Weighted Coordinate-Wise Pegasos

Vilen Jumutc, Johan A.K. Suykens

Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, B-3001, Leuven, Belgium
{vilen.jumutc, johan.suykens}@esat.kuleuven.be

Abstract. Pegasos is a popular and reliable machine learning algorithm for making linear Support Vector Machines solvable at the larger scale. It benefits from the strongly convex optimization objective, faster convergence rates and lower computational and memory costs. In this paper we devise a new weighted formulation of the Pegasos algorithm which favors from the different coordinate-wise λ_i regularization parameters. Together with the proposed extension we give a brief theoretical justification of its convergence to an optimal solution and analyze at a glance its computational costs. We conclude our paper with the numerical results obtained for UCI datasets and demonstrate the merits and the importance of our approach for achieving a better classification accuracy and convergence rates in the partially or fully stochastic setting.

1 Introduction

Recent advances in linear Support Vector Machines (SVM) and the first order stochastic optimization [1–4] unlocked a room for improvement in the large-scale machine learning and black-box modelling. Usually computation of a full gradient for such large-scale problems is not bearable on a single machine, so one might consider to use a stochastic approximation to the original problem. In expectation [2, 4] such approach converges to the global optima if we consider strongly convex optimization objectives. Additionally the latter approach considerably saves the memory and only moderately increases the number of iterations to converge.

In this paper we propose a new weighted formulation of the Pegasos optimization objective and revise some of the algorithmic steps in order to maintain the consistency with the underlying theory. The key feature of the Pegasos [2] algorithm is a strongly convex optimization objective with the proper projection step. This combination helps to achieve a solution of accuracy ϵ in $O(\frac{R^2}{\lambda\epsilon})$ iterations where λ is the regularization parameter. On the other hand this approach imposes uniformed convergence speed for every single dimension. In general such uniform convergence is useful but doesn't reflect the importance and contribution of each dimension to the final classification result. The latter problem in Bayesian inference relates to Automatic Relevance Determination (ARD) [5–7]. We will compare this approach with the tuning via cross-validation for finding optimal λ_i hyperparameters.

The task of inferring optimal tuning parameters for our weighted coordinate-wise Pegasos is interesting by itself. In this paper we will highlight recent research in this subject and stress the difference in assumptions and behavior of Bayesian approach and tuning via global optimization techniques like Coupled Simulated Annealing (CSA) [8]. In our formulation of the Pegasos algorithm we deal with the linear approach and we are interested in obtaining the fittest regularization parameters λ_i *w.r.t* generalization and training speed-up.

This paper is organized as follows. Section 2 outlines original and Weighted Coordinate-Wise Pegasos formulations, explains in detail our algorithm and gives a theoretical background, *i.e.* convergence proofs. Section 3 highlights Automatic Relevance Determination and Coupled Simulated Annealing used for tuning λ_i regularization hyperparameters. Experimental setup and numerical results are given in Section 4 while Section 5 concludes the paper.

2 Weighted Coordinate-Wise Pegasos

Our approach is based on a simple assumption related to Automatic Relevance Determination. If we impose regularization differently for each dimension or data point we end up with the method which has a weighted impact of the features, dimensions or data samples on the classification output. In the case of the dual representation of SVMs we might talk about box constraints [9] which are related to the dual α unknowns. The larger is the upper bound of the particular box constraint the greater might be the influence of that data point on the decision boundary. On the other hand with ARD techniques and Bayesian inference one maximizes the posterior and the evidence can be used to assign a preference to alternative values of the hyperparameters [5].

To obtain such weighted formulation for the Pegasos algorithm first we propose a new optimization objective

$$f_{wcw}(w; \mathcal{A}_t) = \frac{1}{2} w^T \Lambda w + \frac{1}{|\mathcal{A}_t|} \sum_{(x,y) \in \mathcal{A}_t} \mathbb{L}(w; (x, y)), \tag{1}$$

where f_{wcw} stands for our new "weighted coordinate-wise" instantaneous optimization objective and it differs *w.r.t* original Pegasos objective only in the way we define our hyperparameters. Here Λ stands for the diagonal matrix with entries corresponding to coordinate-wise λ_i regularization hyperparameters.

Now we can see that the subgradient term becomes

$$\nabla_t = \Lambda w_t - \frac{1}{|\mathcal{A}_t|} \sum_{(x,y) \in \mathcal{A}_t^+} yx. \tag{2}$$

These modifications to the Pegasos algorithm do not guarantee a convergence without further modifications to the projection step defined in the original Pegasos as a projection onto the set $\mathcal{B} = \{w : \|w\| \leq 1/\sqrt{\lambda}\}$. In the next subsection we will take a closer look at our Weighted Coordinate-Wise Pegasos algorithm and present these missing details.

Algorithm 1: Weighted Coordinate-Wise Pagasos

Data: $\mathcal{S}, \Lambda, T, k, \epsilon$

1 Set $\lambda_{min} = \min_i \Lambda_{ii}$
2 Select w_1 randomly s.t. $\|w^{(1)}\| \leq 1/\sqrt{\lambda_{min}}$
3 **for** $t = 1 \rightarrow T$ **do**
4 Set $\eta_t = \frac{1}{t}\Lambda^{-1}$
5 Select $\mathcal{A}_t \subseteq \mathcal{S}$, where $|\mathcal{A}_t| = k$
6 $\rho = \frac{1}{|\mathcal{S}|}\sum_{(x,y)\in\mathcal{S}}(y - \langle w_t, x\rangle)$
7 $\mathcal{A}_t^+ = \{(x,y) \in \mathcal{A}_t : y(\langle w_t, x\rangle + \rho) < 1\}$
8 $w_{t+\frac{1}{2}} = w_t - \eta_t(\Lambda w_t - \frac{1}{k}\sum_{(x,y)\in\mathcal{A}_t^+} yx)$
9 $w_{t+1} = \min\left\{1, \frac{1/\sqrt{\lambda_{min}}}{\|w_{t+\frac{1}{2}}\|}\right\} w_{t+\frac{1}{2}}$
10 **if** $\|w_{t+1} - w_t\| \leq \epsilon$ **then**
11 **return** $(w_{t+1}, \frac{1}{|\mathcal{S}|}\sum_{(x,y)\in\mathcal{S}}(y - \langle w_t, x\rangle))$
12 **end**
13 **end**
14 **return** $(w_{T+1}, \frac{1}{|\mathcal{S}|}\sum_{(x,y)\in\mathcal{S}}(y - \langle w_t, x\rangle))$

2.1 The algorithm

Hereby we present a brief summary of the modified Pegasos in Algorithm 1. Next we continue with the theoretical guarantees for convergence of this algorithm in the next subsection. In Algorithm 1 we can see a major "for" loop where gradient and projection steps are taking place and a minor "if" condition terminates an execution if the norm of the difference of two subsequent w vectors is less than ϵ. In Algorithm 1 we denote the whole dataset by $\mathcal{S}$ and at each iteration select randomly k samples for computation of the subgradient in Eq.(2). We should emphasize the importance of the projection step at the Line 9 where we are projecting our solution back on the ball with the radius $1/\sqrt{\lambda_{min}}$ and updated step size at the Line 4.

2.2 Analysis

This subsection presents a convergence analysis and theoretical background which brings our algorithm to the similar bounds as in the original paper of Shalev-Shwartz *et al.* [2]. We extend the analysis given in their paper to our new instantaneous objective by presenting Theorem 1.

Theorem 1. *Let $\mathcal{B}$ be a closed convex set and define $\prod_{\mathcal{B}}(w) = arg\min_{w'\in\mathcal{B}} \|w - w'\|$. Let $w_1, \ldots, w_T$ be a sequence of vectors such that $w_1 \in \mathcal{B}$ and for $t \geq 1$, $w_{t+1} = \prod_{\mathcal{B}}(w_t - \eta_t\nabla_t)$, where ∇_t is a subgradient of Eq.(1) at w_t and $\eta_t = \frac{1}{t}\Lambda^{-1}$. Let w^* be the solution of the optimization problem in Eq.(1), $\lambda_{min} = \min_i \Lambda_{ii}$, $\lambda_{max} = \max_i \Lambda_{ii}$, $\lambda_{min} > 0$ and let $G = \lambda_{max}/\sqrt{\lambda_{min}} + R$. Assume $\|x\| \leq R$ for all $(x,y) \in \mathcal{S}$. Then, for $T \geq 1$ we have*

$$\|w_T - w^*\| \leq 2G\frac{\lambda_{max}}{\lambda_{min}^2}(\ln(T) + 1) + \frac{2R}{\lambda_{min}}.$$

Proof. To prove our theorem we refer to Lemma 4 in [11] and Section 9.1.2 in [12]. To apply results from Boyd's book [12] first we need to show that our problem is strongly convex. It is easy to verify that the first term in Eq.(1) is λ_{min}-strongly convex with respect to w. Since f is a sum of λ_{min}-strongly convex function and another convex function (hinge loss), it is also strongly convex. From Section 9.1.2 in [12] we know that one can bound the distance to the minimizer w^* of the objective in Eq.(1) by

$$\|w_T - w^*\| \leq \frac{2}{\lambda_{min}} \|\nabla f(w_T)\|.$$

The simplistic bound on $\|\nabla f(w_T)\|$ is derived using the upper-bound on the matrix-vector product as follows

$$\|\nabla f(w_T)\| \leq \|\Lambda\| \|w_T\| + \frac{1}{|\mathcal{A}_T|} \sum_{(x,y) \in \mathcal{A}_T^+} \|yx\| \leq \lambda_{max} \|w_T\| + R.$$

Next using the fact that there exists t-independent bound on $\|\nabla f(w)\| \leq G$, $w_t = w_{t-1} - \eta_t \nabla_t$, $w_0 = 0$ and applying induction and the triangular inequality one can show that

$$\|w_T\| \leq \sum_{t=1}^{T} \|\eta_t \nabla f(w_t)\| \leq G \|\Lambda^{-1}\| \sum_{t=1}^{T} \frac{1}{t} \leq \frac{G}{\lambda_{min}} (\ln(T) + 1).$$

Next by assuming $\mathcal{B} = \{w : \|w\| \leq 1/\sqrt{\lambda_{min}}\}$ and the fact that $\|x\| \leq R$ we can derive t-independent bound on subgradient $\nabla f(w)$. The explicit form for the subgradient evaluated at intermediate solution w_t is given by Eq.(2). Using the triangular inequality, taking into account that Λ is a diagonal matrix one obtains

$$\|\nabla f(w)\| \leq \|\Lambda\| \|w\| + \tfrac{1}{|\mathcal{A}_t|} \textstyle\sum_i \|x_i\| \leq \lambda_{max} \|w\| + R \leq \lambda_{max} / \sqrt{\lambda_{min}} + R.$$

Finally we have to show that $w^* \in \mathcal{B}$ and that there exist a universal upper bound for $\|w\|$. As a starting point we take Lemma 4 in [11]. First we need to derive the dual form of our instantaneous optimization objective for any subset $\mathcal{A}_t$ at iteration t. We use the fact that there exists a vector $\alpha^* \in [0,1]^m$ which maximizes our dual objective and duality gap equals to zero. Assuming these conditions and $|\mathcal{A}_t| = k$ one gets

$$\frac{1}{2} w^{*T} \Lambda w^* + \frac{1}{k} \sum_{(x,y) \in \mathcal{A}_t} \mathbb{L}(w^*; (x,y)) = -\frac{1}{2} w^{*T} \Lambda w^* + \frac{1}{k} \|\alpha^*\|_1.$$

Rearranging the above, using the non-negativity of the hinge loss, the bound on $\|\alpha\|_1 \leq k$ and the fact that $w^{*T} \Lambda w^* \geq \lambda_{min} \|w^*\|^2$ we obtain our initial condition of set $\mathcal{B}$: $\|w\| \leq 1/\sqrt{\lambda_{min}}$. Now we can plug-in everything back to the inequality in Theorem 1 and complete the proof by putting all together.■

3 Obtaining λ_i hypermarameters

In this section we focus on the problem of estimating λ_i hyperparameters. This problem is interesting by itself and gained a lot of attention within the framework of Automatic Relevance Determination in neural networks and kernel methods [5–7]. In our experiments with Weighted Coordinate-Wise Pegasos algorithm we will use two distinct strategies for obtaining these hyperparameters.

The first one is related to cross-validation and global optimization techniques, like Coupled Simulated Annealing [8]. Following this approach we evaluate 10-fold cross-validation for each iteration in every standard simulated annealing process. In CSA one couples together acceptance probability functions of these processes in order to control general statistical measures that may have crucial influence on the performance of the optimization.

The other technique is purely Bayesian and relies on the recent research in this area [13]. The main idea of the latter approach is to estimate these hyperparameters from the data by first marginalizing over the coefficients w and then performing what is commonly referred to as evidence maximization or type-II maximum likelihood [6, 14, 15]. Mathematically, in the context of Weighted Coordinate-Wise Pegasos this formulation is equivalent to minimizing

$$\mathcal{L}(\lambda) \triangleq -\log p(y;\lambda) \equiv \log|\Sigma_y| + y^T \Sigma_y^{-1} y, \tag{3}$$

where a flat hyperprior on λ_i hyperparameters is assumed and $\Sigma_y \triangleq \gamma I + \Phi\Lambda\Phi^T$. Here Φ denotes evaluated dataset and is given by the original mapping in the input space, *i.e.* $\phi(x) = x$, where $(x, y) \in \mathcal{S}$ as it is given in Theorem 1. In this setting we are following the proposed optimization via Iterative Re-Weighted Minimum l_1 in [13], where one is estimating the λ_i unknowns via alternating update rules which can be derived by decoupling $\mathcal{L}(\lambda)$ using upper bounding functions. This approach also relates to Sparse Bayesian Learning (SBL) that has been successful in a variety of applications [6].

4 Experiments

4.1 Setup

For the first part of our experiments we used a 2-step procedure for tuning the λ_i hyperparameters in Algorithm 1. This procedure consists of Coupled Simulated Annealing [8] initialized with 5 random sets of parameters for the first step and the simplex method [16] for the second step. After CSA converges to some local minima we select a tuple of λ_i hyperparameters which attains the lowest cross-validation error and start the simplex procedure to refine our selection. On every iteration step for CSA and simplex method we proceed with a 10-fold cross-validation.

For the second part of our experiments we used the ARD approach described in Section 3. For learning hyperparameters we used the procedure described in Section 2.1 of [13] and we defined $\gamma = 10^{-3}$ parameter related to our flat

hyperprior in Eq.(3) (we can bind this parametrization to the numerical stability of the underlying approach). Finally we defined $\epsilon = 10^{-5}$ stopping criterion which controls the norm of the difference for two subsequent solutions. Once we drop below this threshold iterative re-weighting procedure halts. In this part we experimented mostly with small-scale UCI datasets because of the computational burden of Σ_y matrix inversion and storage.

All experiments with large-scale UCI datasets were repeated 50 times with the random split to training and test sets in proportion 1:1. For the smaller UCI datasets we performed 50 iterations with splitting in proportion 1:9 taking 90% of the dataset for training. In the presence of 3 or more classes we performed binary classification where we learned to classify the first class versus all others. Only for Pen Digits dataset we performed several experiments where this setting was changed. For Algorithm 1 we fixed parameters: $T = 1000$ and $\epsilon = 10^{-5}$. Parameter k was set to $|\mathcal{S}|/10$ for all small-scale UCI datasets and we performed experiments with two different values of k for larger-scale UCI datasets.

4.2 Results

In this subsection we give numerical results on running Algorithm 1 within cross-validation and ARD settings. In Table 1 we can see our results for small-scale UCI datasets. Notation Pegasos$_{wcw}$ denotes our proposed approach with the tuning of λ_i hyperparameters via cross-validation and Pegasos$_{wcw/ard}$ stands for the same approach with hyperparameters obtained via the ARD method described in Section 3.

From Table 1 we can notice that Bayesian inference together with Algorithm 1 doesn't attain better results than the basic Pegasos algorithm. On the other hand Weighted Coordinate-Wise Pegasos performs better with respect to both of them. In Table 3a we can observe results for large-scale UCI datasets within

Table 1: Test errors for small-scale datasets

Dataset	Pegasos	Pegasos$_{wcw}$	Pegasos$_{wcw/ard}$
Ionosphere	0.16889	**0.12714**	0.28922
Parkinsons	0.20832	**0.18274**	0.23253
Sonar	**0.23848**	0.28667	0.32176
Iris	0.34933	**0.30133**	0.61467
Ecoli	0.06500	**0.05710**	0.16299

partially stochastic setting where we set k to be 10% of $|\mathcal{S}|$. We can notice that we perform equally better for almost all datasets in comparison with original Pegasos algorithm.

In Table 3b we can see that when going to a completely stochastic setting with $k = 1$ we are slightly deteriorating our performance. We can notice that our approach outperforms Pegasos algorithm for all datasets and it is well-suited for learning more accurately large scale linear SVMs via stochastic programming.

Table 2: Test errors for larger-scale datasets

Dataset	Pegasos	Pegasos$_{wcw}$
Magic	0.30550	**0.23607**
Shuttle	0.21450	**0.08549**
Red Wine	**0.26924**	0.27747
White Wine	0.32443	**0.30541**
Covertype	0.36466	**0.32807**
Pen Digits (1 vs all)	0.10432	**0.08984**
Pen Digits (2 vs all)	0.08448	**0.05133**
Pen Digits (5 vs all)	0.09609	**0.06830**
Pen Digits (6 vs all)	0.05835	**0.02896**

(a) $k = 10\%$ of $|\mathcal{S}|$ (partially stochastic)

Dataset	Pegasos	Pegasos$_{wcw}$
Magic	0.32288	**0.27743**
Shuttle	0.21751	**0.08598**
Red Wine	0.29552	**0.29224**
White Wine	0.32807	**0.30599**
Covertype	0.36474	**0.34962**
Pen Digits (1 vs all)	0.10409	**0.09255**
Pen Digits (2 vs all)	0.08437	**0.05317**
Pen Digits (5 vs all)	0.09635	**0.07059**
Pen Digits (6 vs all)	0.05863	**0.02847**

(b) $k = 1$ (fully stochastic)

5 Conclusions and remarks

In this paper we proposed an extension of the Pegasos algorithm which is suitable for learning linear SVMs at the larger scale. It performs considerably better in terms of the generalization error. We presented two different approaches for learning λ_i regularization hyperparameters and showed that cross-validation together with CSA attains better results than a Bayesian approach. The latter results can be explained by the nature of ARD-related data-dependent prior distribution and inconsistencies between our assumptions and observed data. Finally we gave numerical results which demonstrated the merits of the proposed methods and verified the importance of the coordinate-wise parametrization for linear SVMs in terms of generalization error and training speed-up.

Acknowledgments

This work was supported in part by the scholarship of the Flemish Government; Research Council KUL: GOA/11/05 Ambiorics, GOA/10/09 MaNet, CoE EF/05/006 Optimization in Engineering (OPTEC), IOF-SCORES4CHEM, several PhD/postdoc & fellow grants; Flemish Government:FWO: PhD/postdoc grants, projects: G0226.06 (cooperative systems and optimization), G.0302.07 (SVM/Kernel), G.0320.08 (convex MPC), G.0558.08 (Robust MHE), G.0557.08 (Glycemia2), G.0588.09 (Brain-machine) research communities (WOG: ICCoS, ANMMM, MLDM); G.0377.09 (Mechatronics MPC), G.0377.12 (Structured models), IWT: PhD Grants, Eureka-Flite+, SBO LeCo-Pro, SBO Climaqs, SBO POM, O&O-Dsquare; Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, Dynamical systems, control and optimization, 2007-2011); IBBT; EU: ERNSI; ERC AdG A-DATADRIVE-B, FP7-HD-MPC (INFSO-ICT-223854), COST intelliCIS, FP7-EMBOCON (ICT-248940); Contract Research: AMINAL; Other: Helmholtz: viCERP, ACCM, Bauknecht, Hoerbiger. Johan Suykens is a professor at KU Leuven, Belgium.

References

1. Joachims, T.: Training linear SVMs in linear time. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. KDD '06, New York, NY, USA, ACM (2006) 217–226
2. Shalev-Shwartz, S., Singer, Y., Srebro, N.: Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In: Proceedings of the 24th international conference on Machine learning. ICML '07, New York, NY, USA (2007) 807–814
3. Chapelle, O.: Training a support vector machine in the primal. Neural Computation **19** (2007) 1155–1178
4. Nemirovski, A., Juditsky, A., Lan, G., Shapiro, A.: Robust stochastic approximation approach to stochastic programming. SIAM J. on Optimization **19**(4) (January 2009) 1574–1609
5. MacKay, D.J.: Comparison of approximate methods for handling hyperparameters. Neural Computation **11** (1999) 1035–1068
6. Tipping, M.E.: Sparse bayesian learning and the relevance vector machine. J. Mach. Learn. Res. **1** (September 2001) 211–244
7. Chu, W., Keerthi, S.S., Ong, C.J.: Bayesian trigonometric support vector classifier. Neural Comput. **15**(9) (September 2003) 2227–2254
8. Xavier-De-Souza, S., Suykens, J.A.K., Vandewalle, J., Bollé, D.: Coupled simulated annealing. IEEE Trans. Sys. Man Cyber. Part B **40**(2) (April 2010) 320–335
9. Vapnik, V.: Statistical learning theory. 1 edn. Wiley (September 1998)
10. Suykens, J.A.K., Van Gestel, T., De Brabanter, J., De Moor, B., Vandewalle, J.: Least Squares Support Vector Machines. World Scientific, Singapore (2002)
11. Shalev-Shwartz, S., Singer, Y.: Logarithmic regret algorithms for strongly convex repeated games. Technical report, The Hebrew University (2007)
12. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York, NY, USA (2004)
13. Wipf, D., Nagarajan, S.: A New View of Automatic Relevance Determination. In Platt, J.C., Koller, D., Singer, Y., Roweis, S., eds.: Advances in Neural Information Processing Systems 20. MIT Press, Cambridge, MA (2008) 1625–1632
14. MacKay, D.J.: Bayesian interpolation. Neural Computation **4** (1991) 415–447
15. Neal, R.M.: Bayesian Learning for Neural Networks. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1996)
16. Nelder, J.A., Mead, R.: A simplex method for function minimization. Computer Journal **7** (1965) 308–313