.), Support Vector Machines: Theory and Applications. Springer, 2005.

[18] L. Wang, J. Zhu, and H. Zou. The Doubly Regularized Support Vector Machine. Statistica Sinica, 16(2), 589-616, 2006.

[19] Z. Wang. Extra-gradient solver for LP. Available at `http://www0.cs.ucl.ac.uk/staff/Zhuoran.Wang`

[20] Z. Wang and J. Shawe-Taylor. Large-Margin Structured Prediction via Linear Programming. AISTATS, 2009.

[21] W. Yin. Analysis and Generalizations of the Linearized Bregman Method. SIAM Journal of Imaging Sciences, Vol. 3, No. 4, pp. 856-877. 2010.

[22] J. Zhu, E. P. Xing and B. Zhang. Primal Sparse Max-Margin Markov Networks. SIGKDD, 2009.

[23] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *J. Roy. Statist. Soc. Ser.* B 67, 301-320, 2005.