Approximating a Gram Matrix for Improved Kernel-Based Learning
(Extended Abstract)

Petros Drineas\textsuperscript{1} and Michael W. Mahoney\textsuperscript{2}

\textsuperscript{1} Department of Computer Science, Rensselaer Polytechnic Institute, Troy, New York 12180
drinep@cs.rpi.edu

\textsuperscript{2} Department of Mathematics, Yale University, New Haven, CT 06520
mahoney@cs.yale.edu

Abstract. A problem for many kernel-based methods is that the amount of computation required to find the solution scales as $O(n^3)$, where $n$ is the number of training examples. We develop and analyze an algorithm to compute an easily-interpretable low-rank approximation to an $n \times n$ Gram matrix $G$ such that computations of interest may be performed more rapidly. The approximation is of the form $\tilde{G}_k = CW_k^+ C^T$, where $C$ is a matrix consisting of a small number $c$ of columns of $G$ and $W_k$ is the best rank-$k$ approximation to $W$, the matrix formed by the intersection between those $c$ columns of $G$ and the corresponding $c$ rows of $G$. An important aspect of the algorithm is the probability distribution used to randomly sample the columns; we will use a judiciously-chosen and data-dependent nonuniform probability distribution. Let $\|\cdot\|_2$ and $\|\cdot\|_F$ denote the spectral norm and the Frobenius norm, respectively, of a matrix, and let $G_k$ be the best rank-$k$ approximation to $G$. We prove that by choosing $O(k/\epsilon^4)$ columns

$$\|G - CW_k^+ C^T\|_\xi \leq \|G - G_k\|_\xi + \epsilon \sum_{i=1}^n G_{ii}^2,$$

both in expectation and with high probability, for both $\xi = 2, F$, and for all $k : 0 \leq k \leq \text{rank}(W)$. This approximation can be computed using $O(n)$ additional space and time, after making two passes over the data from external storage.

1 Introduction

1.1 Background

Given a collection $\mathcal{X}$ of data points, which are often but not necessarily elements of $\mathbb{R}^m$, techniques such as linear Support Vector Machines (SVMs), Gaussian Processes (GPs), Principle Component Analysis (PCA), and the related Singular Value Decomposition (SVD), identify and extract structure from $\mathcal{X}$ by
computing linear functions, i.e., functions in the form of dot products, of the data. For example, in PCA the subspace spanned by the first $k$ eigenvectors is used to give a $k$ dimensional model of the data with minimal residual; thus, it provides a low-dimensional representation of the data. Such spectral analysis has a rich theoretical foundation and has numerous practical applications.

In many cases, however, there is nonlinear structure in the data (or the data, e.g. text, may not support the basic linear operations of addition and scalar multiplication). In these cases, kernel-based learning methods have proved to be quite useful [7, 27]. Kernel-based learning methods are a class of statistical learning algorithms, the best known examples of which are SVMs [7]. In this approach, data items are mapped into high-dimensional spaces, where information about their mutual positions (in the form of inner products) is used for constructing classification, regression, or clustering rules. Kernel-based algorithms exploit the information encoded in the inner product between all pairs of data items and are successful in part because there is often an efficient method to compute inner products between very complex or even infinite dimensional vectors. Thus, kernel-based algorithms provide a way to deal with nonlinear structure by reducing nonlinear algorithms to algorithms that are linear in some feature space $\mathcal{F}$ that is nonlinearly related to the original input space.

More precisely, assume that the data consists of vectors $X^{(1)}, \ldots, X^{(n)} \in \mathcal{X} \subset \mathbb{R}^m$ and let $X = [X^{(1)} \ldots X^{(n)}] \in \mathbb{R}^{m \times n}$ be the matrix whose $i$-th column is $X^{(i)}$. In kernel-based methods, a set of features is chosen that define a space $\mathcal{F}$, where it is hoped relevant structure will be revealed, the data $\mathcal{X}$ are then mapped to the feature space $\mathcal{F}$ using a mapping $\Phi : \mathcal{X} \rightarrow \mathcal{F}$, and then classification, regression, or clustering is performed in $\mathcal{F}$ using traditional methods such as linear SVMs, GPs, or PCA. If $\mathcal{F}$ is chosen to be a dot product space and if one defines the kernel matrix, also known as the Gram matrix, $G \in \mathbb{R}^{n \times n}$ as $G_{ij} = k(x_i, x_j) = (\Phi(x_i), \Phi(x_j))$, then any algorithm whose operations can be expressed in the input space in terms of dot products can be generalized to an algorithm which operates in the feature space by substituting a kernel function for the inner product. In practice, this means presenting the Gram matrix $G$ in place of the input covariance matrix $X^T X$. Relatedly, using the kernel $k$ instead of a dot product in the input space corresponds to mapping the data set into a (usually) high-dimensional dot product space $\mathcal{F}$ by a (usually nonlinear) mapping $\Phi : \mathbb{R}^m \rightarrow \mathcal{F}$, and taking dot products there, i.e., $k(x_i, x_j) = (\Phi(x_i), \Phi(x_j))$. Note that for the commonly-used Mercer kernels, $G$ is a symmetric positive semidefinite (SPSD) matrix.

The generality of this framework should be emphasized. For example, there has been much work recently on dimensionality reduction for nonlinear manifolds in high-dimensional spaces. See, e.g., Isomap, local linear embedding, and graph Laplacian eigenmap [29, 26, 4] as well as Hessian eigenmaps and semidefinite embedding [9, 30]. These methods first induce a local neighborhood structure on the data and then use this local structure to find a global embedding of the manifold in a lower dimensional space. The manner in which these different algorithms use the local information to construct the global embedding is quite