Variants of Pegasos

SooWoong Ryu Youngsoo Choi
bshboy@stanford.edu yc344@stanford.edu

December 11, 2009

1 Introduction

Developing a new SVM algorithm is ongoing research topic. Among many exiting SVM algorithms, we will focus on Pegasos. Pegasos uses alternating iteration schemes, which are by alternating stochastic gradient descent steps and projection step. The Pegasos can solve a text classification problem from Reuters Corpus Volume 1 with 800,000 training examples in 5 seconds. We will examine the algorithms robustness by applying the algorithm to rather simple text classification problem (namely, the one given in the cs229 homework 2). We also present three possible variants of Pegasos; namely, *undo* method, *two-out* method and *probability* method. We will examine con and pro of each method.

2 Algorithm description

2.1 Pegasos

The Pegasos is a subgradient based algorithm that minimizes the following unconstrained strictly convex objective function.

$$\mathrm{f}(\omega) = \frac{\lambda}{2}\|\omega\|^2 + \frac{1}{m}\sum_{(x,y)\in S} l(\omega;(x,y)) \tag{1}$$

where

$$l(\omega;(x,y)) = \max\{0, 1 - y\langle\omega, x\rangle\} \tag{2}$$

However, it does not use eq(1) as an objective function at each iteration. Instead, it uses

$$\mathrm{f}(\omega;A_t) = \frac{\lambda}{2}\|\omega\|^2 + \frac{1}{k}\sum_{(x,y)\in A_t} l(\omega;(x,y)) \tag{3}$$

where a set $A_t \subseteq S$ whose size is k and S, a training set. The Pegasos consists of two major steps in updating ω_t : 1. substeepest gradient update $\omega_{t+\frac{1}{2}}$ and 2. projection step. Specifically, in each iteration, the Pegasos looks for a steepest gradient search direction given A_t and sets the learning rate to be $\eta_t = \frac{1}{\lambda t}$. Then, it projects the updated $\omega_{t+\frac{1}{2}}$ into the ball, $B = \{\omega : \|\omega\| \le \frac{1}{sqrt(\lambda)}\}$. The reason for projection is that we already know the optimal solution, ω^*, resides in B.

2.2 A Variant of Pegasos

A key point in a variant of pegasos is how to deal with outliers in training set, S. There are outliers in S. These outliers result in increase in objective function value in some iteration since pegasos chooses A_t i.i.d. from S. To prevent this, three methods are used; 1. *Undo* method 2. *two-out* method and 3. *probability* method. A modification from pegasos is applied at the end of iteration when ω_{t+1} is computed. In the *undo* method, with computed ω_{t+1}, the new objective function value, $f(\omega_{t+1})$, is also computed. If $f(\omega_{t+1}) > f(\omega_t)$ then we do not use computed ω_{t+1}. Instead we let ω_{t+1} to be equal to ω_t. If $f(\omega_{t+1}) \le f(\omega_t)$, then we keep

ω_{t+1} as updated. In the *two-out* method, the training set, S, is updated at each iteration. If $f(\omega_{t+1}) > f(\omega_t)$, then we do not use compute ω_{t+1} and also append a flag to each data, (x_i, y_i) in A_{t+1} and count the number of flags in each data in A_{t+1}. If the number of flags in a data becomes two (i.e $n_f(x_i, y_i) = 2$), the data is taken away from S_{t+1}. Let D_{t+1} to be a set of those data that are taken away from S_{t+1}. Then, we obtain $S_{t+2} = S_{t+1} / D_{t+1}$. At the iteration of t + 2, S_{t+2} is used as a training set. Again, if $f(\omega_{t+1}) \le f(\omega_t)$, then we keep ω_{t+1} as updated as in *undo* method. The *probability* method is similar to the *two-out* method. The difference is in how to update S. In the *probability* method, for each data, (x_i, y_i), the probability of being sampled to A_t at t iteration (let's call that probability $p(x_i, y_i)$) is set to 1 in the beginning. As the iteration proceeds, these probabilities are updated by examining the objective function value. If $f(\omega_{t+1}) > f(\omega_t)$, then we decrease $p(x_i, y_i)$ for each (x_i, y_i) in A_t. The variant algorithm is shown below.

INPUT: S, λ, T, k

INITIALIZE: Choose w_1 s.t. $\|w_1\|^2 \le \frac{1}{\sqrt{\lambda}}$

FOR $t = 1, 2, 3, ..., T$

Choose $A_t \subseteq S$, where $|A_t| = k$.

Set $A_t^+ = \{(x, y) \in A_t : y \langle w_t, x \rangle < 1\}$, $\eta = \frac{1}{\lambda t}$

Set $w_{t+\frac{1}{2}} = (1 - \eta_t \lambda) w_t + \frac{\eta_t}{k} \sum_{(x,y) \in A_t^+} yx$

IF $f(\omega_{t+1}) > f(\omega_t)$

Set $\omega_{t+1} = \omega_t$

FOR $i = 1, 2, 3, ..., Size(A_t)$

$n_f(x_i, y_i) = n_f(x_i, y_i) + 1$

IF $n_f(x_i, y_i) = 2$

Set $D_{t+1} = D_{t+1} \cup \{(x_i, y_i)\}$, $S_{t+2} = S_{t+1} / D_{t+1}$

ELSE

Set $\omega_{t+1} = \min\{1, \frac{1/\sqrt{\lambda}}{\|w_{t+\frac{1}{2}}\|}\} w_{t+\frac{1}{2}}$

OUTPUT: ω_{t+1}

3 Result

First of all, we downloaded the Pegasos source code from http://www.cs.huji.ac.il/shais/code/index.html. There are three sample sets of training data and one test data set: One training set of size 2000, two other sets of musch smaller size and a test set of size 600. One interesting characteristic of this data set is that l_2 norm of every data is one. This puts all the data onto the surface of 2 spheres. We needed to look for more general data sets which do not exhibit such a characteristic. Thus, we use the training set and test sets given for cs229 homework #2. The size of training set is 2144 and test set has size of 800.

One of the interesting characteristics of the Pegasos is that we have a control on the size of A_t, which is k. If k = 1, then Pegasos is very similar to stochastic gradient method. If $k = |S|$,

then it results in a modified gradient-descent algorithm.

3.1 undo method

What the *undo* method actually dose in subgradient based algorithm is to prevent the ascent direction. Please see fig.(1) and (2). In the beginning of the iterations, the *undo* method seems to do much better than the pegasos. However, as the iteration reaches the point close to the convergence region, the pegasos does better than *undo* method. This result is analogs to the comparison between steepest descent algorithm and conjugate gradient method. Although the steepest descent may exhibit better performance in the beginning of iterations, the one that wins the performance is conjugate gradient method. The lesson here is that allowing ascent direction sometimes can be poison at that moment, but can be medicine. As described in algorithm description section, we combined the *undo*, *two-out*, and *probability* method and the result is that the *undo* method is dominant in choosing the search direction. In fig.(3) and (4), all the three methods (i.e. *undo* method, *undo-probability* combined method, and *undo-probability-two out* combined method exhibit almost identical performance).

3.2 two-out method

Since we've found that the *undo* method should not be used, we decided not to use it in both *two-out* method and *probability* method. Thus, in *two-out* method, although $f(\omega_{t+1}) > f(\omega_t)$ in some t, we keep the w_{t+1} as updated. Only thing we change is to update S every iteration by eliminating data whose flag is equal to two. The result is almost identical to the original pegasos method. (i.e. see the fig.(5) and (6). We strongly suspect that the reason for the almost identical result is that the training set we have (i.e. the size of about 2000) is too small to take an advantage of *two-out* method. To take an advantage of *two-out* method, we need enough iteration to get rid of outliers. The average number of data taken away from the training set until convergence is around 80. Among 80 data, there may be good data in it. It is because taking away the data whose number of flag is two is too cruel. Thus, we took away the data whose number of flag is four. Then, however, the number of data taken away from the training set is very small (i.e. zero, one or two until the convergence). The reason that we have hope in this method is that even with our training set, the *two-out* method exhibits slightly better performance than the pegasos right before the convergence occur (i.e. see fig.(7)). However, this is simply our hope and it need to be proven with bigger training set in the future.

3.3 probability method

As in case of the *two-out* method, the *probability* method exhibits almost identical result to pegasos (i.e. see fig.(8) and (9)). This also leaves us a hope that the *probability* method may 3 works better than pegasos when the training set size is much bigger.

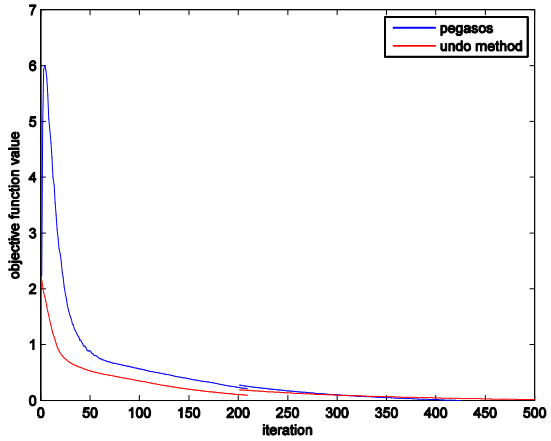

Figure1. Comparison between pegasos and *undo* method

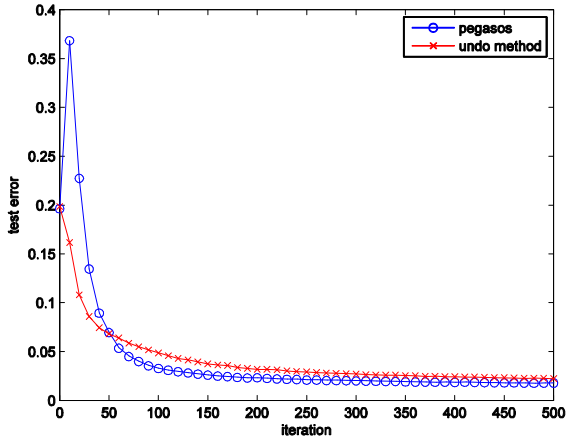

Figure2. Comparison between pegasos and *undo* method

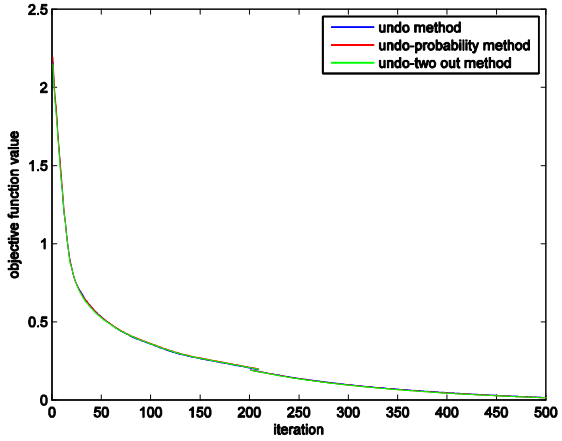

Figure3. *undo* method is dominant in choosing search direction

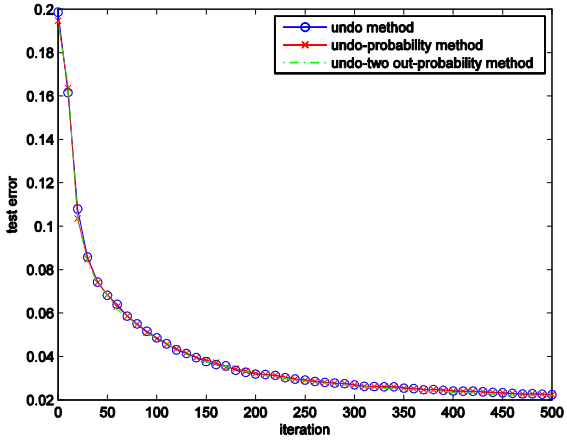

Figure4. *undo* method is dominant in choosing search direction

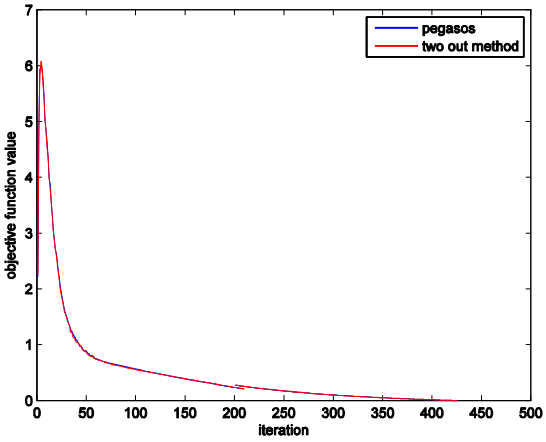

Figure5. comparison between pegasos and *two-out* method

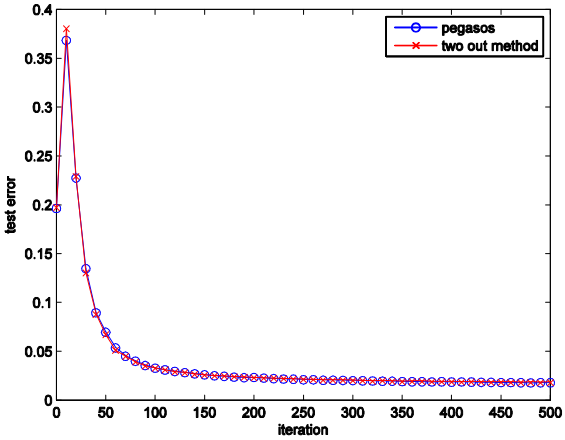

Figure6. comparison between pegasos and *two-out* method

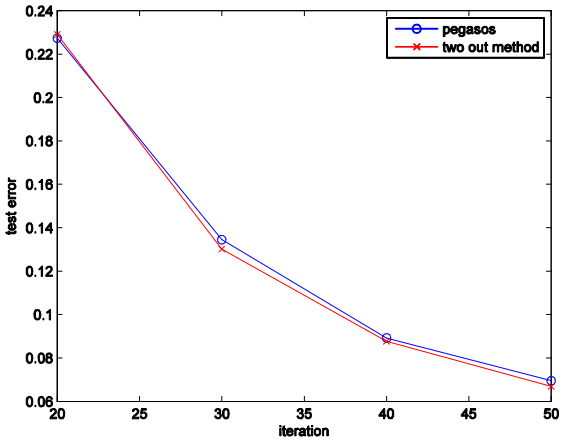

Figure7. zoomed in comparison between pegasos and *two-out* method

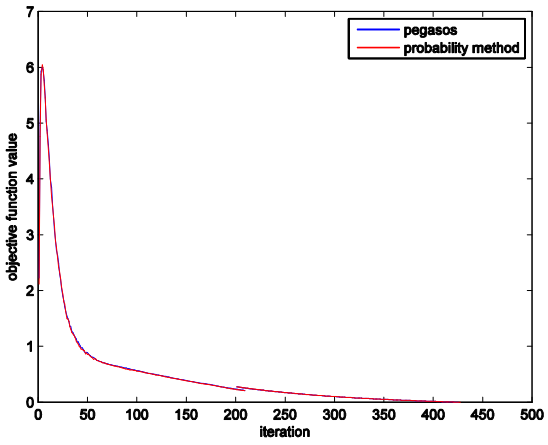

Figure8. comparison between pegasos and *probability* method

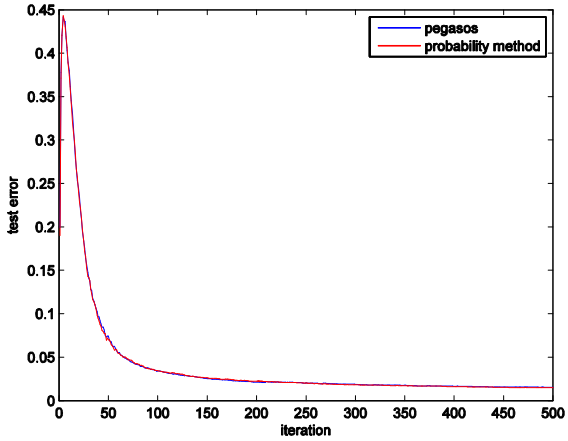

Figure9. comparison between pegasos and *probability* method

4 Conclusion

Developing a new algorithm is a very hard job. After reading many papers and getting very deep insight about the algorithm, then finally someone can develop what is so called 'noble algorithm.' Throughout the project, we have tried many other algorithms which have not been presented in this report. However the results were all pessimistic. Initially, for example, we tried to extract only support vectors from training set to increase the convergence rate. However, due to the presence of outliers, when we extract the support vectors (SVs), outliers were also extracted with SVs. This caused even worse test error than pegasos gave. In this report, we presented three different methods as a variant of Pegasos: *undo* method, *two-out* method and *probability* method. The *undo* method acted like steepest descent method. It may converge to a good solution, but in other times, it would not converge to a good solution. Both *two-out* method and *probability* method give a way of finding outliers in training set. Taking away data that gets two flags during the simulations, we may have high risk of giving away a good data. This result can be fixed either by increasing two to four or five. However to accomplish this, we need a bigger training set. The *probability* method also shows the potential to be used in bigger training set.

5 Reference

1. **Shalev-Shwartz, S., Singer, Y. and Srebro, N. (2007). *Pegasos: Primal Estimated sub-GrAdient Solver for SVM.* Proceedings of the 24th International Conference on Machine Learning.**
2. **Joachims, T. (1999). *Making Large-Scale SVM Learning Practical.* MIT Press Cambridge, MA, USA**
3. **Joachims, T. (2006). *Training Linear SVMs in Linear Time.* Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining**