Learning Sparse SVM for Feature Selection on Very High Dimensional Datasets

Mingkui Tan † TANMINGKUI@GMAIL.COM
Li Wang †‡ LIW022@UCSD.EDU
Ivor W. Tsang † IVORTSANG@NTU.EDU.SG
† School of Computer Engineering, Nanyang Technological University, Singapore
‡ Department of Mathematics, University of California, San Diego, USA

Abstract

A sparse representation of Support Vector Machines (SVMs) with respect to input features is desirable for many applications. In this paper, by introducing a 0-1 control variable to each input feature, l_0-norm Sparse SVM (SSVM) is converted to a mixed integer programming (MIP) problem. Rather than directly solving this MIP, we propose an efficient cutting plane algorithm combining with multiple kernel learning to solve its convex relaxation. A global convergence proof for our method is also presented. Comprehensive experimental results on one synthetic and 10 real world datasets show that our proposed method can obtain better or competitive performance compared with existing SVM-based feature selection methods in term of sparsity and generalization performance. Moreover, our proposed method can effectively handle large-scale and extremely high dimensional problems.

1. Introduction

In many machine learning applications, there is a great desire of sparsity with respect to input features. Several facts account for this. Firstly, many real datasets such as texts and Microarray data are represented as very high dimensional vectors, resulting in great challenges for further processing. Secondly, most features in high dimensional vectors are usually non-informative or noisy and may seriously affect the generalization performance. Thirdly, a sparse classifier can lead to a simplified decision rule for faster prediction in large-scale problems. Finally, in some applications like Microarray data analysis, a small set of features is desirable to interpret the results.

Recently, numerous feature selection methods regarding Support Vector Machine (SVM) have been proposed (Blum & Langley, 1997; Guyon & Elisseeff, 2003). To obtain a sparse decision rule for SVM, many researchers (Bradley & Mangasarian, 1998; Zhu et al., 2003; Fung & Mangasarian, 2004) proposed l_1-norm Sparse SVM (SSVM), which uses $\|w\|_1$ as the regularizer. The resultant problem is convex, and can be solved optimally by Linear Programming (LP) solvers or Newton method (Fung & Mangasarian, 2004). Apart from l_1-norm SSVM, (Weston et al., 2003) proposed an Approximation of the zero norm Minimization (AROM) to solve SSVM with $\|w\|_0$ as the regularizer, namely l_0-norm SSVM. However, the resultant optimization is non-convex and may suffer from local minima. Recently, (Chan et al., 2007) proposed another two direct convex relaxations to l_0-norm SSVM, namely QCQP-SSVM and SDP-SSVM, which can be solved by Quadratically Constrained Quadratic Programming (QCQP) and Semi-Definite Programming (SDP), respectively. Though both the relaxed optimization problems are convex, they are computationally expensive, especially for high dimensional problems.

Besides sparse regularization, (Guyon et al., 2002) proposed an effective Recursive Feature Elimination (RFE) scheme for feature selection. SVM-RFE can obtain nested subsets of input features and has shown state-of-the-art performance on gene selection in Microarray data analysis (Guyon et al., 2002). However, as described by (Xu et al., 2009), the nested "monotonic" feature selection scheme may be suboptimal in identifying the most informative subset of input features. Here, the "monotonic" property refers to the problem that, if an informative feature is wrongly eliminated from a subset S, it will not be in its nested subsets (Xu et al., 2009). This issue becomes extremely critical when dealing with problems with large number of noise features and therefore an accurate SVM model is hard to be obtained to rightly measure the importance of features. To overcome this problem, (Xu et al., 2009) proposed a non-monotonic feature selection method, namely NMMKL. However, their method requires to solve a QCQP problem with $|S|$ quadratic constraints, where $|S|$ denotes

Appearing in *Proceedings of the 27^{th} International Conference on Machine Learning*, Haifa, Israel, 2010.

the number of input features. Hence, NMMKL is computationally infeasible for high dimensional problems.

In this paper, we propose to learn a sparse solution with respect to input features to SVM, namely Feature Generating Machine (FGM). It iteratively generates a pool of violated sparse feature subsets and then combines them via efficient Multiple Kernel Learning (MKL) algorithm. FGM shows great scalability to non-monotonic feature selection on large-scale and very high dimensional datasets. We also provide a proof of global convergence for FGM. The rest of this paper is organized as follows. Section 2 describes the sparse SVM problem and our proposed cutting plane algorithm. Experimental results are presented in Section 3. The last section gives the conclusive remarks.

2. Learning Sparse SVM

In the sequel, we denote the transpose of vector/matrix by the superscript $'$ and a vector with all entries equal to one as $\mathbf{1} \in R^n$. We also denote $\|v\|_p$ as the l_p-norm of a vector v. Moreover, $A \odot B$ represents the elementwise product between two matrices A and B.

Given a set of labeled patterns $\{x_i, y_i\}_{i=1}^n$, where $x_i \in R^m$ is the input and $y_i \in \{\pm 1\}$ is the output label, we learn a linear decision hyperplane $f(x) = w'x$ that minimizes the following structural risk functional: $\min_w \Omega(\|w\|_p) + C\sum_{i=1}^n \ell(-y_i w' x_i)$, where $w = [w_1, \ldots, w_m]' \in R^m$ is the weight vector of the decision hyperplane, $\Omega(\|w\|_p)$ is the regularizer that defines the characteristic (e.g. sparsity) of the decision hyperplane, $\ell(\cdot)$ is a convex loss function, and $C > 0$ is a regularization parameter that trades off the model complexity and the fitness of the decision hyperplane. For standard SVMs, $\Omega(\|w\|_p)$ is set to $\frac{1}{2}\|w\|_2^2$, which is a non-sparse regularizer. Hence, the learned decision hyperplane is usually non-sparse.

In order to obtain a sparse solution of SVM, we firstly introduce a feature selection vector $d = [d_1, \ldots, d_m]' \in \mathcal{D}$ which controls the sparsity of the SVM decision hyperplane: $f(x) = w'x = (\tilde{w} \odot d)'x = \tilde{w}'(d \odot x)$, where $\mathcal{D} = \{d | \sum_{j=1}^m d_j \le B,\ d_j \in \{0,1\},\ j = 1, \cdots, m\}$ is the domain of d, and B controls the sparsity of d. For simplicity, we here focus on square hinge loss[1], and the positive constraint $\xi_i \ge 0$ satisfies automatically and can be omitted. Then, the objective of Sparse SVM (SSVM) can be simplified as:

$$\min_{d\in\mathcal{D}} \min_{\tilde{w},\xi,\rho} \quad \frac{1}{2}\|\tilde{w}\|_2^2 + \frac{C}{2}\sum_{i=1}^n \xi_i^2 - \rho \qquad (1)$$
$$s.t. \quad y_i \tilde{w}'(x_i \odot d) \ge \rho - \xi_i,\ \ i = 1, \cdots, n.$$

The inner minimization problem can be solved by its dual, then (1) can be rewritten as follows:

$$\min_{d\in\mathcal{D}} \max_{\alpha\in\mathcal{A}} -\frac{1}{2}\left\|\sum_{i=1}^n \alpha_i y_i (x_i \odot d)\right\|^2 - \frac{1}{2C}\alpha'\alpha, \qquad (2)$$

[1]This loss function can be solved efficiently by LIBLinear.

where $\alpha = [\alpha_1, \ldots, \alpha_n]'$ is a vector of dual variables for the inequality constraints in the inner minimization problem (1), and $\mathcal{A} = \{\alpha | \sum_{i=1}^n \alpha_i = 1, \alpha_i \ge 0, i = 1, \cdots, n\}$ is the domain of α.

2.1. Convex Relaxation

Observe that (2) is still a mixed integer programming (MIP) problem, which is computationally expensive in general. Following (Li et al., 2009b;a), we introduce a mild convex relaxation for our SSVM formulation in (2). According to the minimax inequality (Kim & Boyd, 2008), when we interchange the order of $\min_{d\in\mathcal{D}}$ and $\max_{\alpha\in\mathcal{A}}$ in (2), then the saddle-point problem (2) can be lower-bounded by

$$\max_{\alpha\in\mathcal{A}} \min_{d\in\mathcal{D}} -\frac{1}{2}\left\|\sum_{i=1}^n \alpha_i y_i (x_i \odot d)\right\|^2 - \frac{1}{2C}\alpha'\alpha. \qquad (3)$$

Define $S(\alpha, d) = -\frac{1}{2}\left\|\sum_{i=1}^n \alpha_i y_i (x_i \odot d)\right\|^2 - \frac{1}{2C}\alpha'\alpha$ and bring in an additional variable $\theta \in R$, (3) becomes:

$$\max_{\alpha\in\mathcal{A},\theta} -\theta \quad : \quad \theta \ge -S(\alpha, d^t),\ \ \forall\, d^t \in \mathcal{D}, \qquad (4)$$

which is a convex QCQP problem. Let $\mu_t \ge 0$ be the dual variable for each constraint. Its Lagrangian is written as: $\mathcal{L}(\theta, \mu) = -\theta + \sum_{t, d^t\in\mathcal{D}} \mu_t(\theta + S(\alpha, d^t))$. Setting its derivative w.r.t. θ to zero, we have $\sum \mu_t = 1$. Let μ be the vector of μ_t's, and $\mathcal{M} = \{\mu | \sum \mu_t = 1, \mu_t \ge 0\}$ be the domain of μ. The Lagrangian $\mathcal{L}(\theta, \mu)$ can be rewritten as:

$$\max_{\alpha\in\mathcal{A}} \min_{\mu\in\mathcal{M}} \sum_{d^t\in\mathcal{D}} \mu_t S(\alpha, d^t) \qquad (5)$$
$$= \min_{\mu\in\mathcal{M}} \max_{\alpha\in\mathcal{A}} -\frac{1}{2}(\alpha \odot y)'\Big(\sum_{d^t\in\mathcal{D}} \mu_t X_t X_t^{'} + \frac{1}{C} I\Big)(\alpha \odot y),$$

where $X_t = [x_1 \odot d^t, \cdots, x_n \odot d^t]'$, and the last equality holds due to the fact that the objective function is concave in α and convex in μ. Moreover, (5) can be regarded as a MKL problem (Rakotomamonjy et al., 2008), where the kernel matrix $\sum_{t, d^t\in\mathcal{D}} \mu_t X_t X_t^{'}$ to be learned is a convex combination of $|\mathcal{D}|$ base kernel matrices $X_t X_t'$, each of which is constructed from a feasible feature selection vector $d^t \in \mathcal{D}$.

2.2. Cutting Plane Algorithm

Although (4) is convex, a huge number of base kernels make it impractical to be solved by existing MKL techniques. Fortunately, not all constraints in (4) are active at optimality. Alternatively, we can efficiently solve this problem by cutting plane algorithm (Kelley, 1960), which iteratively generates a pool of sparse feature subsets to construct the quadratic inequality constraints in (4).

The overall algorithm of FGM is described in Algorithm 1. Denote the subset of constraints by $\mathcal{C} \subset \mathcal{D}$. First, we initialize the vector of dual variables α to $\frac{1}{n}\mathbf{1}$ and find the most violated $\hat{d} \in \mathcal{D}$, initialize the working set $\mathcal{C} = \{\hat{d}\}$. Since

the number of d in $\mathcal{C}$ (i.e. the number of base kernel matrices) is no longer large, one can perform MKL with a subset of kernel matrices in $\mathcal{C}$ and obtain new α from (5). Then the most violated d is obtained and is added to $\mathcal{C}$, which is known as the "feature generation". The whole process is repeated until the termination criterion is met.

Algorithm 1 The cutting plane algorithm for FGM.

1: Initialize $\alpha = \frac{1}{n}\mathbf{1}$. Find the most violated $\hat{d}$, and set $\mathcal{C} = \{\hat{d}\}$.
2: Run MKL for the subset of kernel matrices selected in $\mathcal{C}$ and obtain α and μ from (5).
3: Find the most violated $\hat{d}$ and set $\mathcal{C} = \mathcal{C} \bigcup \hat{d}$.
4: Repeat step 2-3 until convergence.

2.3. MKL with a Subset of Kernel Matrices

Several efficient MKL approaches have been proposed in recent years. For simplicity, in this paper we apply SimpleMKL (Rakotomamonjy et al., 2008) to solve the MKL problem defined on the subset of kernel matrices selected in $\mathcal{C}$. More specifically, suppose that the current working set is $\mathcal{C} = \{d^1, \cdots, d^T\}$, the MKL problem in (5) thus corresponding to the following primal optimization problem:

$$\begin{aligned} &\min_{\mu\in\mathcal{M},\tilde{w},\rho,\xi} \ \frac{1}{2}\sum_{t=1}^{T}\frac{1}{\mu_t}\|\tilde{w}_t\|^2 + \frac{C}{2}\sum_{i=1}^{n}\xi_i^2 - \rho \\ &s.t. \sum_{t=1}^{T}\tilde{w}_t'(y_i x_i \odot d^t) \geq \rho - \xi_i, \forall i = 1, \cdots, n. \end{aligned} \tag{6}$$

Following SimpleMKL, we solve (5) (or, equivalently, (6)) iteratively. First, we fix the coefficients μ of the base kernel matrices and solve the dual of SVM: $\max_{\alpha\in\mathcal{A}} -\frac{1}{2}(\alpha \odot y)'\left(\sum_{t=1}^{T}\mu_t X_t X_t' + \frac{1}{C}I\right)(\alpha \odot y)$. Then, we fix α and use the reduced gradient method for updating μ. These two steps are iterated until convergence.

2.4. Finding the Most Violated $\hat{d}$

To find the most violated $\hat{d}$ in (4), we have to solve the following equivalent optimization problem: $\max_{d\in\mathcal{D}} \frac{1}{2}\left\|\sum_{i=1}^{n}\alpha_i y_i (x_i \odot d)\right\|^2 = \frac{1}{2}\sum_{j=1}^{m} c_j^2 d_j$ with $c_j = \sum_{i=1}^{n}\alpha_i y_i x_{ij}$. This problem is a linear integer programming subject to one linear constraint $\sum_{j=1}^{m} d_i \leq B$. The globally optimal solution of this problem can be obtained without any numeric optimization solver. That is, it can be solved by first sorting c_j^2's and then setting the first B numbers corresponding to d_j to 1 and the rests to 0.

2.5. Prediction

When the algorithm converges, we get α^* and μ^*, the decision function can be obtained by $f(x) = \sum_{t=1}^{T}\mu_t\sum_{i=1}^{n}\alpha_i y_i (x_i \odot d^t)' x = \sum_{i=1}^{n}\alpha_i y_i (x_i \odot \tilde{d})' x$, where $\tilde{d} = \sum_{t=1}^{T}\mu_t d^t$.

2.6. Global Convergence

In this subsection, we consider the convergence properties of FGM. Let $\mathcal{A} \times \mathcal{D}$ be the constraint domain for problem (4), where $\mathcal{A} = \left\{\alpha \middle| \sum_{i=1}^{n}\alpha_i = 1, \alpha_i \geq 0, i = 1, \cdots, n\right\}$ and $\mathcal{D} = \left\{d \middle| \sum_{j=1}^{m} d_j \leq B,\ d_j \in \{0,1\},\ j = 1, \cdots, m\right\}$. In FGM, we iteratively find and add the most violated constraint to the set $\mathcal{C}$, which is a subset of $\mathcal{D}$, i.e, $\mathcal{C} \subseteq \mathcal{D}$. Further denote by $\mathcal{C}_k$ be the constraint set in kth iteration, then we have $\mathcal{C}_k \subseteq \mathcal{C}_{k+1}$. In kth iteration, we find a new constraint d^{k+1} based on α_k, i.e., $-S(\alpha_k, d^{k+1}) = \max_{d\in\mathcal{D}} -S(\alpha_k, d)$. Define

$$\beta_k = \max_{1\leq i\leq k} -S(\alpha_k, d^i) = \min_{\alpha\in\mathcal{A}}\max_{1\leq i\leq k} -S(\alpha, d^i) \tag{7}$$

and

$$\varphi_k = \min_{1\leq j\leq k} -S(\alpha_j, d^{j+1}), \tag{8}$$

where $-S(\alpha_j, d^{j+1}) = \max_{d\in\mathcal{D}} -S(\alpha_j, d)$. Similar to (Chen & Ye, 2008), we have the following theorem that indicates FGM gradually approaches to the optimal solution.

Theorem 1. *Let (α^*, θ^*) be the globally optimal solution pair of (4), $\{\beta_k\}$ and $\{\varphi_k\}$ as defined above, then:*

$$\beta_k \leq \theta^* \leq \varphi_k. \tag{9}$$

With the number of iteration k increasing, $\{\beta_k\}$ is monotonically increasing and the sequence $\{\varphi_k\}$ is monotonically decreasing.

Proof. $\theta^* = \min_{\alpha\in\mathcal{A}}\max_{d\in\mathcal{D}} -S(\alpha, d)$. For a fixed feasible α, we have $\max_{d\in\mathcal{C}_k} -S(\alpha, d) \leq \max_{d\in\mathcal{D}} -S(\alpha, d)$, then $\min_{\alpha\in\mathcal{A}}\max_{d\in\mathcal{C}_k} -S(\alpha, d) \leq \min_{\alpha\in\mathcal{A}}\max_{d\in\mathcal{D}} -S(\alpha, d)$, i.e. $\beta_k \leq \theta^*$. On the other hand, for $\forall j = 1, \cdots, k$, $-S(\alpha_j, d^{j+1}) = \max_{d\in\mathcal{D}} -S(\alpha_j, d)$, thus $(\alpha_j, -S(\alpha_j, d^{j+1}))$ is a feasible solution pairs of (4). Then $\theta^* \leq -S(\alpha_j, d^{j+1})$ for $j = 1, \cdots, k$, and hence we have $\theta^* \leq \varphi_k = \min_{1\leq j\leq k} -S(\alpha_j, d^{j+1})$. With the number of iteration k increasing, the subset $\mathcal{C}_k$ is monotonically increasing, so β_k is monotonic increasing while $\{\varphi_k\}$ is monotonically decreasing. The proof is completed. □

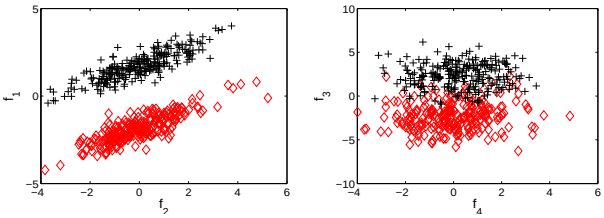
The following theorem further indicates that FGM can obtain a global solution to (4) within a finite number of steps.

Theorem 2. *Assume that in each exchanged iteration, subproblems in Section 2.3 and 2.4 can be solved, FGM stops after a finite number of steps with a global solution of (4).*

Proof. We can measure the convergence of FGM by the gap difference of series $\{\beta_k\}$ and $\{\varphi_k\}$. After a finite iterations, the objective value will no longer improve. Assume in kth iteration, there is no update of $\mathcal{C}_k$, i.e. $d^{k+1} = \arg\max_{d\in\mathcal{D}} -S(\alpha_k, d) \in \mathcal{C}_k$, then $\mathcal{C}_k = \mathcal{C}_{k+1}$. So, we

Table 1. Datasets used in the experiments

DATASET	# FEATURES	# TRAINING PTS.	# TEST PTS.
WDBC	30	569	–
USPS	241	1,500	–
BREAST CANCER	7,129	38	–
LEUKEMIA	7,129	72	–
REAL-SIM	20,958	32,309	40,000
RCV1.BINARY	47,236	20,242	677,399
ARXIV ASTRO-PH	99,757	62,369	32,487
NEWS20.BINARY	1,355,191	9,996	10,000
URL0	3,231,961	16,000	20,000
URL1	3,231,961	20,000	20,000



(a) $\mathbf{f}_1$&$\mathbf{f}_2$ (b) $\mathbf{f}_3$&$\mathbf{f}_4$

Figure 1. Detailed information of the synthetic dataset

can prove that, in this case, (α_k, β_k) is the globally optimal solution pair of (4). First, since $\mathcal{C}_k = \mathcal{C}_{k+1}$, in Algorithm 1, there will be no update of α, i.e. $\alpha_{k+1} = \alpha_k$. Then we have $-S(\alpha_k, d^{k+1}) = \max_{d\in\mathcal{D}} -S(\alpha_k, d) = \max_{d\in\mathcal{C}_k} -S(\alpha_k, d) = \max_{1\le i\le k} -S(\alpha_k, d^i) = \beta_k$, and $\varphi_k = \min_{1\le j\le k} -S(\alpha_j, d^{j+1}) \le \beta_k$. From Theorem 1, we know $\beta_k \le \theta^* \le \varphi_k$, then we obtain $\beta_k = \theta^* = \varphi_k$, and (α_k, β_k) is the globally optimal solution pair of (4). The proof is completed. □

2.7. Computational Complexity

QCQP-SSVM, SDP-SSVM and LPSVM are convex optimization problems, but they are very expensive even on medium-sized datasets. For NMMKL, it has to solve a QCQP problem with m quadratic constraints. Obviously, if m is too large, NMMKL is very computationally expensive. For SVM-RFE, its computational complexity largely depends on the number of features eliminated in each step. Assume that the training of linear SVM takes $O(nm)$ time, if only one feature is removed from the feature list in each elimination step, SVM-RFE will take $O(nm^2)$ time. SVM-RFE method can be speeded up by removing chunks of features in each step, which, however, may lead to a significant decline in the classification performance. In contrast, FGM only needs to solve a series of MKL problems and find the most violated d. Empirically, a maximum of 10 iterations is enough for FGM to converge. Moreover, all base kernels are linear, so the subproblem of finding α of linear SVM in Section 2.3 can be solved by LIBLinear software, which scales linearly in n and m (Hsieh et al., 2008). And the time complexity of MKL is proportional to the complexity of linear SVM. Finding the most violated d can be obtained exactly by finding the B largest ones from m coefficients c_j^2's, which takes only $O(m \log B)$ time. The time complexity of each iteration of FGM is $O(nm+m \log B)$. Thus, FGM is computationally efficient even for large-scale and very high dimensional problems.

3. Experiments

3.1. Datasets

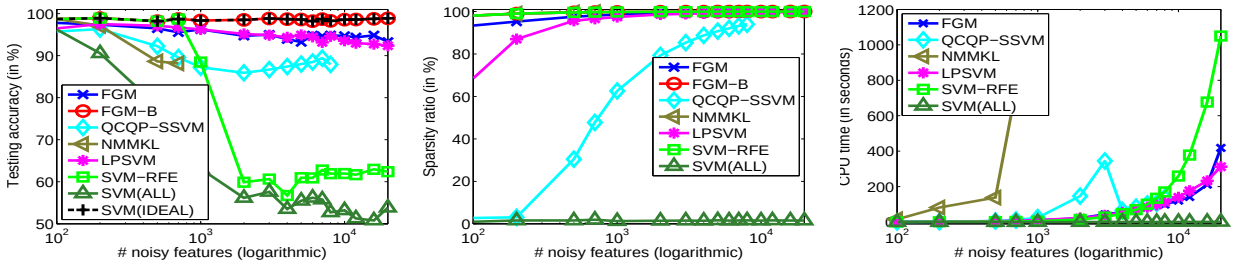
In this Section, we evaluate the performance of various methods on a synthetic dataset and a collection of real world datasets. The synthetic dataset is a binary classification problem with three informative features $\mathbf{f}_1$, $\mathbf{f}_2$ and $\mathbf{f}_3$, and the generation of data follows the description in (Xu et al., 2009). The plot of the informative features are shown in Figure 1, where $\mathbf{f}_4$ is a noise feature. Among the three features, $\mathbf{f}_1$ and $\mathbf{f}_2$ are composite features and generated by two different multi-variate Gaussian distributions. As a group, they are the most informative features to the classification while $\mathbf{f}_3$ is the most informative feature as a single. Ideally, a good SSVM or a "non-monotonic" feature selection method should successfully identify the features $\mathbf{f}_1$ and $\mathbf{f}_2$ as a group which is referred to as *ideal features*.

The real world datasets consist of two categories. The first category includes 2 small datasets and 2 Microarray datasets[2]. For these small datasets, we use the cross validation scheme to validate the performance due to the insufficient samples. For the second category, it contains 6 large-scale and very high dimensional datasets. Among them, `real-sim`, `rcv1.binary` and `news20.binary` are from LIBSVM website[3] while `Arxiv astro-ph` can be referred to (Joachims, 2006). For `Arxiv astro-ph` and `rcv1.binary`, they have already been split into training set and testing set. For `real-sim` and `news20.binary`, we manually split them into training set and testing set. The last two are 2 URL datasets from an anonymized 120-day subset of the ICML-09 URL data (Ma et al., 2009). The original URL dataset contains 120 independent subsets collected from 120 days[4]. Because of space limitation, we only use the data from the first three days. In our experiments, we train on data collected from the previous day and predict on data collected from the next day, resulting in two new datasets denoted by `URL0` and `URL1`. Detailed information of these datasets and the splitting information are summarized in Table 1. For all the datasets, each dimension of the data is normalized to zero mean and unit variance.

[2] `wdbc` is from the UCI machine learning repository, and the binary `usps` dataset is from http://www.kyb.tuebingen.mpg.de/ssl-book/benchmarks.html; ii) Microarray datasets `Breast cancer` and `Leukemia` are from http://www.kyb.tuebingen.mpg.de/bs/people/weston/l0

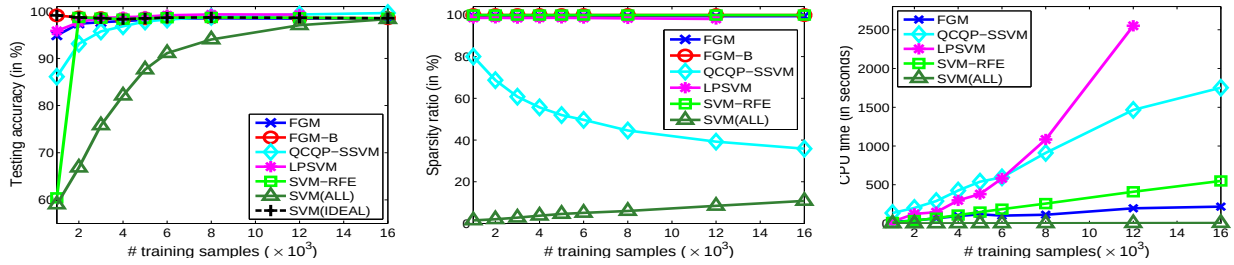
[3] http://www.csie.ntu.edu.tw/∼cjlin /libsvmtools/datasets/

[4] http://archive.ics.uci.edu/ml/datasets/url+reputation



(a) Testing accuracy (b) Sparsity (c) Training time

Figure 2. Results on synthetic dataset with varying noise features



(a) Testing accuracy (b) Sparsity (c) Training time

Figure 3. Results on synthetic dataset with varying training samples

Table 2. Sparsity ratio of various methods on small datasets

DATA SETS	LPSVM		FGM	QCQP-SSVM
	(IN %)	B	(IN %)	(IN %)
WDBC	17.56±16.00	$0.1m$	84.00±2.38	**86.67±0.00**
		$0.2m$	**77.78±2.20**	65.56±15.50
		$0.3m$	**69.11±1.94**	0.33±1.02
		$0.4m$	**59.22±2.09**	0.00±0.00
USPS	48.23±14.91	$0.1m$	**87.74± 3.34**	6.03± 4.29
		$0.2m$	**80.98±2.59**	7.98±5.82
		$0.3m$	**73.44±5.26**	7.95±5.78
		$0.4m$	**73.99±1.71**	8.49±6.46
BREASTCANCER	89.46±29.94	2	**99.92±0.02**	12.59±29.99
		5	**99.86±0.03**	79.90±40.57
		10	**99.81±0.03**	99.72±0.05
		20	**99.71±0.03**	99.44±0.05
LEUKEMIA	99.79±0.03	2	**99.90±0.02**	1.26± 1.19
		5	**99.85±0.03**	49.46±50.35
		10	**99.71±0.04**	99.53±0.07
		20	**99.62±0.03**	99.25±0.08

3.2. Experimental Setup

In our experiments, comparisons are conducted among FGM, QCQP-SSVM, NMMKL, LPSVM and SVM-RFE. And we do not include the results of SDP-SSVM for its high computational cost (Chan et al., 2007), because SimpleMKL may select multiple kernels (i.e. multiple d's may be selected), the number of features selected by FGM is self-determined and may be larger than B. This property, although may lead to a decline in sparsity, is very important to the "non-monotonic" feature selection because usually we have no idea of how many features should be selected in advance. Alternatively, we can select B features by ranking the features as in SVM-RFE and NMMKL according to the score c_j, and form the final feature subset. We name this strategy as FGM-B in this paper. For the better illustration of the sparsity property, we define a sparsity ratio as: $\rho(w) = 1 - \frac{Card(w)}{m}$, which means the ratio of zeros in w. $Card(w)$ here denotes the number of nonzeros in w. However, for LPSVM and QCQP-SSVM, it is hard for them to achieve completely sparse solutions for some problems. Alternatively, we define $Card(w)$ for LPSVM and QCQP-SSVM as the number of weights w_j with large relative magnitude, i.e the number of elements with $|w_j|/\max_i(|w_i|) \geq 10^{-4}$ (Chan et al., 2007). Experiments are performed with a 2.27GHZ Interl(R)Core(TM) 4 DUO CPU running Windows Server 2003 with 24.0GB main memory. We use MOSEK (version 5.0.0.127) for solving QCQP and LPSVM, and use LIBlinear[5] to solve FGM, NMMKL and SVM-RFE. The dual coordinate descent for L2-SVM (DCDL2) algorithm is adopted as the baseline classification method.

3.3. Experiments on Synthetic Data

To thoroughly study the performance of different methods, two synthetic experiments are performed. At first, we generate 1000 instances (500 for each class) and then randomly choose 500 for training and the rest for testing. Then, we gradually increase the noise features to the dataset and test whether the considered methods can successfully identify the former 2 informative features. The initial number of noise features are set to 100. For the noise features, half of

[5]http://www.csie.ntu.edu.tw/ ∼cjlin/liblinear/

Table 3. Testing accuracies (in %) of various methods on small datasets

DATA SETS	LPSVM		FGM	FGM-B	QCQP-SSVM	NMMKL	SVM-RFE
	(IN %)	B	(IN %)	(IN %)	(IN %)	(IN %)	(IN %)
WDBC	88.71±1.57	$0.1m$	91.11±2.52	90.96±1.50	86.77±2.40	**92.15±1.17**	**92.15±1.29**
		$0.2m$	91.81±2.37	**92.44±1.10**	90.85±2.44	92.37±1.14	92.25±1.20
		$0.3m$	92.46±1.13	92.43±0.99	90.70±1.42	92.30±1.45	**92.51±1.10**
		$0.4m$	92.47±1.13	92.37±1.08	90.57±1.46	92.18±1.24	**92.51±1.22**
USPS	89.05±1.79	$0.1m$	**90.78±1.36**	79.69±1.89	84.32±9.06	80.38±1.20	80.62±1.33
		$0.2m$	**91.47±1.27**	81.97±2.27	84.76±8.91	83.47±2.02	85.78±1.24
		$0.3m$	**91.23±1.88**	87.49±1.05	83.08±8.14	87.90±1.26	88.82±1.30
		$0.4m$	**91.99±1.43**	89.60±1.07	83.08±8.30	89.47±1.06	90.08±1.40
BREAST CANCER	80.63±15.60	2	**86.88±8.10**	67.92±17.50	48.96±14.13	-	76.04±17.61
		5	**88.75±8.90**	87.50±11.72	72.29±18.62	-	85.00±10.46
		10	87.92±11.60	**89.58±11.76**	79.79±15.46	-	88.54±10.52
		20	87.71±10.57	88.13±11.17	81.04±16.29	-	**89.17±10.63**
LEUKEMIA	87.01 ± 8.53	2	**91.61±5.85**	85.75±5.71	62.87±22.10	-	88.39±4.75
		5	93.91±5.56	**95.06± 4.31**	80.80±15.09	-	93.91±4.85
		10	94.37±4.39	94.48±5.17	93.91±6.38	-	**95.63±4.87**
		20	95.52±4.26	**95.75±3.91**	91.26±7.56	-	**95.75±3.81**

them are uniformly distributed and another half are generated by Gaussian distribution. In this experiment, the parameter C for all algorithms is set to 0.1 and B to 2. The parameter τ in NMMKL is set to $\frac{1}{C}$. In SVM-RFE, as suggested in (Guyon et al., 2002), when the total features are greater than 200, we remove 10 features at each time; otherwise, we eliminate one feature at one time. To show whether the mentioned methods can identify the composite features $\mathbf{f}_1$ and $\mathbf{f}_2$, in Figure 2(a), we denote the testing accuracy obtained by using ideal features as SVM(IDEAL). Meanwhile, we also use SVM with all features as the baseline method which is denoted by SVM(ALL). All experiments are repeated 5 times.

In Figure 2(a), with the number of noise features increasing, only FGM-B can obtain the same accuracy as SVM(IDEAL), i.e. FGM-B successfully identifies the composite features (or ideal features). As for QCQP-SSVM, its sparsity increases along with the increasing number of features. However, the testing accuracy decreases when the number of noise feature increases. Another problem of QCQP-SSVM has a rapid increase in the memory usage with the increasing features. In our experiment, the MOSEK solver for QCQP encounters out-of-memory problem when the number of features exceeds 8000. NMMKL can identify the composite feature when the noise is relatively small. When the noise increases, the performance of NMMKL will decline. In addition, from Figure 2(c), NMMKL cannot deal with very high dimensional problems. LPSVM shows competitive performance compared with FGM. SVM-RFE performs well when the number of noise features is not very large (less than 2500). For SVM(ALL), from Figure 2(a), when the number of noises exceeds 2000, SVM cannot learn an accurate classifier. Figure 2(c) shows the training time of various methods. Obviously, compared with SVM-RFE, FGM is more computationally efficient when dealing with higher dimensional problems.

In the second experiment, we fix the number of features to 2000 and then gradually increase the number of instances. Half of the whole instances are used as training samples and the other half are for testing. The initial instances are set to 1000. Because NMMKL is incapable of dealing with large dimensions, we did not include its results in this experiment. Figure 3 shows the testing accuracy, sparsity ratio and training time of various methods. Although the testing accuracy of FGM is lower than SVM(IDEAL), the testing accuracy of FGM-B is the same as SVM(IDEAL), which indicates that FGM can also identify the composite features. As to SVM-RFE, it fails to identify the first two composite features when the number of training samples is relatively small. However, as the training examples increasing, the performance becomes better. Finally, from Figure 3(b), QCQP-SSVM and LPSVM cannot provide good sparsity for classifiers. Both of them also cannot handle large sample problems.

3.4. Experiments on Small Datasets

For the experiments on the small datasets, we randomly split them into 60% for training and the rest 40% for testing. For `wdbc` and `usps`, the parameter C is selected using 5-fold cross validation over the range $\{0.01, 0.05, 0.1, 0.5, 1, 5\}$. For the Microarray datasets, due to the small sample size, we simply set $C = 0.01$. As to the second parameter B, it is chosen in a range of $\{0.1m, 0.2m, 0.3m, 0.4m\}$ for the first two datasets and $\{2, 5, 10, 20\}$ for the Microarray datasets. As to SVM-RFE, we follow the experimental settings of the synthetic experiment. All the methods are repeated 30 times and averaged performances are reported.

Table 2 summarizes the performances of sparsity achieved by various methods. For FGM-B, SVM-RFE and NMMKL, as their sparsity can be directly computed by using the parameter B, we do not include their sparsity results in Table 2. Here, we did not obtain the results of

Table 4. Training time (in s) of various methods on small datasets

DATA SETS	LPSVM		FGM	QCQP-SSVM	NMMKL	SVM-RFE
	(IN S)	B	(IN S)	(IN S)	(IN S)	(IN S)
WDBC	0.25±0.03	$0.1m$	2.09±1.49	**0.23±0.03**	1.17±0.66	0.45±0.07
		$0.2m$	1.41±1.04	**0.23±0.03**	1.30±0.73	0.29±0.18
		$0.3m$	1.14±0.70	**0.23±0.03**	1.26±0.62	0.33±0.14
		$0.4m$	0.95±0.68	**0.23±0.03**	1.41±0.80	0.35±0.10
USPS	6.66± 1.12	$0.1m$	73.15±66.31	**3.06±0.46**	94.33±69.43	10.60±12.95
		$0.2m$	18.84±8.36	**2.88±0.31**	49.04±46.70	7.86±5.05
		$0.3m$	13.60±8.22	**2.85±0.23**	55.44±24.58	7.73±5.43
		$0.4m$	7.58±1.94	**2.90±0.28**	72.82±13.20	7.98±7.60
BREAST CANCER	**0.67±0.10**	2	9.24±3.54	8.29±0.62	-	9.35±0.78
		5	9.18±3.61	8.78±0.59	-	9.33±0.75
		10	7.84±2.50	9.02±0.69	-	9.32±0.86
		20	5.96±2.36	9.11±0.59	-	9.31±0.77
LEUKEMIA	**2.07±0.21**	2	13.01±5.96	9.71±0.96	-	28.67±2.89
		5	18.32±7.37	10.10±0.94	-	28.43±2.92
		10	17.01±4.38	10.34±0.70	-	28.40±3.00
		20	16.16±3.52	10.48±0.63	-	28.18±2.79

NMMKL on the Microarray datasets because it cannot handle such high dimensional problems. As expected, in general, FGM can obtain the most sparse results on all datasets. The testing accuracies of various methods are listed in Table 3. From this table, we can observe that FGM can obtain competitive results or even better results on all the small benchmark datasets. Table 4 lists the training time of various methods spent on different small datasets. From this table, we can observe that QCQP-SSVM and LPSVM shows better efficiency on dealing with small problems. However, both of them are incapable of very large problems.

3.5. Experiments on Very High Dimensional Datasets

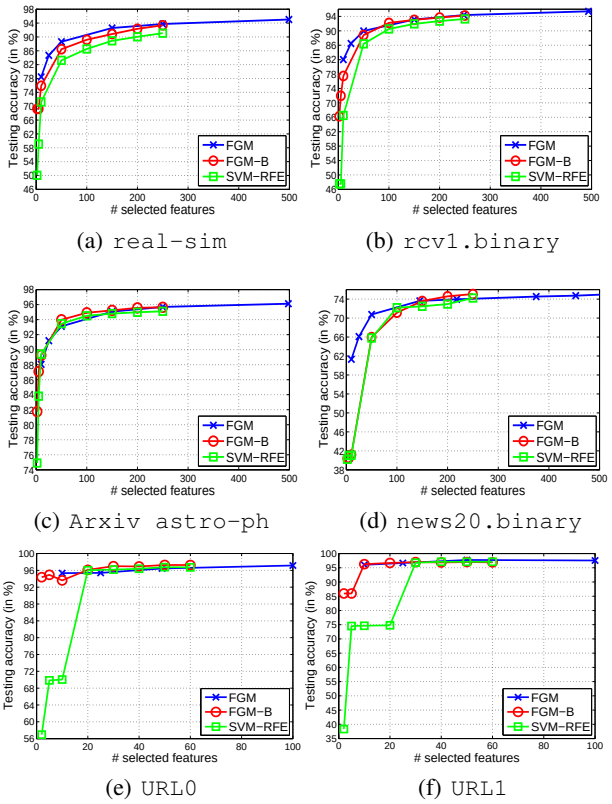
In this subsection, we verify the performance of FGM on large-scale datasets listed in Table 1. These datasets have both large number of instances and features. Note, some of the methods, such as NMMKL, QCQP-SSVM and LPSVM cannot be used due to their high computational cost or high memory storage. Therefore, we only consider the comparison among FGM, FGM-B and SVM-RFE. For the parameter settings, we did the experiments by fixing $C = 5$. As to the parameter B for FGM and SVM-RFE, we set it in the range of $\{2, 5, 10, 50, 100, 150, 200, 250\}$ for the former four datasets and $\{2, 5, 10, 20, 30, 40, 50, 60\}$ for the two URL datasets. For SVM-RFE, we remove 100 features in each step for the first four large datasets. However, for the two URL datasets, SVM-RFE with 100 features elimination of each step is still very computationally expensive, hence we remove 10,000 features in each step if the number of remaining features is larger than 20,000. We respectively recorded the testing accuracy against the number of selected features and the training time versus different B in Figure 4 and Figure 5, respectively. From Figure 4, we have the following observations: (a) On `real-sim`, `rcv1.binary` and `news20.binary` datasets, FGM obviously outperforms SVM-RFE and FGM-B on testing accuracy with selected features. Meanwhile, FGM-B also shows improved performance compared with SVM-RFE on these datasets. (b) On `Arxiv astro-ph` dataset, although FGM does not show significant improvements compared with SVM-RFE, its counterpart, FGM-B, is slightly better than SVM-RFE. (c) From the results of the two URL datasets, we can easily see that FGM is much better than SVM-RFE when identifying a small number of features. Finally, from Figure 5, we can conclude that FGM is very efficient when dealing with very high dimensional problems.

4. Conclusion

In this paper, we propose a novel SVM algorithm to learn a sparse feature subset for classification. In particular, a 0-1 vector is introduced into SVM to control whether or not the features are selected, resulting in a Mixed Integer Programming (MIP) problem. By introducing a convex relaxation, the MIP is further transformed into a convex Multiple Kernel Learning problem with exponentially large number of base kernels. Finally, an efficient and scalable cutting plane algorithm, namely "Feature Generating Machine (FGM)", is introduced to iteratively generates and learns a pool of informative and sparse feature subsets. Because FGM only requires to solve a small number of MKL problems with very few linear kernels, and the internal subproblem of MKL only involves linear SVM that can be solved by state-of-the-art LIBLinear software, FGM is very suitable for solving large-scale and very high dimensional problems. Moreover, with the property of global convergence, the size of the final feature subset in FGM can be optimally determined, catering to the "non-monotonic" requirement in feature selection. Comprehensive experiments on both synthetic dataset and real-word datasets verify the good classification performance and efficiency of FGM.

Acknowledgments

This research was in part supported by Singapore MOE AcRF Tier-1 Research Grant (RG15/08).

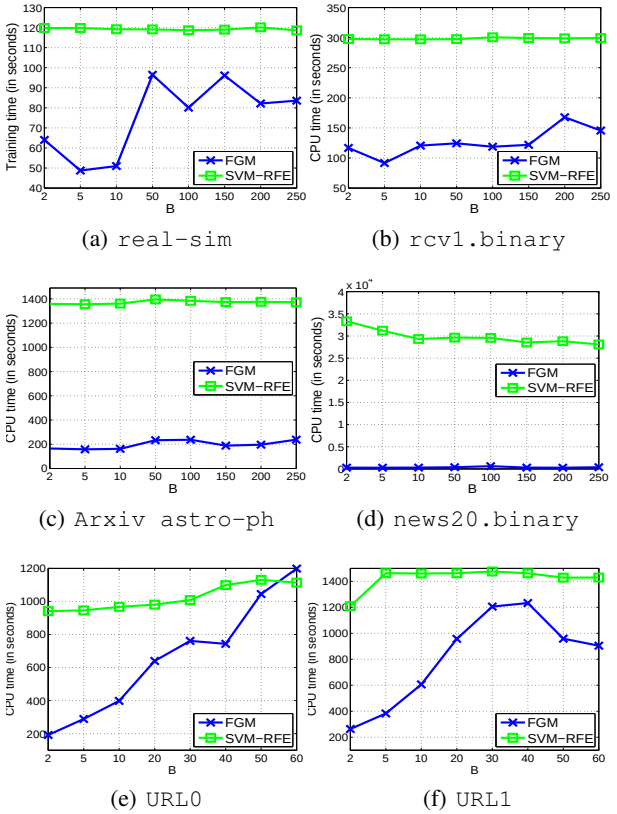


(a) real-sim (b) rcv1.binary

(c) Arxiv astro-ph (d) news20.binary

(e) URL0 (f) URL1

Figure 4. Testing accuracy on different data sets



(a) real-sim (b) rcv1.binary

(c) Arxiv astro-ph (d) news20.binary

(e) URL0 (f) URL1

Figure 5. Training time on various data sets

References

Blum, A. L. and Langley, P. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245–271, 1997.

Bradley, P. S. and Mangasarian, O. L. Feature selection via concave minimization and support vector machines. In *ICML*, 1998.

Chan, A.B., Vasconcelos, N., and Lanckriet, G.R.G. Direct convex relaxations of sparse SVM. In *ICML*, 2007.

Chen, J. and Ye, J. Training SVM with indefinite kernels. In *ICML*, 2008.

Fung, G.M. and Mangasarian, O.L. A feature selection newton method for support vector machine classification. *Computational Optimization and Applications*, 28: 185–202, 2004.

Guyon, I. and Elisseeff, A. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

Hsieh, C.-J., Chang, K.-W., Lin, C.-J., Keerthi, S. S., and Sundararajan, S. A dual coordinate descent method for large-scale linear SVM. In *ICML*, 2008.

Joachims, T. Training linear SVMs in linear time. In *ACM KDD*, 2006.

Kelley, J. E. The cutting plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

Kim, S.-J. and Boyd, S. A minimax theorem with applications to machine learning, signal processing, and finance. *SIAM Journal on Optimization*, 2008.

Li, Y.F., Kwok, J.T., Tsang, I.W., and Zhou, Z.H. A convex method for locating regions of interest with multi-instance learning. In *ECML*, 2009a.

Li, Y.F., Tsang, I.W., Kwok, J.T., and Zhou, Z.H. Tighter and convex maximum margin clustering. In *AISTATS*, 2009b.

Ma, J., Saul, L. K., Savage, S., and Voelker, G. M. Identifying suspicious URLs: An application of large-scale online learning. In *ICML*, 2009.

Rakotomamonjy, A., F., Bach, Y., Grandvalet, and S., Canu. SimpleMKL. *J. Mach. Learn. Res.*, 9:2491–2521, 2008.

Weston, J., Elisseeff, A., and Scholkopf, B. Use of the zero-norm with linear models and kernel methods. *J. Mach. Learn. Res.*, 3:1439–1461, 2003.

Xu, Z., Jin, R., J., Ye, Lyu, Michael R., and I, King. Non-monotonic feature selection. In *ICML*, 2009.

Zhu, J., Rossett, S., Hastie, T., and Tibshirani, R. 1-norm support vector machines. In *NIPS*, 2003.