Speeding up the solution of multilabel problems with Support Vector Machines

Fabio Aiolli
University of Pisa
Department of Computer Science
Via F. Buonarroti, 2
56127 Pisa, Italy
aiolli@di.unipi.it

Filippo Portera, Alessandro Sperduti
University of Padova
Department of Pure and Applied Mathematics
Via G. Belzoni, 7
35131 Padova, Italy
filo@math.unipd.it, sperduti@math.unipd.it

***Abstract*— The classical SVM approach to solve multilabel problems consists in training a single classifier for each class. We propose a compact model that considers the whole set of classifiers at once. Our strategy focuses on the shared use of the kernel matrix information between different classifiers in order to reduce the complexity of the learning task. Experiments with the Reuters-21578 corpus show a speedup in term of kernel computations (cache misses) preserving state-of-the-art performance.**

I. Introduction

A multilabel classification problem is an automated predicting task where a set of categories have to be assigned to a set of given patterns. Text categorization and bio-informatics are some of the domains in which multilabel problems arise. A machine learning approach tries to solve the problem by learning a classifier for each category given a labeled data set, and, subsequently, it makes use of the induced classifiers to predict the label set for new and unlabeled patterns. At present, statistical learning theory [1] and SVM constitutes a solid theoretical and empirical ground for the solution of pattern classification problem, especially for a binary setting.

For example, in [2], a set of independently trained binary SVM classifiers produces top performing results when applied to text categorization problems.

Considering that SVM algorithms complexity is tightly related to the number of kernel computations, we propose a learning model, based on margins, that combines all classifiers in a single optimization procedure, sharing the information of the kernel matrix between the different classifiers. Following this strategy, every classifier is built in parallel with each other obtaining a higher efficiency. We show that this method is mathematically equivalent to a learning approach that builds in a sequential manner each classifier. Thus, the efficacy of the proposed algorithm is not compromised.

II. Single classifiers vs. one Multilabel classifier

Let $S = \{(\bar{x}_1, c_1), .., (\bar{x}_m, c_m)\}$ be a set of m training examples, independently drawn from a set $X \times L$ and identically distributed, where $X \subseteq \Re^n$ and c_i is a subset of the set of integers $L = \{1, .., q\}$.

A multilabel classifier is a function $H : X \rightarrow 2^L$ that maps an instance $\bar{x}$ to a set of labels $c \in 2^L$. We consider a specific class of classifiers having the form:

$$H_M(\bar{x}) = \{r \in L : \bar{M}_r \cdot \bar{x} \geq 0\}$$

where $\bar{M}_r$ is the r-th row of a $q \times n$ matrix M. Given a classifier $H_M(\bar{x})$ and an example $(\bar{x}_i, c_i)$ we say that $H_M(\cdot)$ misclassifies $\bar{x}_i$ if $H_M(\bar{x}_i) \neq c_i$. However each example can have several labels, so we count as errors each misclassified label. Consequently the empirical error for a multilabel classifier is defined as the expected number of errors on the training set:

$$\epsilon_S(M) = \frac{1}{mq} \sum_{i=1}^{m} |H_M(\bar{x}_i) \triangle c_i|$$

where the operator $\triangle$ is the symmetric difference of sets operator.

This functional is hard to manipulate directly since $\epsilon_S(M)$ is not continuous. Therefore, let $y_i^r = 1$ if $r \in c_i$, -1 otherwise, we replace the error function $|H_M(\bar{x}_i) \triangle c_i|$ with a piecewise linear bound:

$$\epsilon_i(M) = |H_M(\bar{x}_i) \triangle c_i| \leq \sum_{r=1}^{q} [1 - y_i^r(\bar{M}_r \cdot \bar{x}_i)]_+$$

where $[v]_+$ is the hinge loss that is equal to v if $v \geq 0$ and 0 otherwise.

Summing over all the examples in S we get an upper bound for the empirical loss:

$$\epsilon_S(M) \leq \frac{1}{mq} \sum_{i=1}^{m} \sum_{r=1}^{q} [1 - y_i^r(\bar{M}_r \cdot \bar{x}_i)]_+. \tag{1}$$

Notice that this loss is always greater or equal to 0. We say that a sample S is *linearly separable* by a multilabel machine if there exists a matrix M such that the loss in eq. (1) is equal to 0, that is:

$$\forall i, \;\; \sum_{r=1}^{q} [1 - y_i^r(\bar{M}_r \cdot \bar{x}_i)]_+ = 0. \tag{2}$$

In previous use of SVM for multilabel classification problems (e.g. [2] for text categorization), a set of q linear classifiers is independently trained. The training process is based on the solution of the following problems, $\forall r \in [1..q]$:

$$\begin{array}{l} \min_{\bar{M}_r} \frac{1}{2}||\bar{M}_r||^2 + C_+ \sum_{i|y_i^r=1} \xi_{i,r} + C_- \sum_{i|y_i^r\neq 1} \xi_{i,r} \\ \text{subject to: } \forall i,\ y_i^r(\bar{M}_r \cdot \bar{x}_i) \geq 1 - \xi_{i,r} \end{array} \tag{3}$$

where C_+ and C_- are two scalar constants that weight the trade off between the machine complexity and the empirical error and they are associated to positive and negative examples in order to take care of unbalanced data sets [3].

Considering that kernel matrix computation plays an important role in SVM algorithms, our proposal investigates on a more efficient use of it.

Since the constraints and the objective functions of (3) are all independent, the set of q problems can be written as a single model. In fact, the solution of (3) is equivalent to the solution of:

$$\begin{array}{l} \min_M \frac{1}{2}||M||^2 + C_+ \sum_{i,r|y_i^r=1} \xi_{i,r} + C_- \sum_{i,r|y_i^r\neq 1} \xi_{i,r} \\ \text{subject to: } \forall i,r\ y_i^r(\bar{M}_r \cdot \bar{x}_i) \geq 1 - \xi_{i,r} \end{array} \tag{4}$$

It can be noticed that the resulting objective function is still quadratic and convex. Introducing a kernel function $K_{i,j} \equiv K(\bar{x}, \bar{y})$, the dual formulation of (4) can be written as:

$$\begin{array}{l} \max_\alpha \sum_{i,r} \alpha_{i,r} - \frac{1}{2} \sum_{i,j,r} \alpha_i^r y_i^r \alpha_j^r y_j^r K_{i,j} \\ \quad \text{subject to:} \\ \quad \forall i,r\ y_i^r = 1 \quad 0 \leq \alpha_{i,r} \leq C_+ \\ \quad \forall i,r\ y_i^r \neq 1 \quad 0 \leq \alpha_{i,r} \leq C_- \end{array} \tag{5}$$

In literature similar formulations to solve multiclass learning problems have been proposed [4], [5]. Given a pattern $\bar{x}_p$, those methods predict the unique associated class r by selecting the classifier that attains the maximum score on pattern $\bar{x}_p$. However in a multilabel task, where a set of predicted labels have to be identified for a given example, those constraints are not sufficient to discriminate positive labels from negative labels and thus these formulations cannot be directly applied. Thus, to overcome this problem, we adopted the standard SVM rule, i.e. an absolute threshold constraint centered in the origin.

The bias term $\bar{b}$ doesn't appear explicitly in our model. However, using a kernel $K(\bar{x}, \bar{y}) = \bar{x} \cdot \bar{y} + 1$ is equivalent to have a classifier $\bar{M}_r \cdot \bar{x} + b_r$ with the introduction of the additional term $\sum_r b_r^2$ in the objective function, i.e. the addition of a regularization term on the bias. Actually, this solution may not give optimal generalization results, since even if in [6] no difference on the generalization performance is observed, in experiments (that we do not report here), where we explicitly used the bias terms without regularization on them, we obtained slightly better generalization performance. In this case, however, the optimization procedure could not be carried on with the selection of a single pattern due to an additional constraint over the whole training set that emerges because of the not regularized bias terms. Therefore another technique that selects two randomly chosen patterns for local optimization has been used. The resulting algorithm is significantly slower than the one reported here.

A limit of our approach is the use of the same kernel function for all the classifiers. If each classifier have to be trained with a distinct kernel, the shared use of the kernel matrix is not feasible. This is also the case when distinct kernel hyper-parameters are chosen.

On the other side, training classifiers for different classes with different values of C is not a problem, since a generalization of problem (5) can consider a refined objective function where we introduce a C_r for each class:

$$\begin{array}{l} \max_\alpha \sum_{i,r} \alpha_{i,r} - \frac{1}{2} \sum_{i,j,r} \alpha_i^r y_i^r \alpha_j^r y_j^r K_{i,j} \\ \quad \text{subject to:} \\ \quad \forall i,r\ y_i^r = 1 \quad 0 \leq \alpha_{i,r} \leq C_r^+ \\ \quad \forall i,r\ y_i^r \neq 1 \quad 0 \leq \alpha_{i,r} \leq C_r^- \end{array} \tag{6}$$

We decided to consider problem (5) because of its simplicity. In fact, the calibration of problem (6) would have required a larger amount of time.

In the following we give an optimization procedure for problem 5 that can be interpreted as a parallel adaptation of the classical single-class SVM.

III. Optimization

The dual formulation (5) is a quadratic optimization problem with independent constraints on each variable $\alpha_{i,r}$. Therefore we can apply a pattern selection strategy and, then, optimize the subproblem associated with the selected pattern. The global optimization procedure is described in Figure 2.

We first give a description of the pattern optimization procedure.

1) Pattern Optimization: Given a pattern p, we first show how to analytically solve the associated problem with respect to an update involving a single variable $\alpha_p^{\hat{r}}$. In this case we have:

$$\alpha_p^{\hat{r}\prime} \leftarrow \alpha_p^{\hat{r}} + \theta\ .$$

We now consider how this update affects the objective function of 5:

$$\begin{array}{rcl} Q(\bar{\alpha}_p + \theta\bar{\delta}) & = & Q(\bar{\alpha}) + \theta(1 - y_p^{\hat{r}} \sum_i \alpha_i^{\hat{r}} y_i^{\hat{r}} K_{i,p}) + \\ & & -\frac{1}{2}\theta^2 K_{p,p} \end{array} \tag{7}$$

where $\bar{\delta}$ is a vector with all components equal to 0 but component $\hat{r}$ that is equal to 1. Thus, defining $s_r(\bar{x}_i) = \sum_j \alpha_j^r y_j^r K_{j,i}$ as the score for classifier r on pattern $\bar{x}_i$, an update θ will change the value of the Lagrangian of the following amount:

$$D_p^{\hat{r}}(\theta_r) = \theta(1 - y_p^{\hat{r}} s_{\hat{r}}(\bar{x}_p)) - \frac{1}{2}\theta^2 K_{p,p}. \tag{8}$$

The above equation is concave on θ, so the maximum gain for the value of the Lagrangian is obtained when the first derivative of $D_p^{\hat{r}}(\theta)$ is equal to zero, that is for:

$$\theta = \frac{1 - y_p^{\hat{r}} s_{\hat{r}}(\bar{x}_p)}{K_{p,p}} .$$

If the value of the updated $\alpha_p^{\hat{r}}$ is not feasible for the constraint in (5) we clip the value of θ. The variable is updated if and only if the related Lagrangian contribute is positive.

The algorithm given above may be considered incremental in the sense that a solution that has been found for a given pattern forms the initial condition for the next pattern optimization.

In Figure 1 the pseudo-code of the proposed algorithm is presented. At each iteration, the algorithm applies the basic step to the m variables associated with the pattern which has been chosen for optimization, then only if necessary, the optimization is done over all the m variables associated with the selected pattern, until the relative Lagrangian improvement is below the input threshold ϕ_V. It must be said that for each iteration the scores for every pattern in the training set must be updated before being used for the selection phase, which is described in the following section.

OptimizeOnPattern($\bar{x}_p$,φ_V)

$t := 0$, $V(0) := 0$. $\forall r$, $\alpha'_r := \alpha_p^r$, $s_r := s_r(\bar{x}_p)$
do
  $t := t + 1$, $V(t) := 0$
  for each $\hat{r} \in L$
    if ($y_i^{\hat{r}} = 1$) **then** $C := C_+$ **else** $C := C_-$
    $\theta := \frac{1-(y_p^{\hat{r}} s_{\hat{r}})}{K_{pp}}$
    if ($\alpha'_{\hat{r}} + \theta < 0$) **then** $\theta := -\alpha'_{\hat{r}}$
    if ($\alpha'_{\hat{r}} + \theta > C$) **then** $\theta := C - \alpha'_{\hat{r}}$
    $V(t) := V(t) + \theta\left(1 - y_p^{\hat{r}} s_{\hat{r}}(\bar{x}_p) - \frac{1}{2}\theta K_{p,p}\right)$
    $\alpha'_{\hat{r}} := \alpha'_{\hat{r}} + \theta$;
    $s_{\hat{r}} := s_{\hat{r}} + y_p^{\hat{r}} \theta K_{pp}$;
until ($\frac{V(t)-V(t-1)}{V(t)} \leq \varphi_V$)
return $\{\bar{\alpha}'\}$

Fig. 1. Algorithm for the optimization of the variables associated with a given pattern.

OptimizeAMG(**S**, φ_V)

// $m = |S|$, $q = |L|$
for $i := 1$ **to** m
  for $r := 1$ **to** q
    $\alpha_{i,r} := 0$
do
  $\rho := \frac{\sum_i \text{AMGValue}(\bar{x}_i)}{m}$
  for $p := 1$ **to** m {
    amg := AMGValue($\bar{x}_p$)
    if (amg $\geq \rho$) **then**
      OptimizeOnPattern($\bar{x}_p$, φ_V)
  }
  $P := \frac{1}{2}||M||^2 + C\sum_{i,r|y_i^r=1} \xi_{i,r}$
  $L := \sum_{i,r} \alpha_{i,r} - \frac{1}{2}\sum_{i,j,r} \alpha_i^r y_i^r \alpha_j^r y_j^r K_{i,j}$
until $\frac{P-L}{P} < \zeta$

Fig. 2. Optimization algorithm for the solution of the training problem (5).

2) Selection criteria: In order to decide which pattern has to be selected for optimization we evaluated a procedure called Approximate Maximum Gain [7] (here denoted with AMG); in this case the value is computed as $V_p = \sum_r [D_r^p]_+$, from eq. (8). This value is a lower bound of the actual increment in the Lagrangian obtained when the pattern p is selected for optimization. In fact, when optimizing on a single pattern the procedure iterates several times on the same or different pairs of indexes. Furthermore, it is worthwhile to note that the $AMG(p)$ condition represent a well founded measure of the distance from optimality of the variables associated with the pattern p.

At the beginning of each iteration, a threshold ρ is computed. For each example of the training set the AMG criterion is applied to it and the example will be selected for optimization if the value returned is greater than the threshold. The threshold ρ was set using $\rho = \text{mean}_p(V_p)$.

The algorithm ends whenever the relative gap between the value of the objective function of the primal problem (P) and the value of the objective function of the Lagrangian problem (L) is below a threshold ζ, that is, when:

$$\frac{P - L}{P} < \zeta$$

In our experiments we decided to set $\zeta = 0.01$.

IV. Experiments

In this section we describe the experiments that have been conducted in order to measure the efficiency and the generalization ability of our algorithm.

We performed experiments on the "Reuters-21578, Dist. 1.0" corpus ("ModApte" split). We selected three subset of categories. The first is made of the 10 most frequent categories, the second is made of the 90 categories that have at least one document in the training set and one document in the test set; the third is made of 115 categories that are present at least once in training documents. We used the same preprocessing as in [8].

We have measured effectiveness in terms of precision π_r wrt class r and recall ρ_r wrt class r defined in the usual way. We consider both micro and macro averaged values of precision and recall as defined in [8]. We combined the precision and recall values obtained with the F_1 measure, defined as:

$$F_1 = \frac{2\pi\rho}{\pi + \rho}.$$

On each experiment, we compare the results obtained with the sequential application of the single category based SVM and the results obtained with our algorithm.

We experimented our algorithm with a polynomial kernel of degree 1, $K(\bar{x}, \bar{y}) = \bar{x} \cdot \bar{y} + 1$ in order to add a regularized bias, moreover the values of the regularization constants $C_+ = 0.5$, $C_- = 1$ have been selected with a 5-fold cross validation on the training set on the 10 most frequent categories.

Problem	*micro $F1$*	*macro $F1$*	*Speedup*	*# Kernel cache misses ($\times m^{-1}$)*
q=10, S-OPT (Sζ)	0.9467	0.8946	1	13574
q=10, S-OPT (Pζ)	0.9469	0.8957	1.0361	13573
q=10, P-OPT	0.9467	0.8952	2.9906	3758
q=90, S-OPT (Sζ)	0.8775	0.5830	1	55554
q=90, S-OPT (Pζ)	0.8776	0.5833	1.0444	55173
q=90, P-OPT	0.8778	0.5860	2.9283	17064
q=115, S-OPT (Sζ)	0.8664	0.6932	1	65506
q=115, S-OPT (Pζ)	0.8666	0.6945	1.1089	64845
q=115, P-OPT	0.8672	0.6963	2.0678	35747

TABLE I

AVERAGE MICRO- AND RELATED MACRO-AVERAGED $F1$, SPEEDUP FACTOR AND THE NUMBER OF KERNEL CACHE MISSES.

We evaluated the set of independently trained single label classifiers (we called this algorithm S-OPT) and our algorithm (called P-OPT). Each independently trained classifier has been computed using the proposed algorithm on a single class. The results obtained on the three datasets are shown in Table I, where we have reported both the performance at convergence, the speedup in terms of CPU time spent for training, and the number of kernel cache misses, i.e. the number of times that a row of the kernel matrix has to be computed because it is not found in the cache.

The initial comparison framework for the two algorithms established that the value of ζ, that determines the termination condition of each one of the q computation of S-OPT and of the computation of P-OPT, to be the same (ζ = 0.01). The results obtained with this rule are presented in Table I in row Sζ.

We noticed that when P-OPT terminates, the gaps calculated on the subproblems related to a single class can be above and below the value of ζ, while, in S-OPT, we require that each solver terminates whenever its relative gap is below ζ. Therefore, the solutions of S-OPT and P-OPT obtained with this rule could be substantially different. We adopted the following method in order to align the termination conditions of the two algorithms. We first apply the algorithm P-OPT obtaining a solution with the desired convergence gap. Then, given a class r, we fix the gap threshold that must be achieved by S-OPT on problem r to terminate the computation to be below the last gap computed by P-OPT on the r-th subproblem of (4). The results obtained with this rule are shown in Table I in row Pζ.

It is clear from Table I that the time efficiency of a "parallel" training of the classifiers is due to the reduction of the number of kernel evaluations. With our algorithm, different classes can reuse kernel information that has already been computed, while in the classical approach this is not the case. The generalization ability is not compromised and we observed that, on the 115 classes problem, our training algorithm obtains a micro $F1$ performance comparable to state-of-the-art results and the macro $F1$ performance is among the best reported.

Results show that the generalization performance for P-OPT and S-OPT (Pζ) are slightly different. In fact, P-OPT and S-OPT (Pζ) follows different trajectories because the AMG criterion behaves in different ways for each one of the two approaches. In P-OPT the sum of each AMG associated for each class is considered, while in S-OPT, the AMG is computed on a single class. Moreover, the S-OPT (Pζ) algorithm ends whenever the duality gap is greater than the fixed threshold, while in P-OPT these values are exactly reached.

V. CONCLUSIONS

We proposed a compact model for multilabel classification problems and an optimization algorithm which allows the simultaneous training of the classifiers. It should be noted that related approaches for multiclass single label classification problems [4], [5] cannot be directly applied to multilabel classification problems. When applied to the Reuters-21578 dataset our approach speeds up the training by a factor of around 2 while maintaining the same generalization performance. Further investigations will consider different values of ζ, the application of our algorithm to other datasets, and a faster selection criterion for the solver. A disadvantage of the proposed approach is the use of the same kernel function for all the classifiers, i.e. the use of distinct kernel functions or different values of kernel hyper-parameters is not achievable with this methodology. On the other side it is very easy to extend the proposed model in order to use a different regularization parameter for each class.

REFERENCES

[1] V. N. Vapnik, *Statistical Learning Theory*. Wiley, New York, 1998.

[2] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *Proceedings of ECML-98, 10th European Conference on Machine Learning*, 1998, pp. 137–142.

[3] K. Morik, P. Brockhausen, and T. Joachims, "Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring," in *Proc. 16th ICML-99*, 1999, pp. 268–277. [Online]. Available: citeseer.nj.nec.com/morik99combining.html

[4] J. Weston and C. Watkins, "Support vector machines for multi-class pattern recognition," in *Proceedings of the Seventh European Symposium On Artificial Neural Networks*, 4 1999. [Online]. Available: citeseer.nj.nec.com/article/weston99support.html

[5] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *Journal of Machine Learning Research*, vol. 2, pp. 265–292, 2001.

[6] C. Hsu and C. Lin, "A comparison of methods for multi-class support vector machines," *IEEE Transactions on Neural Networks*, no. 13, pp. 415–425, 2002. [Online]. Available: citeseer.nj.nec.com/hsu01comparison.html

[7] F. Aiolli and A. Sperduti, "An efficient smo-like algorithm for multiclass svm," in *Proc. NNSP 2002*, Martigny, Switzerland, September 2002.

[8] S. F. Debole, F., "Supervised term weighting for automated text categorization," in *Proc. SAC-03*, March 2003.