Random Projections for Support Vector Machines

Saurabh Paul
Computer Science Dept.
Rensselaer Polytechnic Inst.
pauls2@rpi.edu

Christos Boutsidis
Mathematical Sciences Dept.
IBM Research
cboutsi@us.ibm.com

Malik Magdon-Ismail
Computer Science Dept.
Rensselaer Polytechnic Inst.
magdon@cs.rpi.edu

Petros Drineas
Computer Science Dept.
Rensselaer Polytechnic Inst.
drinep@cs.rpi.edu

Abstract

Let $\mathbf{X}$ be a data matrix of rank ρ, representing n points in d-dimensional space. The linear support vector machine constructs a hyperplane separator that maximizes the 1-norm soft margin. We develop a new oblivious dimension reduction technique which is precomputed and can be applied to any input matrix $\mathbf{X}$. We prove that, with high probability, the margin and minimum enclosing ball in the feature space are preserved to within ϵ-relative error, ensuring comparable generalization as in the original space. We present extensive experiments with real and synthetic data to support our theory.

1 Introduction

The Support Vector Machine (SVM) (Cristianini and Shawe-Taylor, 2000) is a popular classifier in machine learning today. The training data set consists of n points $\mathbf{x}_i \in \mathbb{R}^d$, with respective labels $y_i \in \{-1, +1\}$ for $i = 1 \ldots n$. For linearly separable data, the primal form of the SVM learning problem is to construct a hyperplane $\mathbf{w}^*$ which maximizes the geometric *margin* (the minimum distance of a data point to the hyperplane), while separating the data. For non-separable data the "soft" 1-norm margin is maximized. The dual lagrangian formulation of the problem leads to the following quadratic program:

$$\max_{\{\alpha_i\}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to} \sum_{i=1}^{n} y_i \alpha_i = 0, \tag{1}$$

$$0 \le \alpha_i \le C, \quad i = 1 \ldots n.$$

In the above formulation, the unknown lagrange multipliers $\{\alpha_i\}_{i=1}^{n}$ are constrained to lie inside the "box constraint" $[0, C]^n$, where C is part of the input. In order to measure the out-of-sample performance of the SVM, we can use the VC-dimension of *fat*-separators. Assuming that the data lie in a ball of radius B, and that the hypothesis set consists of hyperplanes of width γ (corresponding to the margin), then the VC-dimension of this hypothesis set is $O(B^2/\gamma^2)$ (Vapnik, 1998). Now, given the in-sample error, we can obtain a bound for the out-of-sample error, which is monotonic in the VC-dimension (Vapnik and Chervonenkis, 1971).

The main intuition behind our work is that if we can preserve a subspace geometry, then we should be able to preserve the performance of a distance based algorithm, like the SVM classifier. We construct dimension reduction matrices $\mathbf{R} \in \mathbb{R}^{d \times r}$ which produce r-dimensional feature vectors $\tilde{\mathbf{x}}_i = \mathbf{R}^T \mathbf{x}_i$; the matrices $\mathbf{R}$ do not depend on the data. We show that for the data in the dimension-reduced space, the margin of separability and the minimum enclosing ball radius are preserved, since the subspace geometry is preserved. So, an SVM with an appropriate structure defined by the margin (width) of the hyperplanes (Vapnik and Chervonenkis, 1971) will have comparable VC-dimension and, thus, generalization error. To state our results precisely, we first need some SVM basics.

1.1 Notation and SVM Basics

$\mathbf{A}, \mathbf{B}, \ldots$ denote matrices and $\boldsymbol{\alpha}, \mathbf{b}, \ldots$ denote column vectors; $\mathbf{e}_i$ (for all $i = 1 \ldots n$) is the standard basis, whose dimensionality will be clear from context; and $\mathbf{I}_n$ is the $n \times n$ identity matrix. The Singular Value Decomposition (SVD) of a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ of rank $\rho \le \min\{n, d\}$ is equal to $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, where $\mathbf{U} \in \mathbb{R}^{n \times \rho}$ is an orthogonal matrix containing the left singular vectors, $\boldsymbol{\Sigma} \in \mathbb{R}^{\rho \times \rho}$ is a diagonal matrix containing the singular values $\sigma_1 \ge \sigma_2 \ge \ldots \sigma_\rho > 0$, and $\mathbf{V} \in \mathbb{R}^{d \times \rho}$ is a matrix containing the right singular

vectors. The spectral norm of $\mathbf{A}$ is $\|\mathbf{A}\|_2 = \sigma_1$.

We introduce matrix notation that we will use for the remainder of the paper. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the matrix whose rows are the vectors $\mathbf{x}_i^T$, $\mathbf{Y} \in \mathbb{R}^{n \times n}$ be the diagonal matrix with entries $\mathbf{Y}_{ii} = y_i$, and $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_n] \in \mathbb{R}^n$ be the vector of lagrange multipliers to be determined by solving eqn. (1). The SVM optimization problem is

$$\max_{\boldsymbol{\alpha}} \mathbf{1}^T \boldsymbol{\alpha} - \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{Y} \mathbf{X} \mathbf{X}^T \mathbf{Y} \boldsymbol{\alpha} \tag{2}$$
$$\text{subject to } \mathbf{1}^T \mathbf{Y} \boldsymbol{\alpha} = 0; \qquad \text{and} \qquad \mathbf{0} \leq \boldsymbol{\alpha} \leq \boldsymbol{C}.$$

(In the above, $\mathbf{1}$, $\mathbf{0}$, $\boldsymbol{C}$ are vectors with the implied constant entry.) Let $\boldsymbol{\alpha}^*$ be an optimal solution of the above problem. The optimal separating hyperplane is given by $\mathbf{w}^* = \mathbf{X}^T \mathbf{Y} \boldsymbol{\alpha}^* = \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i$, and the points $\mathbf{x}_i$ for which $\alpha_i^* > 0$, i.e., the points which appear in the expansion $\mathbf{w}^*$, are the support vectors. The geometric margin, γ^*, of this canonical optimal hyperplane is $\gamma^* = 1/\|\mathbf{w}^*\|_2$, where $\|\mathbf{w}^*\|_2^2 = \sum_{i=1}^n \alpha_i^*$. The data radius is $B = \min_{\mathbf{x}^*} \max_{\mathbf{x}_i} \|\mathbf{x}_i - \mathbf{x}^*\|_2$. It is this γ^* and B that factor into the generalization performance of the SVM through the ratio B/γ^*.

1.2 Dimension Reduction

Our goal is to study how the SVM performs under (linear) dimensionality reduction transformations in the feature space. Let $\mathbf{R} \in \mathbb{R}^{d \times r}$ be the dimension reduction matrix that reduces the dimensionality of the input from d to $r < d$. We will choose $\mathbf{R}$ to be a random projection matrix (see Section 2). The transformed dataset into r dimensions is given by $\tilde{\mathbf{X}} = \mathbf{X} \mathbf{R}$, and the SVM optimization problem becomes

$$\max_{\tilde{\boldsymbol{\alpha}}} \mathbf{1}^T \tilde{\boldsymbol{\alpha}} - \frac{1}{2} \tilde{\boldsymbol{\alpha}}^T \mathbf{Y} \mathbf{X} \mathbf{R} \mathbf{R}^T \mathbf{X}^T \mathbf{Y} \tilde{\boldsymbol{\alpha}},$$
$$\text{subject to } \mathbf{1}^T \mathbf{Y} \tilde{\boldsymbol{\alpha}} = 0, \qquad \text{and} \qquad \mathbf{0} \leq \tilde{\boldsymbol{\alpha}} \leq \boldsymbol{C}. \tag{3}$$

Solving the dimensionally-reduced problem above is computationally more efficient than solving the original, d-dimensional problem. We will present a construction for $\mathbf{R}$ that leverages the fast Hadamard transform. The running time needed to apply this construction to the original data matrix is $O(nd \log r)$. Notice that while this running time is nearly linear on the size of the original data, it does not take advantage of any sparsity in the input. In order to address this deficiency, we leverage the recent work of Clarkson and Woodruff (2013), which proposes a construction for $\mathbf{R}$ that can be applied to $\mathbf{X}$ in $O\left(nnz(\mathbf{X}) + poly\left(n\epsilon^{-1}\right)\right)$ time; here $nnz(\mathbf{X})$ denotes the number of non-zero entries of $\mathbf{X}$. To the best of our knowledge, this is the first independent implementation and evaluation of this potentially ground-breaking random projection

technique (a few experimental results were presented by Clarkson and Woodruff (2013)). All constructions for $\mathbf{R}$ are oblivious of the data and hence they can be precomputed. Also, all generalization bounds that depend on the final margin and radius of the data will continue to hold.

1.3 Our Contribution

Let ρ be the rank of $\mathbf{X}$. In the transformed space, let the resulting margin after solving the optimization problem be $\tilde{\gamma}^*$, and assume that the projected data have data radius $\tilde{B}$. Our main theoretical result is to show that, for suitably chosen values of r, both the margin and the data radius are preserved to relative error:

$$\tilde{\gamma}^{*2} \geq (1 - \epsilon) \gamma^{*2}; \qquad \tilde{B}^2 \leq (1 + \epsilon) B^2.$$

Thus, it is possible to *obliviously* reduce the dimension of the data while preserving the good generalization properties of the SVM. We briefly discuss the appropriate values of r: if $\mathbf{R}$ is the randomized Hadamard transform, we need to set $r = O\left(\rho\epsilon^{-2} \cdot \log\left(\rho d \delta^{-1}\right) \cdot \log\left(\rho\epsilon^{-2} \delta^{-1} \log\left(\rho d \delta^{-1}\right)\right)\right)$; if $\mathbf{R}$ is constructed as described in Clarkson and Woodruff (2013), then $r = O\left(\rho\epsilon^{-4} \log\left(\rho/\delta\epsilon\right) \left(\rho + \log\left(1/\delta\epsilon\right)\right)\right)$. The running time needed to apply the former transform on $\mathbf{X}$ is $O\left(nd \log r\right)$; the running time needed to apply the latter transform is $O\left(nnz(\mathbf{X}) + poly\left(n\epsilon^{-1}\right)\right)$.

1.4 Prior work

The work most closely related to our results is that of Krishnan et al. (2008), which improved upon Balcazar et al. (2001), and showed that by using sub-problems based on Gaussian random projections, one can obtain a solution to the SVM problem with a margin that is relative-error close to the optimal. Their sampling complexity (the parameter r in our parlance) depends on B^4, and, most importantly, on $1/\gamma^{*2}$. This bound is not directly comparable to our result, which only depends on the rank of the data manifold, and holds regardless of the margin of the original problem (which could be arbitrarily small). Our results dramatically improve the running time needed to apply the random projections; our running times are (theoretically) linear in the number of non-zero entries in $\mathbf{X}$, whereas Krishnan et al. (2008) necessitates $O(ndr)$ time to apply $\mathbf{R}$ on $\mathbf{X}$.

Shi et al. (2012) discussed the theory of margin and angle preservation after random projections using Gaussian matrices for binary and multi-class problems. There are two main differences between their result and ours. They show margin preservation to within

additive error, whereas we give margin preservation to within relative error. This is a big difference, especially when the margin is small. Moreover, they analyze only the separable case. We analyze the general non-separable dual problem and give a result in terms of the norm of the weight vector. For the separable case, the norm of the weight vector directly relates to the margin. For the non-separable case, one has to analyze the actual quadratic program, and our result essentially claims that the solution in the transformed space will have comparably regularized weights as the solution in the original space. Shi et al. (2009) used hash kernels (which approximately preserved inner product like random projections), to design a biased approximation of the kernel matrix and showed that their generalization bounds on the hash kernel and the original kernel differed by the inverse of the product of the margin and number of datapoints. For smaller margins, this difference will be high. Our generalization bounds are independent of the original margin and hold for arbitrarily small margins.

Zhang et al. (2012) developed algorithms to accurately recover the optimal solution to the original SVM optimization problem using a Gaussian random projection. This is different from our work since we analyze generalization performance using random projection and do not try to recover the solution. Finally, it is worth noting that random projection techniques have been applied extensively in the compressed sensing literature, and our theorems have the same flavor to a number of results in that area. However, to the best of our knowledge, the compressed sensing literature has not investigated the 1-norm soft-margin SVM optimization problem.

2   Random Projection Matrices

Random projections are extremely popular techniques in order to deal with the curse-of-dimensionality. Let the data matrix be $\mathbf{X} \in \mathbb{R}^{n \times d}$ (n data points in $\mathbb{R}^d$) and let $\mathbf{R} \in \mathbb{R}^{d \times r}$ (with $r \ll d$) be a random projection matrix. Then, the projected data matrix is $\tilde{\mathbf{X}} = \mathbf{XR} \in \mathbb{R}^{n \times r}$ (n points in $\mathbb{R}^r$). If $\mathbf{R}$ is carefully chosen, then all pairwise Euclidean distances are preserved with high probability. Thus, the geometry of the set of points in preserved, and it is reasonable to hope that an optimization objective such as the one that appears in SVMs will be only mildly perturbed.

There are many possible constructions for the matrix $\mathbf{R}$ that preserve pairwise distances. The most common one is a matrix $\mathbf{R}$ whose entries are i.i.d. standard Gaussian random variables (Indyk and Motwani, 1998; Dasgupta and Gupta, 2003). Achlioptas (2003) argued that the random sign matrix – **RS** for short – i.e., a matrix whose entries are set to +1 or −1 with equal

probability, also works. Li et al. (2006) used the sparse random projection matrix whose entries were set to +1 or −1 with probability $1/2\sqrt{d}$ and 0 with probability $(1 - 1/\sqrt{d})$. These constructions take $O(ndr)$ time to compute $\tilde{\mathbf{X}}$. More recently, faster methods of constructing random projections have been developed, using, for example, the Fast Hadamard Transform (Ailon and Chazelle, 2006) – **FHT** for short. The Hadamard-Walsh matrix for any d that is a power of two is defined as

$$\mathbf{H}_d = \begin{bmatrix} \mathbf{H}_{d/2} & \mathbf{H}_{d/2} \\ \mathbf{H}_{d/2} & -\mathbf{H}_{d/2} \end{bmatrix} \in \mathbb{R}^{d \times d},$$

with $\mathbf{H}_1 = +1$. The normalized Hadamard-Walsh matrix is $\sqrt{\frac{1}{d}}\mathbf{H}_d$, which we simply denote by $\mathbf{H}$. We set:

$$\mathbf{R}_{\text{SRHT}} = \sqrt{\frac{d}{r}}\mathbf{DHS}, \qquad (4)$$

a rescaled product of three matrices. $\mathbf{D} \in \mathbb{R}^{d \times d}$ is a random diagonal matrix with $\mathbf{D}_{ii}$ equal to ± 1 with probability $\frac{1}{2}$. $\mathbf{H} \in \mathbb{R}^{d \times d}$ is the normalized Hadamard transform matrix. $\mathbf{S} \in \mathbb{R}^{d \times r}$ is a random *sampling matrix* which randomly samples columns of $\mathbf{DH}$; specifically, each of the r columns of $\mathbf{S}$ is independent and selected uniformly at random (with replacement) from the columns of $\mathbf{I}_d$, the identity matrix. This construction assumes that d is a power of two. If not, we just pad $\mathbf{X}$ with columns of zeros (affecting run times by at most a factor of two). The important property of this transform is that the projected features $\tilde{\mathbf{X}} = \mathbf{XR}$ can be computed efficiently in $O(nd \log r)$ time (see Theorem 2.1 of Ailon and Liberty (2008) for details). An important property of $\mathbf{R}$ (that follows from prior work) is that it preserves orthogonality. While the randomized Hadamard transform is a major improvement over prior work, it does not take advantage of any sparsity in the input matrix. To fix this, very recent work (Clarkson and Woodruff, 2013) shows that carefully constructed random projection matrices can be applied in input sparsity time by making use of generalized sparse embedding matrices. Their construction of $\mathbf{R}$ is somewhat complicated and we present it in the full version of our paper (Paul et al., 2012). We will call the method of Clarkson and Woodruff (2013) to construct a sparse embedding matrix **CW**.

3   Geometry of SVM is preserved under Random Projection

We now state and prove our main result, namely that solving the SVM optimization problem in the projected space results in comparable margin and data radius as in the original space. These results are dependent on the main technical result from numerical linear algebra literature which we state in the lemma below.

Lemma 1. *Fix $\epsilon \in (0, \frac{1}{2}]$, $\delta \in (0, 1]$. Let $\mathbf{V} \in \mathbb{R}^{d \times \rho}$ be any matrix with orthonormal columns and set $\mathbf{R} = \mathbf{R}_{\text{SRHT}}$ as in eqn (4), with $r = O(\rho\epsilon^{-2} \cdot \log(\rho d \delta^{-1}) \cdot \log(\rho\epsilon^{-2}\delta^{-1} \log(\rho d \delta^{-1})))$. Then, with probability at least $1 - \delta$, $\|\mathbf{V}^T\mathbf{V} - \mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}\|_2 \leq \epsilon$.*

Remark. Let $\mathbf{R}$ be the random sign matrix described in Section 2. Then, $\|\mathbf{V}^T\mathbf{V} - \mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}\|_2 \leq \epsilon$ still holds with probability at least $1 - 1/n$, by setting $r = O\left(\rho\epsilon^{-2}\log\rho\log d\right)$. The proof of this result is essentially the same, using Theorem 3.1(i) of Magen and Zouzias (2011). A similar result can be proven for the construction of Clarkson and Woodruff (2013) by setting $r = O\left(\rho\epsilon^{-4}\log\left(\rho/\delta\epsilon\right)\left(\rho + \log\left(1/\delta\epsilon\right)\right)\right)$. These lemmas can be found in the full version of our paper (Paul et al., 2012).

Theorem 1. *Let ϵ be an accuracy parameter, $\mathbf{R} \in \mathbb{R}^{d \times r}$ be any matrix for which $\|\mathbf{V}^T\mathbf{V} - \mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}\|_2 \leq \epsilon$, and let $\tilde{\mathbf{X}} = \mathbf{X}\mathbf{R}$. Let γ^* and $\tilde{\gamma}^*$ be the margins obtained by solving the SVM problems using data $\mathbf{X}$ and $\tilde{\mathbf{X}}$ respectively (eqns. (2) and (3)). Then, $\tilde{\gamma}^{*2} \geq (1 - \epsilon) \cdot \gamma^{*2}$.*

In words, Theorem 1 argues that for suitably large r (linear in the rank of $\mathbf{X}$ up to logarithmic factors), the margin is preserved. Theorem 1 will follow from the technical result that $\|\mathbf{V}^T\mathbf{V} - \mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}\|_2 \leq \epsilon$ holds with high probability depending on the choice of the random projection matrix.

Proof: (of Theorem 1) Let $\mathbf{E} = \mathbf{V}^T\mathbf{V} - \mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}$, where $\mathbf{E} \in \mathbb{R}^{\rho \times \rho}$, and $\boldsymbol{\alpha}^* = [\alpha_1^*, \alpha_2^*, \ldots, \alpha_n^*]^T \in \mathbb{R}^n$ be the vector achieving the optimal solution for the problem of eqn. (2) in Section 1. Then,

$$
\begin{aligned}
Z_{opt} &= \sum_{i=1}^{n} \alpha_i^* - \frac{1}{2}\boldsymbol{\alpha}^{*T}\mathbf{Y}\mathbf{X}\mathbf{X}^T\mathbf{Y}\boldsymbol{\alpha}^* \\
&= \sum_{i=1}^{n} \alpha_i^* - \frac{1}{2}\boldsymbol{\alpha}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\boldsymbol{\alpha}^* \\
&= \sum_{i=1}^{n} \alpha_i^* - \frac{1}{2}\boldsymbol{\alpha}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\boldsymbol{\alpha}^* \\
&\quad - \frac{1}{2}\boldsymbol{\alpha}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{E}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\boldsymbol{\alpha}^*. \quad (5)
\end{aligned}
$$

Let $\tilde{\boldsymbol{\alpha}}^* = [\tilde{\alpha}_1^*, \tilde{\alpha}_2^*, \ldots, \tilde{\alpha}_n^*]^T \in \mathbb{R}^n$ be the vector achieving the optimal solution for the dimensionally-reduced SVM problem of eqn. (3) using $\tilde{\mathbf{X}} = \mathbf{X}\mathbf{R}$. Using the SVD of $\mathbf{X}$, we get

$$
\tilde{Z}_{opt} = \sum_{i=1}^{n} \tilde{\alpha}_i^* - \frac{1}{2}\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^*.
$$
$$(6)$$

Since the constraints on $\boldsymbol{\alpha}^*, \tilde{\boldsymbol{\alpha}}^*$ do not depend on the data (see eqns. (2) and (3)), it is clear that $\tilde{\boldsymbol{\alpha}}^*$ is a feasible solution for the problem of eqn. (2). Thus, from the optimality of $\boldsymbol{\alpha}^*$, and using eqn. (6), it follows that

$$
\begin{aligned}
Z_{opt} &= \sum_{i=1}^{n} \alpha_i^* - \frac{1}{2}\boldsymbol{\alpha}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\boldsymbol{\alpha}^* \\
&\quad - \frac{1}{2}\boldsymbol{\alpha}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{E}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\boldsymbol{\alpha}^* \\
&\geq \sum_{i=1}^{n} \tilde{\alpha}_i^* - \frac{1}{2}\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^* \\
&\quad - \frac{1}{2}\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{E}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^* \\
&= \tilde{Z}_{opt} - \frac{1}{2}\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{E}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^*. \quad (7)
\end{aligned}
$$

We now analyze the second term using standard submultiplicativity properties and $\mathbf{V}^T\mathbf{V} = \mathbf{I}_\rho$. Taking $\mathbf{Q} = \tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}$,

$$
\begin{aligned}
\frac{1}{2}\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{E}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^* &\leq \frac{1}{2}\|\mathbf{Q}\|_2\|E\|_2\left\|\mathbf{Q}^T\right\|_2 \\
&= \frac{1}{2}\|\mathbf{E}\|_2\|\mathbf{Q}\|_2^2 \\
&= \frac{1}{2}\|\mathbf{E}\|_2\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\right\|_2^2 \\
&= \frac{1}{2}\|\mathbf{E}\|_2\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\right\|_2^2. \quad (8)
\end{aligned}
$$

Combining eqns. (7) and (8), we get

$$
Z_{opt} \geq \tilde{Z}_{opt} - \frac{1}{2}\|\mathbf{E}\|_2\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\right\|_2^2. \quad (9)
$$

We now proceed to bound the second term in the right-hand side of the above equation. Towards that end, we bound the difference:

$$
\begin{aligned}
&\left|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\mathbf{R}\mathbf{R}^T\mathbf{X}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^* - \tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\mathbf{X}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^*\right| \\
&= \left|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\left(\mathbf{V}^T\mathbf{R}\mathbf{R}^T\mathbf{V} - \mathbf{V}^T\mathbf{V}\right)\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^*\right| \\
&= \left|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\left(-\mathbf{E}\right)\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{Y}\tilde{\boldsymbol{\alpha}}^*\right| \\
&\leq \|\mathbf{E}\|_2\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\right\|_2^2 \\
&= \|\mathbf{E}\|_2\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\right\|_2^2 \\
&= \|\mathbf{E}\|_2\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\right\|_2^2.
\end{aligned}
$$

We can rewrite the above inequality as $\left|\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\mathbf{R}\right\|_2^2 - \left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\right\|_2^2\right| \leq \|\mathbf{E}\|_2\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\right\|_2^2$; thus,

$$
\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\right\|_2^2 \leq \frac{1}{1 - \|\mathbf{E}\|_2}\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\mathbf{R}\right\|_2^2.
$$

Combining with eqn. (9), we get

$$
Z_{opt} \geq \tilde{Z}_{opt} - \frac{1}{2}\left(\frac{\|\mathbf{E}\|_2}{1 - \|\mathbf{E}\|_2}\right)\left\|\tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\mathbf{R}\right\|_2^2. \quad (10)
$$

Now recall from our discussion in Section 1 that $\mathbf{w}^{*T} = \boldsymbol{\alpha}^{*T}\mathbf{Y}\mathbf{X}$, $\tilde{\mathbf{w}}^{*T} = \tilde{\boldsymbol{\alpha}}^{*T}\mathbf{Y}\mathbf{X}\mathbf{R}$, $\|\mathbf{w}^*\|_2^2 = \sum_{i=1}^{n}\alpha_i^*$, and

$\|\tilde{\mathbf{w}}^*\|_2^2 = \sum_{i=1}^n \tilde{\alpha}_i^*$. Then, the optimal solutions Z_{opt} and $\tilde{Z}_{opt}$ can be expressed as follows:

$$Z_{opt} = \|\mathbf{w}^*\|_2^2 - \frac{1}{2}\|\mathbf{w}^*\|_2^2 = \frac{1}{2}\|\mathbf{w}^*\|_2^2, \quad (11)$$

$$\tilde{Z}_{opt} = \|\tilde{\mathbf{w}}^*\|_2^2 - \frac{1}{2}\|\tilde{\mathbf{w}}^*\|_2^2 = \frac{1}{2}\|\tilde{\mathbf{w}}^*\|_2^2. \quad (12)$$

Combining eqns. (10), (11), and (12), we get

$$\|\mathbf{w}^*\|_2^2 \geq \|\tilde{\mathbf{w}}^*\|_2^2 - \left(\frac{\|E\|_2}{1 - \|E\|_2}\right)\|\tilde{\mathbf{w}}^*\|_2^2$$

$$= \left(1 - \frac{\|E\|_2}{1 - \|E\|_2}\right)\|\tilde{\mathbf{w}}^*\|_2^2. \quad (13)$$

Let $\gamma^* = \|\mathbf{w}^*\|_2^{-1}$ be the geometric margin of the problem of eqn. (2) and let $\tilde{\gamma}^* = \|\tilde{\mathbf{w}}^*\|_2^{-1}$ be the geometric margin of the problem of eqn. (3). Then, the above equations imply:

$$\gamma^{*2} \leq \left(1 - \frac{\|E\|_2}{1 - \|E\|_2}\right)^{-1}\tilde{\gamma}^{*2}$$

$$\Rightarrow \tilde{\gamma}^{*2} \geq \left(1 - \frac{\|E\|_2}{1 - \|E\|_2}\right)\gamma^{*2}. \quad (14)$$

$\diamond$

Our second theorem argues that the radius of the minimum ball enclosing all projected points (the rows of the matrix $\mathbf{XR}$) is very close to the radius of the minimum ball enclosing all original points (the rows of the matrix $\mathbf{X}$). The proof can be found in the full version of our paper (Paul et al., 2012).

Theorem 2. *Let ϵ be an accuracy parameter, $\delta \in (0, 1)$ and consider the SVM formulations of eqns. (2) and (3), let B be the radius of the minimum ball enclosing all points in the full-dimensional space, and let $\tilde{B}$ be the radius of the ball enclosing all points in the projected subspace. For $\mathbf{R}$ as in Theorem 1, with probability at least $1 - \delta$, $\tilde{B}^2 \leq (1 + \epsilon)B^2$.*

4 Experiments

We describe experimental evaluations on two real-world datasets, namely a collection of document-term matrices (the TechTC-300 dataset (Davidov et al., 2004) and a population genetics dataset (the joint Human Genome Diversity Panel or HGDP (Li et al., 2008) and the HapMap Phase 3 data (Paschou et al., 2010) and also on three synthetic datasets. The synthetic datasets and the TechTC-300 dataset correspond to binary classification tasks and the joint HapMap-HGDP dataset corresponds to a multi-class classification task, and our algorithms perform well here as well.

In our experimental evaluations, we implemented random projections using three different methods: **RS**,

FHT, and **CW** (see Section 2 for definitions) in MATLAB version 7.13.0.564 (R2011b). We ran the algorithms using the same values of r (the dimension of the projected feature space) for all algorithms, but we varied r across different datasets. We used LIBSVM (Chang and Lin, 2011) as our linear SVM solver with default settings. In all cases, we ran our experiments on the original full data (referred to as "full" in the results), as well as on the projected data. We partitioned the data randomly for ten-fold cross-validation in order to estimate out-of-sample error. We repeated this partitioning ten times to get ten ten-fold cross-validation experiments. In order to estimate the effect of the randomness in the construction of the random projection matrices, we repeated our cross-validation experiments ten times using ten different random projection matrices for all datasets. We report in-sample error (ϵ_{in}), out-of-sample error (ϵ_{out}), the time to compute random projections (t_{rp}), the total time needed to both compute random projections *and* run SVMs on the lower-dimensional problem (t_{run}), and the margin (γ). All results are averaged over the ten cross-validation experiments and the ten choices of random projection matrices. For each of the aforementioned quantities, we report both its mean value μ and its standard deviation σ. For the multi-class experiment of Section 4.3, we do not report a margin.

Table 1: Synthetic data: ϵ_{out} decreases as a function of r in all three families of matrices, using any of the three random projection methods. μ and σ indicate the mean and the standard deviation of ϵ_{out} over ten matrices in each family $D1$, $D2$, and $D3$, ten ten-fold cross-validation experiments, and ten choices of random projection matrices for the three methods that we investigated (a total of 1,000 experiments for each family of matrices).

ϵ_{out}		Projected Dimension r			
		256	512	1024	**full**
D1	CW (μ)	24.08	19.45	16.66	**15.10**
	(σ)	4.52	4.15	3.52	**2.60**
	RS (μ)	24.1.0	19.46	16.36	**15.10**
	(σ)	4.45	3.79	3.22	**2.60**
	FHT (μ)	23.52	19.59	16.67	**15.10**
	(σ)	4.21	4.05	3.37	**2.60**
D2	CW (μ)	25.94	21.07	17.33	**15.44**
	(σ)	4.13	4.16	3.45	**2.54**
	RS (μ)	25.80	20.80	17.47	**15.44**
	(σ)	4.40	3.93	3.42	**2.54**
	FHT (μ)	25.33	21.23	17.58	**15.44**
	(σ)	3.69	4.24	3.53	**2.54**
D3	CW (μ)	27.62	22.97	18.93	**15.83**
	(σ)	3.46	3.22	3.32	**2.00**
	RS (μ)	28.15	23.00	18.72	**15.83**
	(σ)	3.02	3.48	2.78	**2.00**
	FHT (μ)	27.92	23.41	18.73	**15.83**
	(σ)	3.46	3.60	3.02	**2.00**

4.1 Synthetic datasets

The synthetic datasets are separable by construction. More specifically, we first constructed a weight vector $\mathbf{w} \in \mathbb{R}^d$, whose entries were selected in i.i.d. trials from a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ of mean μ and standard-deviation σ. We experimented with the following three distributions: $\mathcal{N}(0,1)$, $\mathcal{N}(1,1.5)$, and $\mathcal{N}(2,2)$. Then, we normalized $\mathbf{w}$ to create $\hat{\mathbf{w}} = \mathbf{w}/\|\mathbf{w}\|_2$. Let $\mathbf{X}_{ij} = \mathcal{N}(0,1)$; then, we set $\mathbf{x}_i$ to be equal to the i-th row of $\mathbf{X}$, while $\mathbf{y}_i = sign\left(\hat{\mathbf{w}}^T \mathbf{x}_i\right)$. We generated families of matrices of different dimen-

Table 2: Synthetic data: γ increases as a function of r in all three families of matrices. See the caption of Table 1 for an explanation of μ and σ.

γ		PROJECTED DIMENSION r			
		256	512	1024	**full**
D1	CW (μ)	5.72	6.67	7.16	**7.74**
	(σ)	0.58	0.58	0.59	**0.59**
	RS (μ)	5.73	6.66	7.18	**7.74**
	(σ)	0.57	0.55	0.55	**0.59**
	FHT (μ)	5.76	6.64	7.15	**7.74**
	(σ)	0.56	0.58	0.56	**0.59**
D2	CW (μ)	6.62	8.09	8.88	**9.78**
	(σ)	0.64	0.62	0.59	**0.66**
	RS (μ)	6.65	8.10	8.88	**9.78**
	(σ)	0.64	0.60	0.63	**0.66**
	FHT (μ)	6.66	8.06	8.84	**9.78**
	(σ)	0.63	0.65	0.63	**0.66**
D3	CW (μ)	7.69	9.84	11.07	**12.46**
	(σ)	0.67	0.60	0.71	**0.69**
	RS (μ)	7.61	9.85	11.05	**12.46**
	(σ)	0.59	0.6212	0.62	**0.69**
	FHT (μ)	7.63	9.83	11.11	**12.46**
	(σ)	0.67	0.64	0.64	**0.69**

sions. More specifically, family $D1$ contained matrices in $\mathbb{R}^{200 \times 5,000}$; family $D2$ contained matrices in $\mathbb{R}^{250 \times 10,000}$; and family $D3$ contained matrices in $\mathbb{R}^{300 \times 20,000}$. We generated ten datasets for each of the families $D1$, $D2$, and $D3$, and we report average results over the ten datasets. We set r to 256, 512, and 1024 and set C to 1,000 in LIBSVM for all the experiments. Tables 1 and 2 show ϵ_{out} and γ for the three datasets $D1$, $D2$, and $D3$. ϵ_{in} is zero for all three data families. As expected, ϵ_{out} and γ improve as r grows for all three random projection methods. Also, the time needed to compute random projections is very small compared to the time needed to run SVMs on the projected data. Figure 1 shows the combined running time of random projections and SVMs, which is nearly the same for all three random projection methods. It is obvious that this combined running time is much smaller than the time needed to run SVMs on the full dataset (without any dimensionality reduction). For instance, for $r = 1024$, t_{run} for $D1$, $D2$, and $D3$ is (respectively) 6, 9, and 25 times smaller than t_{run} on the full-data.
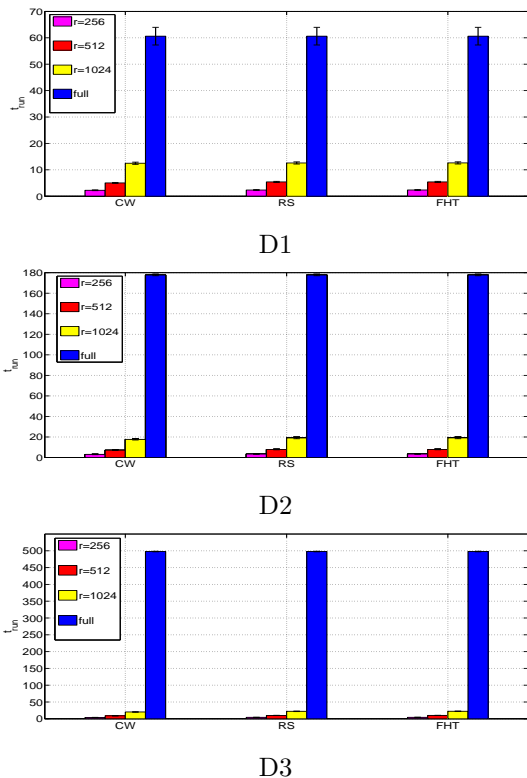

Figure 1: Total (average) running times, in seconds, of random projections *and* SVMs on the lower-dimensional data for each of the three families of synthetic data. Vertical bars indicate the, relatively small, standard deviation (see the caption of Table 1).

4.2 The TechTC-300 dataset

For our first real dataset, we use the TechTC-300 data, consisting of a family of 295 document-term data matrices. The TechTC-300 dataset comes from the Open Directory Project (ODP), which is a large, comprehensive directory of the web, maintained by volunteer editors. Each matrix in the TechTC-300 dataset contains a pair of categories from the ODP. Each category corresponds to a label, and thus the resulting classification task is binary. The documents that are collected from the union of all the subcategories within each category are represented in the bag-of-words model, with the words constituting the features of the data (Davidov et al., 2004). Each data matrix consists of 150-280 documents (the rows of the data matrix $\mathbf{X}$), and each document is described with respect to 10,000-40,000 words (features, columns of the matrix $\mathbf{X}$). Thus, TechTC-300 provides a diverse collection of data sets for a systematic study of the performance of the SVM on the projected versus full data. We set the parameter C to 500 in LIBSVM for all 295 document-term matrices and set r to 128, 256, and 512. We use a lower value of C than for the other data sets for com-

Table 3: Results on the Techtc300 dataset, averaged over 295 data matrices using three different random projection methods. The table shows how ϵ_{out}, γ, t_{rp} (in seconds), and t_{run} (in seconds) depend on r. μ and σ indicate the mean and the standard deviation of each quantity over 295 matrices, ten ten-fold cross-validation experiments, and ten choices of random projection matrices for the three methods that we investigated.

| | | Projected Dimension r | | | |
		128	256	512	**full**
$\epsilon_{\mathbf{out}}$	CW(μ)	24.63	22.84	21.26	**17.35**
	(σ)	10.57	10.37	10.17	**9.45**
	RS(μ)	24.58	22.90	21.38	**17.35**
	(σ)	10.57	10.39	10.23	**9.45**
	FHT (μ)	24.63	22.93	21.35	**17.35**
	(σ)	10.66	10.39	10.2	**9.45**
γ	CW (μ)	1.66	1.88	1.99	**2.09**
	(σ)	3.68	3.79	3.92	**4.00**
	RS (μ)	1.66	1.88	1.99	**2.09**
	(σ)	3.65	3.80	3.91	**4.00**
	FHT (μ)	1.66	1.88	1.98	**2.09**
	(σ)	3.65	3.81	3.88	**4.00**
$\mathbf{t_{rp}}$	CW (μ)	0.0046	0.0059	0.0075	––
	(σ)	0.0019	0.0026	0.0033	––
	RS (μ)	0.0429	0.0855	0.1719	––
	(σ)	0.0178	0.0356	0.072	––
	FHT (μ)	0.0443	0.0882	0.1764	––
	(σ)	0.0206	0.0413	0.0825	––
$\mathbf{t_{run}}$	CW (μ)	1.23	2.22	4.63	**4.85**
	(σ)	0.87	0.93	1.93	**2.12**
	RS (μ)	0.99	1.53	3.02	**4.85**
	(σ)	0.97	0.59	1.12	**2.12**
	FHT (μ)	0.95	1.46	2.83	**4.85**
	(σ)	0.96	0.55	1.02	**2.12**

is lost when SVMs are applied on the dimensionally-reduced data. Indeed, the combined running time t_{run} is fastest for **FHT**, followed by **RS** and **CW**. In all cases, the total running time is smaller than the SVM running time on full dataset. For example, in the case of **FHT**, setting $r = 512$ achieves a running time t_{run} which is about 70% faster than running SVMs on the full dataset; ϵ_{out} increases by less than 4%.

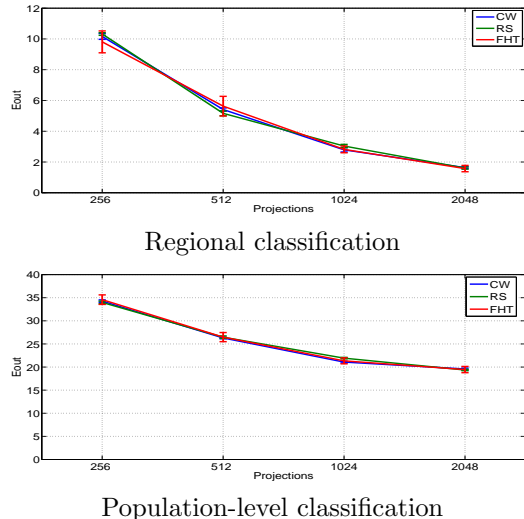Regional classification

Population-level classification

Figure 2: ϵ_{out} as a function of r in the Hapmap-HGDP dataset for three different random projection methods and two different classification tasks. Vertical bars indicate the standard-deviation over the ten ten-fold cross-validation experiments and the ten choices of the random projection matrices for each of the three methods.

putational reasons: larger C is less efficient. We note that our classification accuracy is slightly worse (on the full data) than the accuracy presented in Section 4.4 of Davidov et al. (2004), because we did not fine-tune the SVM parameters as they did, since that is not the focus of this study. For every dataset and every value of r we tried, the in-sample error on the projected data matched the in-sample error on the full data. We thus focus on ϵ_{out}, the margin γ, the time needed to compute random projections t_{rp}, and the total running time t_{run}. We report our results averaged over 295 data matrices. Table 3 shows the behavior of these parameters for different choices of r. As expected, ϵ_{out} and the margin γ improve as r increases, and they are nearly identical for all three random projection methods. The time needed to compute random projections is smallest for **CW**, followed by **RS** and **FHT**. As a matter of fact, t_{rp} for **CW** is ten to 20 times faster than **RS** and **FHT** for different values of r. This is predicted by the theory in Clarkson and Woodruff (2013), since **CW** is optimized to take advantage of input sparsity. However, this advantage

4.3 The HapMap-HGDP dataset

Predicting ancestry of individuals using a set of genetic markers is a well-studied classification problem. We use a population genetics dataset from the Human Genome Diversity Panel (HGDP) and the HapMap Phase 3 dataset (see Paschou et al. (2010) for details), in order to classify individuals into broad geographic regions, as well as into (finer-scale) populations. We study a total of 2,250 individuals from approximately 50 populations and five broad geographic regions. The features in this dataset correspond to $492,516$ Single Nucleotide Polymorphisms (SNPs), which are well-known biallelic loci of genetic variation across the human genome. Each entry in the resulting $2,250 \times 492,516$ matrix is set to $+1$ (homozygotic in one allele), -1 (homozygotic in the other allele), or 0 (heterozygotic), depending on the genotype of the respective SNP for a particular sample. Missing entries were filled in with -1, $+1$, or 0, with probability $1/3$. Each sample has a known population and region of origin, which constitute its label. We

Total running time: regional classification

Total running time: population-level classification
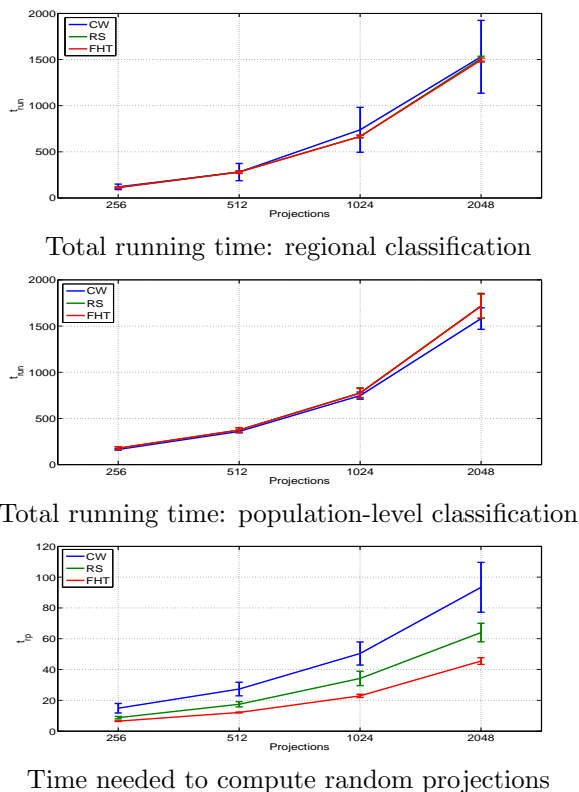

Time needed to compute random projections

Figure 3: Total running time in seconds (random projections *and* SVM classification on the dimensionally-reduced data) for Hapmap-HGDP dataset for three different projection methods using both regional and population-level labels. Notice that the time needed to compute random projection is independent of the classification labels. Vertical bars indicate standard-deviation, as in Figure 2.

set r to 256, 512, 1024, and 2048 in our experiments. Since this task is a multi-class classification problem, we used LIBSVM's one-against-one technique for classification. We ran two sets of experiments: in the first set, the classification problem is to assign samples to broad regions of origin, while in the second experiment, our goal is to classify samples into (fine-scale) populations. We set C to 1,000 in LIBSVM for all the experiments. The in-sample error is zero in all cases. Figure 2 shows the out-of-sample error for regions and populations classification, which are nearly identical for all three random projection methods. For regional classification, we estimated ϵ_{out} to be close to 2%, and for population-level classification, ϵ_{out} is close to 20%. This experiment strongly supports the computational benefits of our methods in terms of main memory. $\mathbf{X}$ is $2,250 \times 492,516$, which is too large to fit into memory in order to run SVMs. Figure 3 shows that the combined running time for three different random projection methods are nearly identical for both regions and population classification tasks. However, the time needed to compute the random projections is different

from one method to the next. **FHT** is fastest, followed by **RS** and **CW**. In this particular case, the input matrix in quite dense, and **CW** seems to be outperformed by the other two methods.

5   Conclusions and open problems

We present theoretical and empirical results indicating that random projections are a useful dimensionality reduction technique for SVM classification problems that handle sparse or dense data in high-dimensional feature spaces. Our theory predicts that the dimensionality of the projected space (denoted by r) has to grow essentially *linearly* (up to logarithmic factors) in ρ (the rank of the data matrix) in order to achieve relative error approximations to the margin and the radius of the minimum ball enclosing the data. Such relative-error approximations imply excellent generalization performance. However, our experiments show that considerably smaller values for r (e.g., in the case of the TechTC data, setting r to 1/70-th of all available features) results in classification that is essentially as accurate as running SVMs on all available features, despite the fact that the matrices have full numerical rank. This seems to imply that our theoretical results can be improved. **FHT** and **RS** work well on dense data while **CW** is an excellent choice for sparse data, as indicated by our experiments. However, this solid performance of **CW** (which is predicted by the theoretical bounds of Clarkson and Woodruff (2013))) comes at a cost, at least according to our experimental evaluation: solving the SVM optimization problem on the resulting low-dimensional dataset is quite expensive, and, as a result, the total running time of the **CW** method is eventually higher than that of **FHT** and **RS**. This seems to indicate that more research is necessary in terms of random projection methods that are both fast (e.g., can be applied on the input matrix in time that is proportional to the number of non-zero entries in the matrix), but also result in low-dimensional data matrices that are "friendly" (e.g., correspond to well-structured problem instances) for SVM solvers. Understanding this aspect of random projection matrices is important and it has not been investigated at all in existing literature.

References

D. Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687, 2003.

N. Ailon and B. Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 557–563, 2006.

N. Ailon and E. Liberty. Fast dimension reduction using Rademacher series on dual BCH codes. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1–9, 2008.

J.L. Balcazar, Y. Dai, and O. Watanabe. A Random Sampling Technique for Training Support Vector Machines. In *Proceedings of the 12th International Conference on Algorithmic Learning Theory*, pages 119–134, 2001.

C-C. Chang and C-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

K.L. Clarkson and D.W. Woodruff. Low Rank Approximation and Regression in Input Sparsity Time. In *Proceedings of the 45th ACM Symposium on the Theory of Computing (To appear)*, 2013.

N. Cristianini and J. Shawe-Taylor. *Support Vector Machines and other kernel-based learning methods.* Cambridge University Press, 2000.

S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures and Algorithms*, 22(1):60–65, 2003.

D. Davidov, E. Gabrilovich, and S. Markovitch. Parameterized generation of labeled datasets for text categorization based on a hierarchical directory. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 250–257, 2004. `http://techtc.cs.technion.ac.il/techtc300/techtc300.html`.

P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998.

S. Krishnan, C. Bhattacharya, and R. Hariharan. A Randomized Algorithm for Large Scale Support Vector Learning. In *Advances in 20th Neural Information Processing Systems*, pages 793–800, 2008.

J.Z. Li, D.M. Absher, H. Tang, A.M. Southwick, A.M. Casto, S. Ramachandran, H.M. Cann, G.S. Barsh, M. Feldman, L.L. Cavalli-Sforza, and R.M. Myers. Worldwide human relationships inferred from genome-wide patterns of variation. *Science*, 319 (5866):1100–1104, 2008.

P. Li, T.J. Hastie, and W.K. Church. Very sparse random projections. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 287–296, 2006.

A. Magen and A. Zouzias. Low rank matrix-valued Chernoff bounds and approximate matrix multiplication. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1422–1436, 2011.

P. Paschou, J. Lewis, A. Javed, and P. Drineas. Ancestry informative markers for fine-scale individual assignment to worldwide populations. *Journal of Medical Genetics*, 47(12):835–47, 2010.

S. Paul, C. Boutsidis, M. Magdon-Ismail, and P. Drineas. Random Projections for Support Vector Machines. abs/1211.6085, 2012. `http://arxiv.org/abs/1211.6085`.

Q. Shi, J. Petterson, G. Dror, J. Langford, A. Smola, and S.V.N. Vishwanathan. Hash kernels for structured data. *Journal of Machine Learning Research*, 10:2615–2637, 2009.

Q. Shi, C. Shen, R. Hill, and A.V.D Hengel. Is margin preserved after random projection ? In *Proceedings of 29th International Conference on Machine Learning*, pages 591–598, 2012.

V. N. Vapnik. Statistical Learning Theory. *Theory of Probability and its Applications*, 16:264–280, 1998.

V.N. Vapnik and A. Chervonenkis. On the Uniform Convergence of Relative Frequencies of Events to their Probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971.

L. Zhang, M. Mahdavi, R. Jin, and T. Yang. Recovering Optimal Solution by Dual Random Projection. abs/1211.3046, 2012. `http://arxiv.org/abs/1211.3046`.