LARGE-SCALE APPROXIMATE KERNEL CANONICAL CORRELATION ANALYSIS

Weiran Wang & **Karen Livescu**
Toyota Technological Institute at Chicago
6045 S. Kenwood Ave., Chicago, IL 60637
Email: {weiranwang,klivescu}@ttic.edu

ABSTRACT

Kernel canonical correlation analysis (KCCA) is a nonlinear multi-view representation learning technique with broad applicability in statistics and machine learning. Although there is a closed-form solution for the KCCA objective, it involves solving an $N \times N$ eigenvalue system where N is the training set size, making its computational requirements in both memory and time prohibitive for large-scale problems. Various approximation techniques have been developed for KCCA. A commonly used approach is to first transform the original inputs to an M-dimensional random feature space so that inner products in the feature space approximate kernel evaluations, and then apply linear CCA to the transformed inputs. In many applications, however, the dimensionality M of the random feature space may need to be very large in order to obtain a sufficiently good approximation; it then becomes challenging to perform the linear CCA step on the resulting very high-dimensional data matrices. We show how to use a stochastic optimization algorithm, recently proposed for linear CCA and its neural-network extension, to further alleviate the computation requirements of approximate KCCA. This approach allows us to run approximate KCCA on a speech dataset with 1.4 million training samples and a random feature space of dimensionality $M = 100000$ on a typical workstation.

1 INTRODUCTION

Canonical correlation analysis (CCA, Hotelling, 1936) and its extensions are ubiquitous techniques in scientific research areas for revealing the common sources of variability in multiple views of the same phenomenon, including meteorology (Anderson, 2003), chemometrics (Montanarella et al., 1995), genomics (Witten et al., 2009), computer vision (Kim et al., 2007; Socher & Li, 2010), speech recognition (Rudzicz, 2010; Arora & Livescu, 2013; Wang et al., 2015a), and natural language processing (Vinokourov et al., 2003; Haghighi et al., 2008; Dhillon et al., 2011; Hodosh et al., 2013; Faruqui & Dyer, 2014; Lu et al., 2015a). CCA seeks linear projections of two random vectors (views), such that the resulting low-dimensional vectors are maximally correlated. Given a dataset of N pairs of observations $(\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_N, \mathbf{y}_N)$ of the random variables, where $\mathbf{x}_i \in \mathbb{R}^{d_x}$ and $\mathbf{y}_i \in \mathbb{R}^{d_y}$ for $i = 1, \dots, N$, the objective of CCA for L-dimensional projections can be written as[1] (see, e.g., Borga, 2001)

$$\max_{\mathbf{U} \in \mathbb{R}^{d_x \times L}, \mathbf{V} \in \mathbb{R}^{d_y \times L}} \quad \operatorname{tr}\left(\mathbf{U}^\top \mathbf{\Sigma}_{xy} \mathbf{V}\right) \tag{1}$$
$$\text{s.t.} \quad \mathbf{U}^\top \mathbf{\Sigma}_{xx} \mathbf{U} = \mathbf{V}^\top \mathbf{\Sigma}_{yy} \mathbf{V} = \mathbf{I}, \qquad \mathbf{u}_i^\top \mathbf{\Sigma}_{xy} \mathbf{v}_j = 0, \text{ for } i \neq j,$$

where $(\mathbf{U}, \mathbf{V})$ are the projection matrices for each view, $\mathbf{\Sigma}_{xy} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i \mathbf{y}_i^\top$, $\mathbf{\Sigma}_{xx} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i \mathbf{x}_i^\top + r_x \mathbf{I}$, $\mathbf{\Sigma}_{yy} = \frac{1}{N} \sum_{i=1}^N \mathbf{y}_i \mathbf{y}_i^\top + r_y \mathbf{I}$ are the cross- and auto-covariance matrices, and $(r_x, r_y) \geq 0$ are regularization parameters (Vinod, 1976; Bie & Moor, 2003). There exists a

[1] In this paper, we assume that the inputs are centered at the origin for notational simplicity; if they are not, we can center them as a pre-processing operation.

closed-form solution to (1) as follows. Let the rank-L singular value decomposition (SVD) of the whitened covariance matrix $\mathbf{T} = \mathbf{\Sigma}_{xx}^{-\frac{1}{2}} \mathbf{\Sigma}_{xy} \mathbf{\Sigma}_{yy}^{-\frac{1}{2}} \in \mathbb{R}^{d_x \times d_y}$ be $\tilde{\mathbf{U}} \Lambda \tilde{\mathbf{V}}^\top$, where Λ contains the top L singular values $\sigma_1 \geq \cdots \geq \sigma_L$ on its diagonal, and $(\tilde{\mathbf{U}}, \tilde{\mathbf{V}})$ are the corresponding singular vectors. Then the optimal projection matrices $(\mathbf{U}, \mathbf{V})$ in (1) are $(\mathbf{\Sigma}_{xx}^{-\frac{1}{2}} \tilde{\mathbf{U}}, \mathbf{\Sigma}_{yy}^{-\frac{1}{2}} \tilde{\mathbf{V}})$, and the optimal objective value, referred to as the canonical correlation, is $\sum_{l=1}^{L} \sigma_l$.[2] The theoretical properties of CCA (Kakade & Foster, 2007; Chaudhuri et al., 2009; Foster et al., 2009) and its connection to other methods (Borga, 2001; Bach & Jordan, 2005; Chechik et al., 2005) have also been studied.

One limitation of CCA is its restriction to linear mappings, which are often insufficient to reveal the highly nonlinear relationships in many real-world applications. To overcome this issue, kernel CCA (KCCA) was proposed indepedently by several researchers (Lai & Fyfe, 2000; Akaho, 2001; Melzer et al., 2001) and has become a common technique in statistics and machine learning (Bach & Jordan, 2002; Hardoon et al., 2004). KCCA extends CCA by mapping the original inputs in both views into reproducing kernel Hilbert spaces (RKHS) and solving linear CCA in the RKHS. By the representer theorem of RKHS (Schölkopf & Smola, 2001), one can conveniently work with the kernel functions instead of the high-dimensional (possibly infinite-dimensional) RKHS, and the projection mapping is a linear combination of kernel functions evaluated at the training samples. KCCA has been successfully used for cross-modality retrieval (Hardoon et al., 2004; Li & Shawe-Taylor, 2005; Socher & Li, 2010; Hodosh et al., 2013), acoustic feature learning (Arora & Livescu, 2013), computational biology (Yamanishi et al., 2004; Hardoon et al., 2007; Blaschkoa et al., 2011), and statistical independence measurement (Bach & Jordan, 2002; Fukumizu et al., 2007; Lopez-Paz et al., 2013).

KCCA also has a closed-form solution, via an $N \times N$ eigenvalue system (see Sec. 2). However, this solution does not scale up to datasets of more than a few thousand training samples, due to the time complexity of solving the eigenvalue system ($\mathcal{O}(N^3)$ for a naive solution) and the memory cost of storing the kernel matrices. As a result, various approximation techniques have been developed, most of which are based on low-rank approximations of the kernel matrices. With rank-M approximations of the kernel matrices, the cost of solving approximate KCCA reduces to $\mathcal{O}(M^2 N)$ (see, e.g., Bach & Jordan, 2002; Lopez-Paz et al., 2014). Thus if $M \ll N$, the approximation leads to significant computational savings. Typically, ranks of a few hundred to a few thousand are used for the low-rank kernel approximations (Yang et al., 2012; Le et al., 2013; Lopez-Paz et al., 2014). In more challenging real-world applications, however, it is observed that the rank M needed for an approximate kernel method to work well can be quite large, on the order of tens or hundreds of thousands (see Huang et al., 2014; Lu et al., 2015b for classification tasks, and Wang et al., 2015a for KCCA). In such scenarios, it then becomes challenging to solve even approximate KCCA.

In this paper, we focus on the computational challenges of scaling up approximate kernel CCA using low-rank kernel approximations when both the training set size N and the approximation rank M are large. The particular variant of approximate KCCA we use, called randomized CCA (Lopez-Paz et al., 2014), transforms the original inputs to an M-dimensional feature space using random features (Rahimi & Recht, 2008; 2009) so that inner products in the new feature space approximate the kernel function. This approach thus turns the original KCCA problem into a very high-dimensional linear CCA problem of the form (1). We then make use of a stochastic optimization algorithm, recently proposed for linear CCA and its deep neural network extension deep CCA (Ma et al., 2015; Wang et al., 2015c), to reduce the memory requirement for solving the resulting linear CCA problem. This algorithm updates parameters iteratively based on small minibatches of training samples. This approach allows us to run approximate KCCA on an $8-$million sample dataset of MNIST digits, and on a speech dataset with 1.4 million training samples and rank (dimensionality of random feature space) $M = 100000$ on a normal workstation. Using this approach we achieve encouraging results for multi-view learning of acoustic transformations for speech recognition.[3]

In the following sections we review approximate KCCA and random features (Sec. 2), present the stochastic optimization algorithm (Sec. 3), discuss related work (Sec. 4), and demonstrate our algorithm on two tasks (Sec. 5).

[2] Alternatively, one could also solve some equivalent $M \times M$ eigenvalue system instead of the SVD of $\mathbf{T}$, at a similar cost.

[3] Our MATLAB implementation is available at `http://ttic.uchicago.edu/~wwang5/knoi.html`

2 APPROXIMATE KCCA

2.1 KCCA SOLUTION

In KCCA, we transform the inputs $\{\mathbf{x}_i\}_{i=1}^N$ of view 1 and $\{\mathbf{y}_i\}_{i=1}^N$ of view 2 using feature mappings ϕ_x and ϕ_y associated with some positive semi-definite kernels k_x and k_y respectively, and then solve the linear CCA problem (1) for the feature-mapped inputs (Lai & Fyfe, 2000; Akaho, 2001; Melzer et al., 2001; Bach & Jordan, 2002; Hardoon et al., 2004). The key property of such kernels is that $k_x(\mathbf{x}, \mathbf{x}') =< \phi_x(\mathbf{x}), \phi_x(\mathbf{x}') >$ (similarly for view 2) Schölkopf & Smola, 2001. Even though the feature-mapped inputs live in possibly infinite-dimensional RKHS, replacing the original inputs $(\mathbf{x}_i, \mathbf{y}_i)$ with $(\phi_x(\mathbf{x}_i), \phi_y(\mathbf{y}_i))$ in (1), and using the KKT theorem (Nocedal & Wright, 2006), one can show that the solution has the form $\mathbf{U} = \sum_{i=1}^N \phi_x(\mathbf{x}_i)\boldsymbol{\alpha}_i^\top$ and $\mathbf{V} = \sum_{i=1}^N \phi_y(\mathbf{y}_i)\boldsymbol{\beta}_i^\top$ where $\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \in \mathbb{R}^L$, $i = 1, \ldots, N$, as a result of the representer theorem (Schölkopf & Smola, 2001). The final KCCA projections can therefore be written as $\mathbf{f}(\mathbf{x}) = \sum_{i=1}^N \boldsymbol{\alpha}_i k_x(\mathbf{x}, \mathbf{x}_i) \in \mathbb{R}^L$ and $\mathbf{g}(\mathbf{y}) = \sum_{i=1}^N \boldsymbol{\beta}_i k_y(\mathbf{y}, \mathbf{y}_i) \in \mathbb{R}^L$ for view 1 and view 2 respectively.

Denote by $\mathbf{K}_x$ the $N \times N$ kernel matrix for view 1, i.e., $(\mathbf{K}_x)_{ij} = k_x(\mathbf{x}_i, \mathbf{x}_j)$, and similarly denote by $\mathbf{K}_y$ the kernel matrix for view 2. Then (1) can be written as a problem in the coefficient matrices $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_N]^\top \in \mathbb{R}^{N \times L}$ and $\mathbf{B} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_N]^\top \in \mathbb{R}^{N \times L}$. One can show that the optimal coefficients $\mathbf{A}$ correspond to the top L eigenvectors of the $N \times N$ matrix $(\mathbf{K}_x + Nr_x\mathbf{I})^{-1}\mathbf{K}_y(\mathbf{K}_y + Nr_y\mathbf{I})^{-1}\mathbf{K}_x$, and a similar result holds for $\mathbf{B}$ (see, e.g., Hardoon et al., 2004). This involves solving an eiaylorgenvalue problem of size $N \times N$, which is expensive both in memory (storing the kernel matrices) and in time (solving the $N \times N$ eigenvalue systems naively costs $\mathcal{O}(N^3)$).

Various kernel approximation techniques have been proposed to scale up KCCA, including Cholesky decomposition (Bach & Jordan, 2002), partial Gram-Schmidt (Hardoon et al., 2004), and incremental SVD (Arora & Livescu, 2012). Another widely used approximation technique for kernel matrices is the Nyström method (Williams & Seeger, 2001). In the Nyström method, we select M (random or otherwise) training samples $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_M$ and construct the $M \times M$ kernel matrix $\tilde{\mathbf{K}}_x$ based on these samples, i.e. $(\tilde{\mathbf{K}}_x)_{ij} = k_x(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$. We compute the eigenvalue decomposition $\tilde{\mathbf{K}}_x = \tilde{\mathbf{R}}\tilde{\mathbf{\Lambda}}\tilde{\mathbf{R}}^\top$, and then the $N \times N$ kernel matrix for the entire training set can be approximated as $\mathbf{K}_x \approx \mathbf{C}\tilde{\mathbf{K}}_x^{-1}\mathbf{C}^\top$ where $\mathbf{C}$ contains the columns of $\mathbf{K}_x$ corresponding to the selected subset, i.e., $\mathbf{C}_{ij} = k_x(\mathbf{x}_i, \tilde{\mathbf{x}}_j)$. This means $\mathbf{K}_x \approx (\mathbf{C}\tilde{\mathbf{R}}\tilde{\mathbf{\Lambda}}^{-\frac{1}{2}})(\mathbf{C}\tilde{\mathbf{R}}\tilde{\mathbf{\Lambda}}^{-\frac{1}{2}})^\top$, so we can use the $M \times N$ matrix $(\mathbf{C}\tilde{\mathbf{R}}\tilde{\mathbf{\Lambda}}^{-\frac{1}{2}})^\top$ as the new feature representation for view 1 (similarly for view 2), where inner products between samples approximate kernel similarities. We can extract such features for both views, and apply linear CCA to them to approximate the KCCA solution (Yang et al., 2012; Lopez-Paz et al., 2014). Notice that using the Nyström method has a time complexity (for view 1) of $\mathcal{O}(M^2 d_x + M^3 + NMd_x + M^2 N)$, where the four terms account for the costs of forming $\tilde{\mathbf{K}}_x \in \mathbb{R}^{M \times M}$, computing the eigenvalue decomposition of $\tilde{\mathbf{K}}_x$, forming $\mathbf{C}$, and computing $\mathbf{C}\tilde{\mathbf{R}}\tilde{\mathbf{\Lambda}}^{-\frac{1}{2}}$, respectively, and a space complexity of $\mathcal{O}(M^2)$ for saving the eigenvalue systems of $\tilde{\mathbf{K}}_x$ and $\tilde{\mathbf{K}}_y$, which are expensive for large M. Although there have been various sampling/approximation strategies for the Nyström method (Li et al., 2010; Zhang & Kwok, 2009; 2010; Kumar et al., 2012; Gittens & Mahoney, 2013), their constructions are more involved.

2.2 APPROXIMATION VIA RANDOM FEATURES

We now describe another approximate KCCA formulation that is particularly well-suited to large-scale problems.

It is known from harmonic analysis that a shift-invariant kernel of the form $k(\mathbf{x}, \mathbf{x}') = \kappa(\mathbf{x} - \mathbf{x}')$ is a positive definite kernel if and only if $\kappa(\Delta)$ is the Fourier transform of a non-negative measure (this is known as Bochner's theorem; see Rudin, 1994; Rahimi & Recht, 2008). Thus we can write the kernel function as an expectation over sinusoidal functions over the underlying probability measures and approximate it with sample averages. Taking as a concrete example the Gaussian radial basis function (RBF) kernel $k(\mathbf{x}, \mathbf{x}') = e^{-\|\mathbf{x}-\mathbf{x}'\|^2/2s^2}$ where s is the kernel width, it can be approximated

as (Lopez-Paz et al., 2014)

$$k(\mathbf{x}, \mathbf{x}') = \int e^{-j\mathbf{w}^\top(\mathbf{x}-\mathbf{x}')} p(\mathbf{w})\, d\mathbf{w} \approx \frac{1}{M} \sum_{i=1}^{M} 2\cos(\mathbf{w}_i^\top \mathbf{x} + b_i)\cos(\mathbf{w}_i^\top \mathbf{x}' + b_i),$$

where $p(\mathbf{w})$ is the multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \frac{1}{s^2}\mathbf{I})$, obtained from the inverse Fourier transform of $\kappa(\Delta) = e^{-\frac{\|\Delta\|^2}{2s^2}}$, and b_i is drawn from a uniform distribution over $[0, 2\pi]$. Approximations for other shift-invariant kernels (Laplacian, Cauchy) can be found in Rahimi & Recht (2008). This approach has been extended to other types of kernels (Kar & Karnick, 2012; Hamid et al., 2014; Pennington et al., 2015). A careful quasi-Monte Carlo scheme for sampling from $p(\mathbf{w})$ (Yang et al., 2014), and structured feature transformation for accelerating the computation of $\mathbf{w}_i^\top \mathbf{x}$ (Le et al., 2013), have also been studied.

Leveraging this result, Rahimi & Recht (2009) propose to first extract M-dimensional *random Fourier features* for input $\mathbf{x}$ as (with slight abuse of notation)

$$\phi(\mathbf{x}) = \sqrt{\frac{2}{M}} \left[\cos(\mathbf{w}_1^\top \mathbf{x} + b_1), \ldots, \cos(\mathbf{w}_M^\top \mathbf{x} + b_M)\right] \in \mathbb{R}^M,$$

so that $\phi(\mathbf{x})^\top \phi(\mathbf{x}') \approx k(\mathbf{x}, \mathbf{x}')$, and then apply linear methods on these features. The computational advantage of this approach is that it turns nonlinear learning problems into convex linear learning problems, for which empirical risk minimization is much more efficient (e.g, Lu et al., 2015b used the recently proposed stochastic gradient method by Roux et al. 2012 for the task of multinomial logistic regression with random Fourier features). Rahimi & Recht (2009) showed that it allows us to effectively learn nonlinear models and still obtain good learning guarantees.

Lopez-Paz et al. (2014) have recently applied the random feature idea to KCCA, by extracting M-dimensional random Fourier features $\{(\phi_x(\mathbf{x}_i), \phi_y(\mathbf{y}_i))\}_{i=1}^N$ for both views and solving exactly a linear CCA on the transformed pairs. They also provide an approximation guarantee for this approach (see Theorem 4 of Lopez-Paz et al., 2014). Comparing random features with the Nyström method described previously, when both techniques use rank-M approximations, the cost of computing the solution to (1) is the same and of order $\mathcal{O}(M^2 N)$. But using random features, we generate the M-dimensional features in a data-independent fashion with a minimal cost $\mathcal{O}(NMd_x)$ (for view 1), which is negligible compared to that of the Nyström method. Furthermore, random features do not require saving any kernel matrix and the random features can be generated on the fly by saving the random seeds. Although the Nyström approximation can be more accurate at the same rank (Yang et al., 2012), the computational efficiency and smaller memory cost of random features make them more appealing for large-scale problems in practice.

3 Stochastic optimization of approximate KCCA

When the dimensionality M of the random Fourier features is very large, solving the resulting linear CCA problem is still very costly as one needs to save the $M \times M$ matrix $\tilde{\mathbf{T}} = \tilde{\mathbf{\Sigma}}_{xx}^{-\frac{1}{2}} \tilde{\mathbf{\Sigma}}_{xy} \tilde{\mathbf{\Sigma}}_{yy}^{-\frac{1}{2}}$ and compute its SVD, where the covariance matrices are now computed on $\{(\phi_x(\mathbf{x}_i), \phi_y(\mathbf{y}_i))\}_{i=1}^N$ instead of $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$. It is thus desirable to develop memory-efficient stochastic optimization algorithms for CCA, where each update of the projection mappings depends only on a small mini-batch of b examples, thus reducing the memory cost to $\mathcal{O}(bM)$. Notice, however, in contrast to the classification or regression objectives that are more commonly used with random Fourier features (Rahimi & Recht, 2009; Huang et al., 2014; Lu et al., 2015b), the CCA objective (1) can not be written as an unconstrained sum or expectation of losses incurred at each training sample (in fact all training samples are coupled together through the constraints). As a result, stochastic gradient descent, which requires unbiased gradient estimates computed from small minibatches, is not directly applicable here.

Fortunately, Ma et al. (2015); Wang et al. (2015c) have developed stochastic optimization algorithms, referred to as AppGrad (Augmented Approximate Gradient) and NOI (Nonlinear Orthogonal Iterations) respectively, for linear CCA and its deep neural network extension deep CCA (Andrew et al., 2013). Their algorithms are essentially equivalent other than the introduction in (Wang et al., 2015c) of a time constant for smoothing the covariance estimates over time.

Algorithm 1 KNOI: Stochastic optimization for approximate KCCA.

Input: Initialization $\mathbf{U} \in \mathbb{R}^{M\times L}, \mathbf{V} \in \mathbb{R}^{M\times L}$, time constant ρ, minibatch size b, learning rate η, momentum μ.

$\Delta_{\mathbf{U}} \leftarrow \mathbf{0}, \quad \Delta_{\mathbf{V}} \leftarrow \mathbf{0}$

Randomly choose a minibatch $(\mathbf{X}_{b_0}, \mathbf{Y}_{b_0})$

$\mathbf{S}_{xx} \leftarrow \frac{1}{|b_0|}\sum_{i\in b_0}\left(\mathbf{U}^\top\phi_x(\mathbf{x}_i)\right)\left(\mathbf{U}^\top\phi_x(\mathbf{x}_i)\right)^\top$,

$\mathbf{S}_{yy} \leftarrow \frac{1}{|b_0|}\sum_{i\in b_0}\left(\mathbf{V}^\top\phi_y(\mathbf{y}_i)\right)\left(\mathbf{V}^\top\phi_y(\mathbf{y}_i)\right)^\top$

for $t = 1, 2, \ldots, T$ **do**

Randomly choose a minibatch $(\mathbf{X}_{b_t}, \mathbf{Y}_{b_t})$ of size b

$\mathbf{S}_{xx} \leftarrow \rho\mathbf{S}_{xx} + (1-\rho)\frac{1}{b}\sum_{i\in b_t}\left(\mathbf{U}^\top\phi_x(\mathbf{x}_i)\right)\left(\mathbf{U}^\top\phi_x(\mathbf{x}_i)\right)^\top$

$\mathbf{S}_{yy} \leftarrow \rho\mathbf{S}_{yy} + (1-\rho)\frac{1}{b}\sum_{i\in b_t}\left(\mathbf{V}^\top\phi_y(\mathbf{y}_i)\right)\left(\mathbf{V}^\top\phi_y(\mathbf{y}_i)\right)^\top$

Compute the gradient $\partial\mathbf{U}$ of the objective

$$\min_{\mathbf{U}} \ \frac{1}{b}\sum_{i\in b_t}\left\|\mathbf{U}^\top\phi_x(\mathbf{x}_i) - \mathbf{S}_{yy}^{-\frac{1}{2}}\mathbf{V}^\top\phi_y(\mathbf{y}_i)\right\|^2$$

as $\partial\mathbf{U} \leftarrow \frac{1}{b}\sum_{i\in b_t}\phi_x(\mathbf{x}_i)\left(\mathbf{U}^\top\phi_x(\mathbf{x}_i) - \mathbf{S}_{yy}^{-\frac{1}{2}}\mathbf{V}^\top\phi_y(\mathbf{y}_i)\right)^\top$

Compute the gradient $\partial\mathbf{V}$ of the objective

$$\min_{\mathbf{V}} \ \frac{1}{b}\sum_{i\in b_t}\left\|\mathbf{V}^\top\phi_y(\mathbf{y}_i) - \mathbf{S}_{xx}^{-\frac{1}{2}}\mathbf{U}^\top\phi_x(\mathbf{x}_i)\right\|^2$$

as $\partial\mathbf{V} \leftarrow \frac{1}{b}\sum_{i\in b_t}\phi_y(\mathbf{y}_i)\left(\mathbf{V}^\top\phi_y(\mathbf{y}_i) - \mathbf{S}_{xx}^{-\frac{1}{2}}\mathbf{U}^\top\phi_x(\mathbf{x}_i)\right)^\top$

$\Delta_{\mathbf{U}} \leftarrow \mu\Delta_{\mathbf{U}} - \eta\partial\mathbf{U}, \quad \Delta_{\mathbf{V}} \leftarrow \mu\Delta_{\mathbf{V}} - \eta\partial\mathbf{V}$

$\mathbf{U} \leftarrow \mathbf{U} + \Delta_{\mathbf{U}}, \quad \mathbf{V} \leftarrow \mathbf{V} + \Delta\mathbf{V}$

end for

Output: The updated $(\mathbf{U}, \mathbf{V})$.

The idea originates from the *alternating least squares (ALS)* formulation of CCA (Golub & Zha, 1995; Lu & Foster, 2014), which computes the SVD of $\mathbf{T}$ using orthogonal iterations (a generalization of power iterations to multiple eigenvalues/eigenvectors, Golub & van Loan, 1996) on $\mathbf{T}\mathbf{T}^\top$ and $\mathbf{T}^\top\mathbf{T}$. Due to the special form of $\mathbf{T}\mathbf{T}^\top$ and $\mathbf{T}^\top\mathbf{T}$, two least squares problems arise in this iterative approach (see, e.g., Wang et al., 2015c, Section III. A for more details). With this observation, Lu & Foster (2014) solve these least squares problems using randomized PCA (Halko et al., 2011) and a batch gradient algorithm. Ma et al. (2015); Wang et al. (2015c) take a step further and replace the exact solutions to the least squares problems with efficient stochastic gradient descent updates. Although unbiased gradient estimates of these subproblems do not lead to unbiased gradient estimates of the original CCA objective, local convergence results (that the optimum of CCA is a fixed point of AppGrad, and the AppGrad iterate converges linearly to the optimal solution when started in its neighborhood) have been established for AppGrad (Ma et al., 2015). It has also been observed that the stochastic algorithms converge fast to approximate solutions that are on par with the exact solution or solutions by batch-based optimizers.

We give our stochastic optimization algorithm for approximate KCCA, named KNOI (Kernel Nonlinear Orthogonal Iterations), in Algorithm 1. Our algorithm is adapted from the NOI algorithm of Wang et al. (2015c), which allows the use of smaller minibatches (through the time constant ρ) than does the AppGrad algorithm of Ma et al. (2015). In each iteration, KNOI adaptively estimates the covariance of the projections of each view ($\in \mathbb{R}^L$) using a convex combination (with $\rho \in [0, 1)$) of the previous estimate and the estimate based on the current minibatch,[4] uses them to whiten the targets of the cross-view least squares regression problems, derives gradients from these problems,[5] and finally updates the projection matrices $(\mathbf{U}, \mathbf{V})$ with momentum. Notice that ρ controls how fast we forget the previous estimate; larger ρ may be necessary for the algorithm to work well if the mini-

[4] In practice we also adaptively estimate the mean of the projections and center each minibatch.

[5] We also use small weight decay regularization ($\sim 10^{-5}$) for $(\mathbf{U}, \mathbf{V})$ in the least squares problems.

batch size b is small (e.g., due to memory constraints), in which case the covariance estimates based on the current minibatch are noisier (see discussions in Wang et al., 2015c). Empirically, we find that using momentum $\mu \in [0, 1)$ helps the algorithm to make rapid progress in the objective with a few passes over the training set, as observed by the deep learning community (Sutskever et al., 2013). Although we have specifically use random Fourier features in Algorithm 1, in principle other low-rank kernel approximations can be used as well.

In each iteration of KNOI, the main cost comes from evaluating the random Fourier features and the projections for a minibatch, and computing the gradients. Since we usually look for low-dimensional projections (L is small), it costs little memory and time to compute the covariance estimates $\mathbf{S}_{xx}$ and $\mathbf{S}_{yy}$ (of size $L \times L$) and their eigenvalue decompositions (for $\mathbf{S}_{xx}^{-\frac{1}{2}}$ and $\mathbf{S}_{yy}^{-\frac{1}{2}}$). Overall, KNOI has a memory complexity of $\mathcal{O}(Mb)$ (excluding the $\mathcal{O}(ML)$ cost for saving $\mathbf{U}$ and $\mathbf{V}$ in memory) and a time complexity of $\mathcal{O}(bM(d_x + d_y + 4L))$ per iteration.

The $(\mathbf{U}, \mathbf{V})$ we obtain from Algorithm 1 do not satisfy the constraints $\mathbf{U}^\top \tilde{\mathbf{\Sigma}}_{xx} \mathbf{U} = \mathbf{V}^\top \tilde{\mathbf{\Sigma}}_{yy} \mathbf{V}$; one can enforce the constraints via another linear CCA in $\mathbb{R}^L$ on $\{(\mathbf{U}^\top \phi_x(\mathbf{x}_i), \mathbf{V}^\top \phi_y(\mathbf{y}_i))\}_{i=1}^N$, which does not change the canonical correlation between the projections. To evaluate the projection of a view 1 test sample $\mathbf{x}$, we generate the random Fourier features $\phi_x(\mathbf{x})$ using the same random seed for the training set, and compute $\mathbf{U}^\top \phi_x(\mathbf{x})$ for it.

Finally, we comment on the choice of hyperparameters in KNOI. Empirically, we find that larger b tends to give more rapid progress in the training objective, in which case ρ can be set to small values or to 0 as there is sufficient covariance information in a large minibatch (also shown by Ma et al., 2015; Wang et al., 2015c). Therefore, we recommend using larger b and $\rho = 0$ if one can afford the memory cost. For large-scale problems with millions of training examples, we set b to be a small portion of the training set (a few thousands) and enjoy the fast convergence of stochastic training algorithms (Bottou & Bousquet, 2008). In our experiments we initialize $(\mathbf{U}, \mathbf{V})$ with values sampled from a Gaussian distribution with standard deviation 0.1, and tune the learning rate η and momentum μ on small grids.

4 Related work

There have been continuous efforts to scale up classical methods such as principal component analysis and partial least squares with stochastic/online updates (Krasulina, 1969; Oja & Karhunen, 1985; Warmuth & Kuzmin, 2008; Arora et al., 2012; 2013; Mitliagkas et al., 2013; Balsubramani et al., 2013; Shamir, 2015; Xie et al., 2015). The CCA objective is more challenging due to the constraints, as also pointed out by Arora et al. (2012).

Avron et al. (2013) propose an algorithm for selecting a subset of training samples that retain the most information for accelerating linear CCA, when there are many more training samples (large N) than features (small M in our case). While this approach effectively reduces the training set size N, it provides no remedy for the large M scenario we face in approximate KCCA.

In terms of online/stochastic CCA, Yger et al. (2012) propose an adaptive CCA algorithm with efficient online updates based on matrix manifolds defined by the constraints (and they use a similar form of adaptive estimates for the covariance matrices). However, the goal of their algorithm is anomaly detection for streaming data with a varying distribution, rather than to perform CCA for a given dataset. Regarding the stochastic CCA algorithms of Ma et al. (2015); Wang et al. (2015c) we use here, an intuitively similar approach is proposed in the context of alternating conditional expectation (Makur et al., 2015).

Another related approach is that of Xie et al. (2015), who propose the Doubly Stochastic Gradient Descent (DSGD) algorithm for approximate kernel machines (including KCCA) based on random Fourier features. KNOI and DSGD are different in several respects. First, the stochastic update rule of DSGD for $(\mathbf{U}, \mathbf{V})$ is derived from the Lagrangian of an eigenvalue formulation of CCA and is different from ours, e.g., DSGD does not have any whitening steps while KNOI does. Second, DSGD gradually increases the number of random Fourier features (or cycles through blocks of random Fourier features) and updates the corresponding portions of $(\mathbf{U}, \mathbf{V})$ as it sees more training samples. While this potentially further reduces the memory cost of the algorithm, it is not essential as we could also process the random Fourier features in minibatches (blocks) within KNOI.

5 EXPERIMENTS

In this section, we demonstrate the KNOI algorithm on two large-scale problems and compare it to several alternatives:

- CCA, solved exactly by SVD.
- FKCCA, low-rank approximation of KCCA using random Fourier features, with the CCA step solved exactly by SVD.
- NKCCA, low-rank approximation of KCCA using the Nyström method, with the CCA step solved exactly by SVD.

We implement KNOI in MATLAB with GPU support. Since our algorithm mainly involves simple matrix operations, running it on a GPU provides significant speedup.

5.1 MNIST 8M

In the first set of experiments, we demonstrate the scability and efficiency of KNOI on the MNIST8M dataset (Loosli et al., 2007). The dataset consists of 8.1 million 28×28 grayscale images of the digits 0-9. We divide each image into the left and right halves and use them as the two views in KCCA, so the input dimensionality is 392 for both views. The dataset is randomly split into training/test sets of size 8M/0.1M. The task is to learn $L = 50$ dimensional projections using KCCA, and the evaluation criterion is the total canonical correlation achieved on the test set (upper-bounded by 50). The same task is used by Xie et al. (2015), although we use a different training/test split.

As in Xie et al. (2015), we fix the kernel widths using the "median" trick[6] for all algorithms. We vary the rank M for FKCCA and NKCCA from 256 to 6000. For comparison, we use the same hyperparamters as those of Xie et al. (2015)[7]: data minibatch size $b = 1024$, feature minibatch size 2048, total number of random Fourier features $M = 20480$,[8] and a decaying step size schedule. For KNOI, we tune hyperparameters on a rough grid based on total canonical correlation obtained on a random subset of the training set with 0.1M samples, and set the minibatch size $b = 2500$, time constant $\rho = 0$, learning rate $\eta = 0.01$, and momentum $\mu = 0.995$. We run the iterative algorithms DSGD and KNOI for one pass over the data, so that they see the same number of samples as FKCCA/NKCCA. We run each algorithm 5 times using different random seeds and report the mean results.

The total canonical correlations achieved by each algorithm on the test set, together with the run times measured on a workstation with 6 $3.6GHz$ CPUs and $64G$ main memory, are reported in Table 1. As expected, all algorithms improve monotonically as M is increased. FKCCA and NKCCA achieve competitive results with a reasonably large M, with NKCCA consistently outperforming FKCCA at the cost of longer run times. KNOI outperforms the other iterative algorithm DSGD, and overall achieves the highest canonical correlation with a larger M. We show the learning curve of KNOI with $M = 40960$ (on the test set) in Figure 1. We can see that KNOI achieves steep improvement in the objective in the beginning, and already outperforms the exact solutions of FKCCA and NKCCA with $M = 4096$ after seeing only 1/4 to 1/2 of the training set. We also run KNOI on an NVIDIA Tesla K40 GPU with 12G memory, and report the run times in parentheses in Table 1; the GPU provides a speedup of more than 12 times. For this large dataset, the KNOI algorithm itself requires less memory (less than 12G) than loading the training data in main memory ($\sim$25G).

5.2 X-RAY MICROBEAM SPEECH DATA

In the second set of experiments, we apply approximate KCCA to the task of learning acoustic features for automatic speech recognition. We use the Wisconsin X-ray microbeam (XRMB)

[6] Following Xie et al. (2015), kernel widths are estimated from the median of pairwise distances between 4000 randomly selected training samples.

[7] We thank the authors for providing their MATLAB implementation of DSGD.

[8] Xie et al. (2015) used a version of random Fourier features with both $\cos$ and $\sin$ functions, so the number of learnable parameters (in $\mathbf{U}$ and $\mathbf{V}$) of DSGD is twice that of KNOI for the same M.

Table 1: Total canonical correlation on MNIST 8M test set and the corresponding run times (with GPU run times in parentheses).

Method	M	Canon. Corr.	Time (minutes)
linear CCA		26.8	0.5
FKCCA	1024	33.5	9.8
	2048	37.6	24.4
	4096	40.7	68.9
	5000	41.4	79.4
	6000	42.1	107.5
NKCCA	1024	39.6	17.1
	2048	42.2	44.9
	4096	44.1	138.8
	5000	44.5	196.3
	6000	44.8	272.2
DSGD	20480	43.4	306.9
KNOI	20480	44.5	97.2 (7.6)
	40960	45.0	194.0 (15.4)
	100000	45.3	502.4 (39.7)

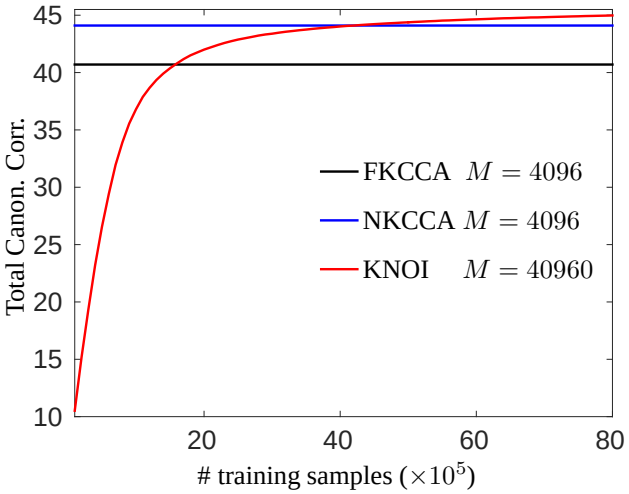

Figure 1: Learning curve of KNOI on the MNIST 8M test set. We show total canonical correlation in the 50-dimensional projections vs. the number of training samples ($\times 10^5$) processed by KNOI. The FKCCA and NKCCA values, always obtained using the entire training set, are shown as horizontal lines.

corpus (Westbury, 1994) of simultaneously recorded speech and articulatory measurements from 47 American English speakers. It has previously been shown that multi-view feature learning via CCA/KCCA greatly improves phonetic recognition performance given audio input alone (Arora & Livescu, 2013; Wang et al., 2015a;b).

We follow the setup of Wang et al. (2015a;b) and use the learned features (KCCA projections) for speaker-independent phonetic recognition.[9] The two input views are acoustic features (39D features consisting of mel frequency cepstral coefficients (MFCCs) and their first and second derivatives) and articulatory features (horizontal/vertical displacement of 8 pellets attached to several parts of the vocal tract) concatenated over a 7-frame window around each frame, giving 273D acoustic inputs and 112D articulatory inputs for each view. The XRMB speakers are split into disjoint sets of 35/8/2/2

[9]Unlike Wang et al. (2015a;b), who used the HTK toolkit (Young et al., 1999), we use the Kaldi speech recognition toolkit (Povey et al., 2011) for feature extraction and recognition with hidden Markov models. Our results therefore don't match those in Wang et al. (2015a;b) for the same types of features, but the relative merits of different types of features are consistent.

Table 2: Mean phone error rates (PER) over 6 folds obtained by each algorithm on the XRMB test speakers.

Method	Mean PER (%)
Baseline (MFCCs)	37.6
CCA	29.4
FKCCA ($M = 5000$)	28.1
FKCCA ($M = 30000$)	26.9
NKCCA ($M = 5000$)	28.0
KNOI ($M = 100000$)	26.4
DCCA	**25.4**

speakers for feature learning/recognizer training/tuning/testing. The 35 speakers for feature learning are fixed; the remaining 12 are used in a 6-fold experiment (recognizer training on 8 speakers, tuning on 2 speakers, and testing on the remaining 2 speakers). Each speaker has roughly $50K$ frames, giving 1.43M training frames for KCCA training. We remove the per-speaker mean and variance of the articulatory measurements for each training speaker. All of the learned feature types are used in a "tandem" speech recognition approach (Hermansky et al., 2000), i.e., they are appended to the original 39D features and used in a standard hidden Markov model (HMM)-based recognizer with Gaussian mixture observation distributions.

For each fold, we select the hyperparameters based on recognition accuracy on the tuning set. For each algorithm, the feature dimensionality L is tuned over $\{30, 50, 70\}$, and the kernel widths for each view are tuned by grid search. We initially set $M = 5000$ for FKCCA/NKCCA, and also test FKCCA at $M = 30000$ (the largest M at which we could afford to obtain an exact SVD solution on a workstation with 64G main memory) with kernel widths tuned at $M = 5000$; we could not obtain results for NKCCA with $M = 30000$ in 48 hours. For KNOI, we set $M = 100000$ and tune the optimization parameters on a rough grid. The tuned KNOI uses minibatch size $b = 2500$, time constant $\rho = 0$, fixed learning rate $\eta = 0.01$, and momentum $\mu = 0.995$. For this combination of b and M, we are able to run the algorithm on a Tesla K40 GPU (with 12G memory), and each epoch (one pass over the 1.43M training samples) takes only 7.3 minutes. We run KNOI for 5 epochs and use the resulting acoustic view projection for recognition. We have also tried to run KNOI for 10 epochs and the recognition performance does not change, even though the total canonical correlation keeps improving on both training and tuning sets.

For comparison, we report the performance of a baseline recognizer that uses only the original MFCC features, and the performance of deep CCA (DCCA) as described in Wang et al. (2015b), which uses 3 hidden layers of 1500 ReLU units followed by a linear output layer in the acoustic view, and only a linear output layer in the articulatory view. With this architecture, each epoch of DCCA takes about 8 minutes on a Tesla K40 GPU, on par with KNOI. Note that this DCCA architecture was tuned carefully for low PER rather than high canonical correlation. This architecture produces a total correlation of about 25 (out of a maximum of $L = 70$) on tuning data, while KNOI achieves 46.7. DCCA using deeper nonlinear networks for the second view can achieve even better total canonical correlation, but its PER performance then becomes significantly worse.

Phone error rates (PERs) obtained by different algorithms are given in Table 2, where smaller PER indicates better recognition performance. It is clear that all CCA-based features significantly improve over the baseline. Also, a large M is necessary for KCCA to be competitive with deep neural network methods, which is consistent with the findings of Huang et al. (2014); Lu et al. (2015b) when using random Fourier features for speech data (where the task is frame classification). Overall, KNOI outperforms the other approximate KCCA algorithms, although DCCA is still the best performer.

6 CONCLUSION

We have proposed kernel nonlinear orthogonal iterations (KNOI), a memory-efficient approximate KCCA algorithm based on random Fourier features and stochastic training of linear CCA. It scales

better to large data and outperforms previous approximate KCCA algorithms in both the objective values (total canonical correlation) and running times (with GPU support).

It is straightforward to incorporate in our algorithm the faster random features of Le et al. (2013) which can be generated (for view 1) in time $\mathcal{O}(NM \log d_x)$ instead of $\mathcal{O}(NMd_x)$, or the Taylor features of Cotter et al. (2011) which is preferable for sparse inputs, and random features for dot product or polynomial kernels (Kar & Karnick, 2012; Hamid et al., 2014; Pennington et al., 2015), which have proven to be useful for different domains. It is also worth exploring parallelization and multiple kernel learning strategies of Lu et al. (2015b) with random Fourier features to further bridge the gap between kernel methods and deep neural network methods.

Finally, as noted before, our algorithm does not use unbiased estimates of the gradient of the CCA objective. However, unbiased gradient estimates are not necessary for convergence of stochastic algorithms in general; a prominent example is the popular Oja's rule for stochastic PCA (see discussions in Balsubramani et al., 2013; Shamir, 2015). Deriving global convergence properties for our algorithm is a challenging topic and the subject of ongoing work.

ACKNOWLEDGEMENT

This research was supported by NSF grant IIS-1321015. The opinions expressed in this work are those of the authors and do not necessarily reflect the views of the funding agency. The Tesla K40 GPUs used for this research were donated by NVIDIA Corporation. We thank Bo Xie for providing his implementation of the doubly stochastic gradient algorithm for approximate KCCA, and Nati Srebro for helpful discussions.

REFERENCES

Akaho, Shotaro. A kernel method for canonical correlation analysis. In *Proceedings of the International Meeting of the Psychometric Society (IMPS2001)*, 2001.

Anderson, T. W. *An Introduction to Multivariate Statistical Analysis*. third edition, 2003.

Andrew, Galen, Arora, Raman, Bilmes, Jeff, and Livescu, Karen. Deep canonical correlation analysis. In *Proc. of the 30th Int. Conf. Machine Learning (ICML 2013)*, pp. 1247–1255, 2013.

Arora, Raman and Livescu, Karen. Kernel CCA for multi-view learning of acoustic features using articulatory measurements. In *Symposium on Machine Learning in Speech and Language Processing (MLSLP)*, 2012.

Arora, Raman and Livescu, Karen. Multi-view CCA-based acoustic features for phonetic recognition across speakers and domains. In *Proc. of the IEEE Int. Conf. Acoustics, Speech and Sig. Proc. (ICASSP'13)*, 2013.

Arora, Raman, Cotter, Andy, Livescu, Karen, and Srebro, Nati. Stochastic optimization for PCA and PLS. In *50th Annual Allerton Conference on Communication, Control, and Computing*, pp. 861–868, 2012.

Arora, Raman, Cotter, Andy, and Srebro, Nati. Stochastic optimization of PCA with capped MSG. In *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pp. 1815–1823, 2013.

Avron, Haim, Boutsidis, Christos, Toledo, Sivan, and Zouzias, Anastasios. Efficient dimensionality reduction for canonical correlation analysis. In *Proc. of the 30th Int. Conf. Machine Learning (ICML 2013)*, pp. 347–355, 2013.

Bach, Francis R. and Jordan, Michael I. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

Bach, Francis R. and Jordan, Michael I. A probabilistic interpretation of canonical correlation analysis. Technical Report 688, Dept. of Statistics, University of California, Berkeley, 2005.

Balsubramani, Akshay, Dasgupta, Sanjoy, and Freund, Yoav. The fast convergence of incremental PCA. In *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pp. 3174–3182, 2013.

Bie, Tijl De and Moor, Bart De. On the regularization of canonical correlation analysis. www.esat.kuleuven.ac.be/sista-cosic-docarch/, 2003. Int. Sympos. ICA and BSS.

Blaschkoa, Matthew B., Sheltonb, Jacquelyn A., Bartelsc, Andreas, Lamperte, Christoph H., and Gretton, Arthur. Semi-supervised kernel canonical correlation analysis with application to human fMRI. *Pattern Recognition Letters*, 32(11):1572–1583, 2011.

Borga, Magnus. Canonical correlation: A tutorial. 2001.

Bottou, Leon and Bousquet, Olivier. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, pp. 161–168, 2008.

Chaudhuri, Kamalika, Kakade, Sham M., Livescu, Karen, and Sridharan, Karthik. Multi-view clustering via canonical correlation analysis. In *Proc. of the 26th Int. Conf. Machine Learning (ICML'09)*, pp. 129–136, 2009.

Chechik, Gal, Globerson, Amir, Tishby, Naftali, and Weiss, Yair. Information bottleneck for Gaussian variables. *Journal of Machine Learning Research*, 6:165–188, 2005.

Cotter, Andrew, Keshet, Joseph, and Srebro, Nathan. Explicit approximations of the gaussian kernel. arXiv:1109.4603 [cs.AI], September 21 2011.

Dhillon, Paramveer, Foster, Dean, and Ungar, Lyle. Multi-view learning of word embeddings via CCA. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24, pp. 199–207, 2011.

Faruqui, Manaal and Dyer, Chris. Improving vector space word representations using multilingual correlation. In *Proceedings of EACL*, 2014.

Foster, Dean P., Johnson, Rie, Kakade, Sham M., and Zhang, Tong. Multi-view dimensionality reduction via canonical correlation analysis. 2009.

Fukumizu, Kenji, Bach, Francis R., and Gretton, Arthur. Statistical consistency of kernel canonical correlation analysis. *Journal of Machine Learning Research*, 8:361–383, 2007.

Gittens, Alex and Mahoney, Michael. Revisiting the nystrom method for improved large-scale machine learning. In *Proc. of the 30th Int. Conf. Machine Learning (ICML 2013)*, pp. 567–575, 2013.

Golub, Gene H. and van Loan, Charles F. *Matrix Computations*. third edition, 1996.

Golub, Gene H. and Zha, Hongyuan. *Linear Algebra for Signal Processing*, volume 69 of *The IMA Volumes in Mathematics and its Applications*, chapter The Canonical Correlations of Matrix Pairs and their Numerical Computation, pp. 27–49. 1995.

Haghighi, Aria, Liang, Percy, Berg-Kirkpatrick, Taylor, and Klein, Dan. Learning bilingual lexicons from monolingual corpora. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2008)*, pp. 771–779, 2008.

Halko, Nathan, Martinsson, Per-Gunnar, Shkolnisky, Yoel, and Tygert, Mark. An algorithm for the principal component analysis of large data sets. *SIAM J. Sci. Comput.*, 33(5):2580–2594, 2011.

Hamid, Raffay, Xiao, Ying, Gittens, Alex, and Decoste, Dennis. Compact random feature maps. In *Proc. of the 31st Int. Conf. Machine Learning (ICML 2014)*, pp. 19–27, 2014.

Hardoon, David R., Szedmak, Sandor, and Shawe-Taylor, John. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004.

Hardoon, David R., Mourão-Miranda, Janaina, Brammer, Michael, and Shawe-Taylor, John. Unsupervised analysis of fMRI data using kernel canonical correlation. *NeuroImage*, 37(4):1250–1259, 2007.

Hermansky, Hynek, Ellis, Daniel P. W., and Sharma, Sangita. Tandem connectionist feature extraction for conventional HMM systems. In *Proc. of the IEEE Int. Conf. Acoustics, Speech and Sig. Proc. (ICASSP'00)*, pp. 1635–1638, 2000.

Hodosh, Micah, Young, Peter, and Hockenmaier, Julia. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013.

Hotelling, Harold. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.

Huang, Po-Sen, Avron, Haim, Sainath, Tara, Sindhwani, Vikas, and Ramabhadran, Bhuvana. Kernel methods match deep neural networks on TIMIT: Scalable learning in high-dimensional random Fourier spaces. In *Proc. of the IEEE Int. Conf. Acoustics, Speech and Sig. Proc. (ICASSP'14)*, pp. 205–209, 2014.

Kakade, Sham M. and Foster, Dean P. Multi-view regression via canonical correlation analysis. In *Proc. of the 20th Annual Conference on Learning Theory (COLT'07)*, number 4539 in Lecture Notes in Computer Science, pp. 82–96, 2007.

Kar, Purushottam and Karnick, Harish. Random feature maps for dot product kernels. In *Proc. of the 15th Int. Workshop on Artificial Intelligence and Statistics (AISTATS 2012)*, pp. 583–591, 2012.

Kim, Tae-Kyun, Kittler, Josef, and Cipolla, Roberto. Discriminative learning and recognition of image set classes using canonical correlations. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(6):1005–1018, 2007.

Krasulina, T. P. A method of stochastic approximation for the determination of the least eigenvalue of a symmetric matrix. *USSR Computational Mathematics and Mathematical Physics*, 9(6):189–195, 1969.

Kumar, Sanjiv, Mohri, Mehryar, and Talwalkar, Ameet. Sampling methods for the nyström method. *Journal of Machine Learning Research*, 13:981–1006, 2012.

Lai, P. L. and Fyfe, C. Kernel and nonlinear canonical correlation analysis. *Int. J. Neural Syst.*, 10(5):365–377, 2000.

Le, Quoc, Sarlos, Tamas, and Smola, Alexander. Fastfood - computing Hilbert space expansions in loglinear time. In *Proc. of the 30th Int. Conf. Machine Learning (ICML 2013)*, pp. 244–252, 2013.

Li, M., Kwok, J. T., and Lu, B. Making large-scale Nyström approximation possible. In *Proc. of the 27th Int. Conf. Machine Learning (ICML 2010)*, pp. 631–638, 2010.

Li, Yaoyong and Shawe-Taylor, John. Using KCCA for japanese-english cross-language information retrieval and classification. *Journal of Intelligent Information Systems*, 27(2):117–133, 2005.

Loosli, Gaëlle, Canu, Stéphane, and Bottou, Léon. Training invariant support vector machines using selective sampling. In *Large Scale Kernel Machines*, pp. 301–320. 2007.

Lopez-Paz, David, Hennig, Philipp, and Schölkopf, Bernhard. The randomized dependence coefficient. In *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pp. 1–9, 2013.

Lopez-Paz, David, Sra, Suvrit, Smola, Alex, Ghahramani, Zoubin, and Schoelkopf, Bernhard. Randomized nonlinear component analysis. In *Proc. of the 31st Int. Conf. Machine Learning (ICML 2014)*, pp. 1359–1367, 2014.

Lu, Ang, Wang, Weiran, Bansal, Mohit, Gimpel, Kevin, and Livescu, Karen. Deep multilingual correlation for improved word embeddings. In *The 2015 Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT 2015)*, 2015a.

Lu, Yichao and Foster, Dean P. Large scale canonical correlation analysis with iterative least squares. In *Advances in Neural Information Processing Systems (NIPS)*, volume 27, pp. 91–99, 2014.

Lu, Zhiyun, May, Avner, Liu, Kuan, Garakani, Alireza Bagheri, Guo, Dong, Bellet, Aurélien, Fan, Linxi, Collins, Michael, Kingsbury, Brian, Picheny, Michael, and Sha, Fei. How to scale up kernel methods to be as good as deep neural nets. arXiv:1411.4000 [cs.LG], 2015b.

Ma, Zhuang, Lu, Yichao, and Foster, Dean. Finding linear structure in large datasets with scalable canonical correlation analysis. In *Proc. of the 32st Int. Conf. Machine Learning (ICML 2015)*, pp. 169–178, 2015.

Makur, Anuran, Kozynski, Fabían, Huang, Shao-Lun, and Zheng, Lizhong. An efficient algorithm for information decomposition and extraction. In *53nd Annual Allerton Conference on Communication, Control and Computing*, 2015.

Melzer, Thomas, Reiter, Michael, and Bischof, Horst. Nonlinear feature extraction using generalized canonical correlation analysis. In *Proc. of the 11th Int. Conf. Artificial Neural Networks (ICANN'01)*, pp. 353–360, 2001.

Mitliagkas, Ioannis, Caramanis, Constantine, and Jain, Prateek. Memory limited, streaming PCA. In *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pp. 2886–2894, 2013.

Montanarella, Luca, Bassani, Maria Rosa, and Bréas, Olivier. Chemometric classification of some European wines using pyrolysis mass spectrometry. *Rapid Communications in Mass Spectrometry*, 9(15):1589–1593, 1995.

Nocedal, Jorge and Wright, Stephen J. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. second edition, 2006.

Oja, Erkki and Karhunen, Juha. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *J. Math. Anal. Appl.*, 106(1):69–84, 1985.

Pennington, Jeffrey, Yu, Felix, and Kumar, Sanjiv. Spherical random features for polynomial kernels. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems (NIPS)*, volume 28, pp. 1837–1845. MIT Press, Cambridge, MA, 2015.

Povey, Daniel, Ghoshal, Arnab, Boulianne, Gilles, Burget, Lukas, Glembek, Ondrej, Goel, Nagendra, Hannemann, Mirko, Motlicek, Petr, Qian, Yanmin, Schwarz, Petr, Silovsky, Jan, Stemmer, Georg, and Vesely, Karel. The Kaldi speech recognition toolkit. In *Proc. of the 2011 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU 2011)*, 2011.

Rahimi, Ali and Recht, Ben. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, pp. 1177–1184, 2008.

Rahimi, Ali and Recht, Benjamin. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 21, pp. 1313–1320, 2009.

Roux, Nicolas Le, Schmidt, Mark, and Bach, Francis. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems (NIPS)*, volume 25, pp. 2672–2680, 2012.

Rudin, Walter. *Fourier Analysis on Groups*. 1994.

Rudzicz, Frank. Correcting errors in speech recognition with articulatory dynamics. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 60–68. Association for Computational Linguistics, 2010.

Schölkopf, Bernhard and Smola, Alexander J. *Learning with Kernels. Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning Series. 2001.

Shamir, Ohad. A stochastic PCA and SVD algorithm with an exponential convergence rate. In *Proc. of the 32st Int. Conf. Machine Learning (ICML 2015)*, pp. 144–152, 2015.

Socher, Richard and Li, Fei-Fei. Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora. In *Proc. of the 2010 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'10)*, pp. 966–973, 2010.

Sutskever, Ilya, Martens, James, Dahl, George, and Hinton, Geoffrey. On the importance of initialization and momentum in deep learning. In *Proc. of the 30th Int. Conf. Machine Learning (ICML 2013)*, pp. 1139–1147, 2013.

Vinod, H. D. Canonical ridge and econometrics of joint production. *Journal of Econometrics*, 4(2):147–166, 1976.

Vinokourov, Alexei, Cristianini, Nello, and Shawe-Taylor, John. Inferring a semantic representation of text via cross-language correlation analysis. In *Advances in Neural Information Processing Systems (NIPS)*, volume 15, pp. 1497–1504, 2003.

Wang, Weiran, Arora, Raman, Livescu, Karen, and Bilmes, Jeff. Unsupervised learning of acoustic features via deep canonical correlation analysis. In *Proc. of the IEEE Int. Conf. Acoustics, Speech and Sig. Proc. (ICASSP'15)*, 2015a.

Wang, Weiran, Arora, Raman, Livescu, Karen, and Bilmes, Jeff. On deep multi-view representation learning. In *Proc. of the 32st Int. Conf. Machine Learning (ICML 2015)*, pp. 1083–1092, 2015b.

Wang, Weiran, Arora, Raman, Srebro, Nati, and Livescu, Karen. Stochastic optimization for deep cca via nonlinear orthogonal iterations. In *53nd Annual Allerton Conference on Communication, Control and Computing*, 2015c.

Warmuth, Manfred K. and Kuzmin, Dima. Randomized online PCA algorithms with regret bounds that are logarithmic in the dimension. *Journal of Machine Learning Research*, 9:2287–2320, 2008.

Westbury, John R. *X-Ray Microbeam Speech Production Database User's Handbook Version 1.0*, 1994.

Williams, Christopher K. I. and Seeger, Matthias. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, volume 13, pp. 682–688, 2001.

Witten, Daniela M., Tibshirani, Robert, and Hastie, Trevor. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534, 2009.

Xie, Bo, Liang, Yingyu, and Song, Le. Scale up nonlinear component analysis with doubly stochastic gradients. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems (NIPS)*, volume 28, pp. 2332–2340. MIT Press, Cambridge, MA, 2015.

Yamanishi, Yoshihiro, Vert, Jean-Philippe, and Kanehisa, Minoru. *Kernel Methods in Computational Biology*, chapter Heterogeneous Data Comparison and Gene Selection with Kernel Canonical Correlation Analysis, pp. 209–229. MIT Press, 2004.

Yang, Jiyan, Sindhwani, Vikas, Avron, Haim, and Mahoney, Michael. Quasi-Monte Carlo feature maps for shift-invariant kernels. In *Proc. of the 31st Int. Conf. Machine Learning (ICML 2014)*, pp. 485–493, 2014.

Yang, Tianbao, Li, Yu-Feng, Mahdavi, Mehrdad, Jin, Rong, and Zhou, Zhi-Hua. Nyström method vs random Fourier features: A theoretical and empirical comparison. In *Advances in Neural Information Processing Systems (NIPS)*, volume 25, pp. 476–484, 2012.

Yger, Florian, Berar, Maxime, Gasso, Gilles, and Rakotomamonjy, Alain. Adaptive canonical correlation analysis based on matrix manifolds. In *Proc. of the 29th Int. Conf. Machine Learning (ICML 2012)*, pp. 1071–1078, 2012.

Young, Steve J., Kernshaw, Dan, Odell, Julian, Ollason, Dave, Valtchev, Valtcho, and Woodland, Phil. The HTK book version 2.2. Technical report, Entropic, Ltd., 1999.

Zhang, Kai and Kwok, James T. Density-weighted Nyström method for computing large kernel eigen-systems. *Neural Computation*, 21(1):121–146, 2009.

Zhang, Kai and Kwok, James T. Clustered Nyström method for large scale manifold learning and dimension reduction. *IEEE Trans. Neural Networks*, 21(10):1576–1587, 2010.