Deep Multi-View Learning via Task-Optimal CCA

Heather D. Couture
Pixel Scientia Labs, Raleigh, NC
heather@pixelscientia.com

Roland Kwitt
University of Salzburg, Austria
roland.kwitt@sbg.ac.at

J.S. Marron **Melissa Troester** **Charles M. Perou** **Marc Niethammer**
University of North Carolina at Chapel Hill
marron@unc.edu, troester@unc.edu, chuck_perou@med.unc.edu, mn@cs.unc.edu

Abstract

Canonical Correlation Analysis (CCA) is widely used for multimodal data analysis and, more recently, for discriminative tasks such as multi-view learning; however, it makes no use of class labels. Recent CCA methods have started to address this weakness but are limited in that they do not simultaneously optimize the CCA projection for discrimination and the CCA projection itself, or they are linear only. We address these deficiencies by simultaneously optimizing a CCA-based and a task objective in an end-to-end manner. Together, these two objectives learn a non-linear CCA projection to a shared latent space that is highly correlated and discriminative. Our method shows a significant improvement over previous state-of-the-art (including deep supervised approaches) for cross-view classification, regularization with a second view, and semi-supervised learning on real data.

1 Introduction

CCA is a popular data analysis technique that projects two data sources into a space in which they are maximally correlated [1, 2]. It was initially used for unsupervised data analysis to gain insights into components shared by the two sources [3–5]. CCA is also used to compute a shared latent space for cross-view classification [6, 4, 7, 8], for representation learning on multiple views that are then joined for prediction [9, 10], and for classification from a single view when a second view is available during training [11]. While some of the correlated CCA features are useful for discriminative tasks, many represent properties that are of no use for classification and obscure correlated information that is beneficial. This problem is magnified with recent non-linear extensions of CCA, implemented via neural networks (NNs), that make significant strides in improving correlation [3–5, 8] but often at the expense of discriminative capability (cf. §4.1). Therefore, we present a new deep learning technique to project the data from two views to a shared space that is also discriminative.

Most prior work that boosts the discriminative capability of CCA is *linear only* [12–14]. More recent work using NNs still remains limited in that it optimizes discriminative capability for an intermediate representation rather than the final CCA projection [10], or optimizes the CCA objective only during pre-training, not while training the task objective [15]. We advocate to jointly optimize CCA and a discriminative objective by computing the CCA projection within a network layer while applying a task-driven operation such as classification. Experimental results show that our method significantly improves upon previous work [10, 15] due to its focus on both the shared latent space and a task-driven objective. The latter is particularly important on small training set sizes.

While alternative approaches to multi-view learning via CCA exist, they typically focus on a reconstruction objective. That is, they transform the input into a shared space such that the input could be reconstructed – either individually, or reconstructing one view from the other. Variations of coupled

Preprint. Under review.

dictionary learning [16–19] and autoencoders [4, 20] have been used in this context. CCA-based objectives, such as the model used in this work, instead learn a transformation to a shared space without the need for reconstructing the input. This task may be easier and sufficient in producing a representation for multi-view classification [4]. We show that the CCA objective can equivalently be expressed as an ℓ_2 distance minimization in the shared space plus an orthogonality constraint. Orthogonality constraints help regularize NNs [21]; we present three techniques to accomplish this. While our method is derived from CCA, by manipulating the orthogonality constraints, we obtain deep CCA approaches that compute a shared latent space that is also discriminative.

Overall, our method enables end-to-end training via mini-batches, and we demonstrate the effectiveness of our model for three different tasks: 1) cross-view classification on a variation of MNIST [22] showing significant improvements in accuracy, 2) regularization when two views are available for training but only one at test time on a cancer imaging and genomic data set with only 1,000 samples, and 3) semi-supervised representation learning to improve speech recognition. In addition, our approach is more robust in the small sample size regime than alternative methods. Our experiments on real data show the effectiveness of our method in learning a shared space that is more discriminative than current state-of-the-art methods for a variety of tasks.

2 Background

We first introduce CCA and present our task-driven approach in §3. Linear and non-linear CCA are unsupervised and find the shared signal between a pair of data sources, by maximizing the sum correlation between corresponding projections. Let $\mathbf{X}_1 \in \mathbb{R}^{d_1 \times n}$ and $\mathbf{X}_2 \in \mathbb{R}^{d_2 \times n}$ be mean-centered input data from two different views with n samples and d_1, d_2 features, respectively.

CCA. The objective is to maximize the correlation between $\mathbf{a}_1 = \mathbf{w}_1^\top \mathbf{X}_1$ and $\mathbf{a}_2 = \mathbf{w}_2^\top \mathbf{X}_2$, where $\mathbf{w}_1$ and $\mathbf{w}_2$ are projection vectors [1]. The first canonical directions are found via

$$\underset{\mathbf{w}_1, \mathbf{w}_2}{\operatorname{argmax}} \operatorname{corr}\left(\mathbf{w}_1^\top \mathbf{X}_1, \mathbf{w}_2^\top \mathbf{X}_2\right)$$

and subsequent projections are found by maximizing the same correlation but in orthogonal directions. Combining the projection vectors into matrices $\mathbf{W}_1 = [\mathbf{w}_1^{(1)}, \ldots, \mathbf{w}_1^{(k)}]$ and $\mathbf{W}_2 = [\mathbf{w}_2^{(1)}, \ldots, \mathbf{w}_2^{(k)}]$ ($k \leq \min(d_1, d_2)$), CCA can be reformulated as a trace maximization under orthonormality constraints on the projections, i.e.,

$$\underset{\mathbf{W}_1, \mathbf{W}_2}{\operatorname{argmax}} \operatorname{tr}(\mathbf{W}_1^\top \mathbf{\Sigma}_{12} \mathbf{W}_2) \quad \text{s.t. } \mathbf{W}_1^\top \mathbf{\Sigma}_1 \mathbf{W}_1 = \mathbf{W}_2^\top \mathbf{\Sigma}_2 \mathbf{W}_2 = \mathbf{I} \tag{1}$$

for covariance matrices $\mathbf{\Sigma}_1 = X_1 X_1^T$, $\mathbf{\Sigma}_2 = X_2 X_2^T$, and cross-covariance matrix $\mathbf{\Sigma}_{12} = X_1 X_2^T$. Let $\mathbf{T} = \mathbf{\Sigma}_1^{-1/2} \mathbf{\Sigma}_{12} \mathbf{\Sigma}_2^{-1/2}$ and its singular value decomposition (SVD) be $\mathbf{T} = \mathbf{U}_1 \operatorname{diag}(\boldsymbol{\sigma}) \mathbf{U}_2^\top$ with singular values $\boldsymbol{\sigma} = [\sigma_1, \ldots, \sigma_{\min(d_1, d_2)}]$ in descending order. $\mathbf{W}_1$ and $\mathbf{W}_2$ are computed from the top k singular vectors of $\mathbf{T}$ as $\mathbf{W}_1 = \mathbf{\Sigma}_1^{-1/2} \mathbf{U}_1^{(1:k)}$ and $\mathbf{W}_2 = \mathbf{\Sigma}_2^{-1/2} \mathbf{U}_2^{(1:k)}$ where $\mathbf{U}^{(1:k)}$ denotes the k first columns of matrix $\mathbf{U}$. The sum correlation in the projection space is equivalent to

$$\sum_{i=1}^{k} \operatorname{corr}\left(\left(\mathbf{w}_1^{(i)}\right)^\top X_1, \left(\mathbf{w}_2^{(i)}\right)^\top \mathbf{X}_2\right) = \sum_{i=1}^{k} \sigma_i^2 \ , \tag{2}$$

i.e., the sum of the top k singular values. A regularized variation of CCA (RCCA) ensures that the covariance matrices are positive definite by computing the covariance matrices as $\hat{\mathbf{\Sigma}}_1 = \frac{1}{n-1} \mathbf{X}_1 \mathbf{X}_1^\top + r\mathbf{I}$ and $\hat{\mathbf{\Sigma}}_2 = \frac{1}{n-1} \mathbf{X}_2 \mathbf{X}_2^\top + r\mathbf{I}$, for regularization parameter $r > 0$ and identity matrix $\mathbf{I}$ [23].

DCCA. Deep CCA adds non-linear projections to CCA by non-linearly mapping the input via a multilayer perceptron (MLP). In particular, inputs $\mathbf{X}_1$ and $\mathbf{X}_2$ are mapped via non-linear functions f_1 and f_2, parameterized by θ_1 and θ_2, resulting in activations $\mathbf{A}_1 = f_1(\mathbf{X}_1; \theta_1)$ and $\mathbf{A}_2 = f_2(\mathbf{X}_2; \theta_2)$ (assumed to be mean centered) [3]. When implemented by a NN, $\mathbf{A}_1$ and $\mathbf{A}$ are the output activations of the final layer with d_o features. Fig. 1(a) shows the network structure. DCCA optimizes the same objective as CCA, see Eq. (1), but using activations $\mathbf{A}_1$ and $\mathbf{A}_2$. Regularized covariance matrices are computed accordingly and the solution for $\mathbf{W}_1$ and $\mathbf{W}_2$ can be computed using SVD just as with linear CCA. When $k = d_o$ (i.e., the number of CCA components is equal to the number of features in

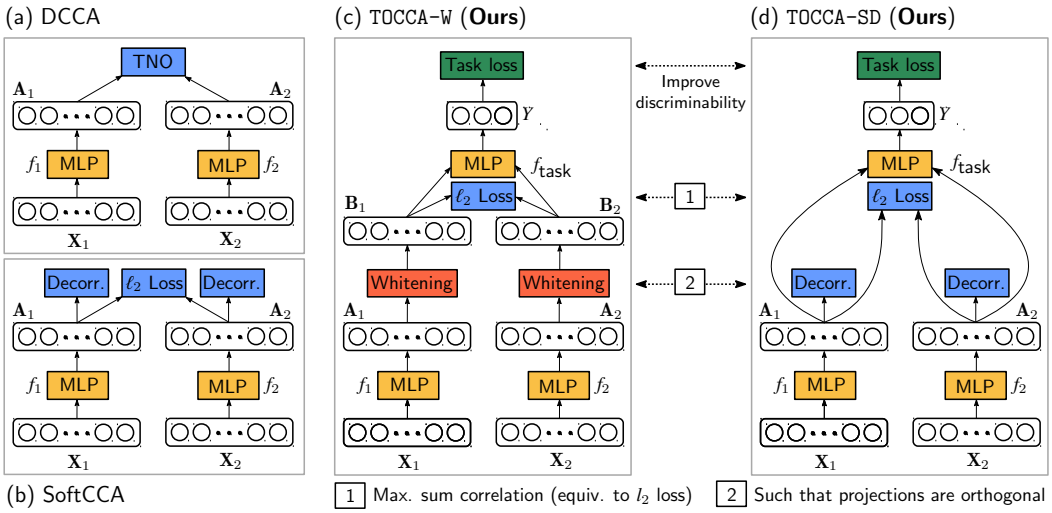

Figure 1: Deep CCA architectures: (a) DCCA maximizes the sum correlation in projection space by optimizing an equivalent loss, the trace norm objective (TNO) [3]; (b) SoftCCA relaxes the orthogonality constraints by regularizing with soft decorrelation (Decorr) and optimizes the ℓ_2 distance in the projection space (equivalent to sum correlation with activations normalized to unit variance) [8]. Our TOCCA methods add a task loss and apply CCA orthogonality constraints by regularizing in two ways: (c) TOCCA-W uses whitening and (d) TOCCA-SD uses Decorr. The third method that we propose, TOCCA-ND, simply removes the Decorr components of TOCCA-SD.

$\mathbf{A}_1$ and $\mathbf{A}_2$), optimizing the sum correlation in the projection space, as in Eq. (2), is equivalent to optimizing the following matrix *trace norm objective (TNO)*

$$\mathcal{L}_{\text{TNO}}(\mathbf{A}_1, \mathbf{A}_2) = \|\mathbf{T}\|_{\text{tr}} = \text{tr}\big(\mathbf{T}^\top \mathbf{T}\big)^{1/2} \quad ,$$

where $\mathbf{T} = \mathbf{\Sigma}_1^{-1/2} \mathbf{\Sigma}_{12} \mathbf{\Sigma}_2^{-1/2}$ as in case of CCA [3]. DCCA optimizes this objective directly, *without* a need to compute the CCA projection within the network. The TNO is optimized first, followed by a linear CCA operation before downstream tasks like classification are performed.

SoftCCA. While DCCA enforces orthogonality constraints on projections $\mathbf{W}_1^\top \mathbf{A}_1$ and $\mathbf{W}_2^\top \mathbf{A}_2$, SoftCCA relaxes them using regularization [8]. Final projection matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ are integrated into f_1 and f_2 as the top network layer. The trace objective for DCCA in Eq. (1) can be rewritten as minimizing the ℓ_2 distance between the projections when each feature in $\mathbf{A}_1$ and $\mathbf{A}_2$ is normalized to a unit variance [24], leading to[1] $\mathcal{L}_{\ell_2 \text{ dist}}(A_1, A_2) = \|\mathbf{A}_1 - \mathbf{A}_2\|_F^2$. Regularization in SoftCCA penalizes the off-diagonal elements of the covariance matrix $\mathbf{\Sigma}$, using a running average computed over batches as $\hat{\mathbf{\Sigma}}$ and a loss of $\mathcal{L}_{\text{Decorr}}(\mathbf{A}) = \sum_{i \neq i}^{d_o} |\hat{\Sigma}_{i,j}|$. Overall, the SoftCCA loss takes the form

$$\mathcal{L}_{\ell_2 \text{ dist}}(\mathbf{A}_1, \mathbf{A}_2) + \lambda\big(\mathcal{L}_{\text{Decorr}}(\mathbf{A}_1) + \mathcal{L}_{\text{Decorr}}(\mathbf{A}_2)\big) \quad .$$

Supervised CCA methods. CCA, DCCA, and SoftCCA are all unsupervised methods to learn a projection to a shared space in which the data is maximally correlated. Although these methods have shown utility for discriminative tasks, a CCA decomposition may not be optimal for classification because features that are correlated may not be discriminative. Our experiments will show that maximizing the correlation objective too much can degrade performance on discriminative tasks.

CCA has previously been extended to supervised settings by maximizing the total correlation between each view and the training labels in addition to each pair of views [12, 13], and by maximizing the separation of classes [6, 10]. Although these methods incorporate the class labels, they do not directly optimize for classification. Dorfer et. al's CCA Layer (CCAL) is the closest to our method. It optimizes a task loss operating on a CCA projection; however, the CCA objective itself is only optimized during pre-training, not in an end-to-end manner [15]. Other supervised CCA methods are linear only [13, 12, 6, 14]. Instead of computing the CCA projection within the network, as in CCAL, we optimize the non-linear mapping into the shared space *together* with the CCA part.

[1] We use this ℓ_2 distance objective in our formulation.

3 Task-Optimal CCA (TOCCA)

To compute a shared latent space that is also discriminative, we start with the DCCA formulation and add a task-driven term to the optimization objective. The CCA component finds features that are correlated between views, while the task component ensures that they are also discriminative. This model can be used for representation learning on multiple views before joining representations for prediction [9, 10] and for classification when two views are available for training but only one at test time [11]. In §4, we demonstrate both use cases on real data. Our methods and related NN models from the literature are summarized in Tab. S1 (suppl. material); Fig. 1 shows schematic diagrams.

While DCCA optimizes the sum correlation through an equivalent loss function (TNO), the CCA projection itself is computed only *after* optimization. Hence, the projections cannot be used to optimize another task simultaneously. The main challenge in developing a task-optimal form of deep CCA that discriminates based on the CCA projection is in computing this projection within the network – a necessary step to enable simultaneous training of both objectives. We tackle this by focusing on the two components of DCCA: maximizing the sum correlation between activations $\mathbf{A}_1$ and $\mathbf{A}_2$ and enforcing orthonormality constraints within $\mathbf{A}_1$ and $\mathbf{A}_2$. We achieve both by transforming the CCA objective and present three methods that progressively relax the orthogonality constraints.

We further improve upon DCCA by enabling mini-batch computations for improved flexibility and test performance. DCCA was developed for large batches because correlation is not separable across batches. While large batch implementations of stochastic gradient optimization can increase computational efficiency via parallelism, small batch training provides more up-to-date gradient calculations, allowing a wider range of learning rates and improving test accuracy [25]. We reformulate the correlation objective as the ℓ_2 distance (following SoftCCA), enabling separability across batches. We ensure a normalization to one via batch normalization without the scale and shift parameters [26].

Task-driven objective. First, we apply non-linear functions f_1 and f_2 (via MLPs) to each view $\mathbf{X}_1$ and $\mathbf{X}_2$, i.e., $\mathbf{A}_1 = f_1(\mathbf{X}_1; \theta_1)$ and $\mathbf{A}_2 = f_2(\mathbf{X}_2; \theta_2)$. Second, a task-specific function $f_{\text{task}}(\mathbf{A}; \theta_{\text{task}})$ operates on the outputs $\mathbf{A}_1$ and $\mathbf{A}_2$. In particular, f_1 and f_2 are optimized so that the ℓ_2 distance between $\mathbf{A}_1$ and $\mathbf{A}_2$ is minimized; therefore, f_{task} can be trained to operate on both inputs $\mathbf{A}_1$ and $\mathbf{A}_2$. We combine CCA and task-driven objectives as a weighted sum with a hyperparameter for tuning. This model is flexible, in that the task-driven goal can be used for classification [27, 28], regression [29], clustering [30], or any other task. See Tab. S1 (suppl. material) for an overview.

Orthogonality constraints. The remaining complications for mini-batch optimization are the orthogonality constraints, for which we propose three solutions, each handling the orthogonality constraints of CCA in a different way: whitening, soft decorrelation, and no decorrelation.

1) Whitening (`TOCCA-W`)**.** CCA applies orthogonality constraints to $\mathbf{A}_1$ and $\mathbf{A}_2$. We accomplish this with a linear whitening transformation that transforms the activations such that their covariance becomes the identity matrix, i.e., features are uncorrelated. Decorrelated Batch Normalization (DBN) has previously been used to regularize deep models by decorrelating features [21] and inspired our solution. In particular, we apply a transformation $\mathbf{B} = \mathbf{U}\mathbf{A}$ to make $\mathbf{B}$ orthonormal, i.e., $\mathbf{B}\mathbf{B}^\top = \mathbf{I}$.

We use a Zero-phase Component Analysis (ZCA) whitening transform composed of three steps: rotate the data to decorrelate it, rescale each axis, and rotate back to the original space. Each of these transformations is learned from the data. Any matrix $\mathbf{U} \epsilon \mathbb{R}^{d_o \times d_o}$ satisfying $\mathbf{U}^\top\mathbf{U} = \mathbf{\Sigma}^{-1}$ whitens the data, where $\mathbf{\Sigma}$ denotes the covariance matrix of $\mathbf{A}$. As $\mathbf{U}$ is only defined up to a rotation, it is not unique. PCA whitening follows the first two steps and uses the eigendecomposition of $\mathbf{\Sigma}$: $\mathbf{U}_{PCA} = \mathbf{\Lambda}^{-1/2}\mathbf{V}^\top$ for $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_{d_o})$ and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_{d_o}]$, where $(\lambda_i, \mathbf{v}_i)$ are the eigenvalue, eigenvector pairs of $\mathbf{\Sigma}$. As PCA whitening suffers from stochastic axis swapping, neurons are not stable between batches [21]. ZCA whitening uses the transformation $\mathbf{U}_{\text{ZCA}} = \mathbf{V}\mathbf{\Lambda}^{-1/2}\mathbf{V}^T$ in which PCA whitening is first applied, followed by a rotation back to the original space. Adding the rotation $\mathbf{V}$ brings the whitened data $\mathbf{B}$ as close as possible to the original data $\mathbf{A}$ [31].

Computation of $\mathbf{U}_{\text{ZCA}}$ is clearly depend on $\mathbf{\Sigma}$. While Huang et al. [21] used a running average of $\mathbf{U}_{\text{ZCA}}$ over batches, we apply this stochastic approximation to $\mathbf{\Sigma}$ for each view using the update $\mathbf{\Sigma}^{(k)} = \alpha\mathbf{\Sigma}^{(k-1)} + (1-\alpha)\mathbf{\Sigma}^b$ for batch k where $\mathbf{\Sigma}^b$ is the covariance matrix for the current batch and $\alpha \in (0, 1)$ is the momentum. We then compute the ZCA transformation from $\mathbf{\Sigma}^{(k)}$ to do whitening as $\mathbf{B} = f_{\text{ZCA}}(\mathbf{A}) = \mathbf{U}_{\text{ZCA}}^{(k)}\mathbf{A}$. At test time, $\mathbf{U}^{(k)}$ from the last training batch is used. Algorithm 1

(suppl. material) describes ZCA whitening in greater detail. In summary, TOCCA-W integrates both the correlation and task-driven objectives, with decorrelation performed by whitening, into

$$\mathcal{L}_{\text{task}}(f_{\text{task}}(\mathbf{B}_1), Y) + \mathcal{L}_{\text{task}}(f_{\text{task}}(\mathbf{B}_2), Y) + \lambda\, \mathcal{L}_{\ell_2 \text{ dist}}(\mathbf{B}_1, \mathbf{B}_2) \ ,$$

where $\mathbf{B}_1$ and $\mathbf{B}_2$ are whitened outputs of $\mathbf{A}_1$ and $\mathbf{A}_2$, respectively.

2) Soft decorrelation (TOCCA-SD). While fully independent components may be beneficial in regularizing NNs on some data sets, a softer decorrelation may be more suitable on others. In this second formulation we relax the orthogonality constraints using regularization, following the Decorr loss of SoftCCA [8]. The loss function for this formulation is

$$\mathcal{L}_{\text{task}}(f_{\text{task}}(\mathbf{A}_1), Y)+\mathcal{L}_{\text{task}}(f_{\text{task}}(\mathbf{A}_2), Y)+\lambda_1\mathcal{L}_{\ell_2 \text{ dist}}(\mathbf{A}_1, \mathbf{A}_2)+\lambda_2\big(\mathcal{L}_{\text{Decorr}}(\mathbf{A}_1)+\mathcal{L}_{\text{Decorr}}(\mathbf{A}_2)\big) \ .$$

3) No decorrelation (TOCCA-ND). When CCA is used in an unsupervised manner, some form of orthogonality constraint or decorrelation is necessary to ensure that f_1 and f_2 do not simply produce multiple copies of the same feature. While this result could maximize the sum correlation, it is not helpful in capturing useful projections. In the task-driven setting, the discriminative term ensures that the features in f_1 and f_2 are not replicates of the same information. TOCCA-ND therefore removes the decorrelation term entirely, forming the simpler objective

$$\mathcal{L}_{\text{task}}(f_{\text{task}}(\mathbf{A}_1), Y) + \mathcal{L}_{\text{task}}(f_{\text{task}}(\mathbf{A}_2), Y) + \lambda\mathcal{L}_{\ell_2 \text{ dist}}(\mathbf{A}_1, \mathbf{A}_2) \ .$$

These three models allow testing whether whitening or soft decorrelation benefit a task-driven model.

Computational complexity. Due to the eigendecomposition, TOCCA-W has a complexity of $O(d_o^3)$ compared to $O(d_o^2)$ for TOCCA-SD, with respect to output dimension d_o. However, d_o is typically small (≤ 100) and this extra computation is only performed once per batch. The difference in runtime is less than 6.5% for a batch size of 100 or 9.4% for a batch size of 30 (Table S3, suppl. material).

In summary, all three variants are motivated by adding a task-driven component to deep CCA. TOCCA-ND is the most relaxed and directly attempts to obtain identical latent representations. Experiments will show that whitening (TOCCA-W) and soft decorrelation (TOCCA-SD) provide a beneficial regularization. Further, since the ℓ_2 distance that we optimize was shown to be equivalent to the sum correlation (cf. §2 SoftCCA paragraph), all three TOCCA models maintain the goals of CCA just with different relaxations of the orthogonality constraints. See Tab. S1 (suppl. material) for an overview.

4 Experiments

We validated our methods on three different data sets: MNIST handwritten digits, the Carolina Breast Cancer Study (CBCS) using imaging and genomic features, and speech data from the Wisconsin X-ray Microbeam Database (XRMB). Our experiments show the utility of our methods for (1) cross-view classification, (2) regularization with a second view during training when only one view is available at test time, and (3) representation learning on multiple views that are joined for prediction.

Implementation.[2] Each layer of our network consists of a fully connected layer, followed by a ReLU activation and batch normalization [26]. We used the Nadam optimizer and tuned hyperparameters on a validation set via random search; settings and ranges are specified in Table S2 (suppl. material). We used Keras with the Theano backend and an Nvidia GeForce GTX 1080 Ti. Our implementations of DCCA, SoftCCA, and Joint DCCA/DeepLDA [10] also use ReLu activation and batch normalization. We modified CCAL-$\mathcal{L}_{\text{rank}}$ [15] to use a softmax function and cross-entropy loss for classification, instead of a pairwise ranking loss for retrieval, referring to this modification as CCAL-$\mathcal{L}_{\text{ce}}$.

4.1 Cross-view classification on MNIST digits

We formed a multi-view data set from the MNIST handwritten digit image data set [22]. Following Andrew et al. [3], we split each 28×28 image in half horizontally, creating left and right views that are each 14×28 pixels. All images were flattened into a vector with 392 features. The full data set consists of 60k training images and 10k test images. We used a random set of up to 50k for training and the remaining training images for validation. We used the full 10k image test set.

[2] Code will be available on GitHub soon.

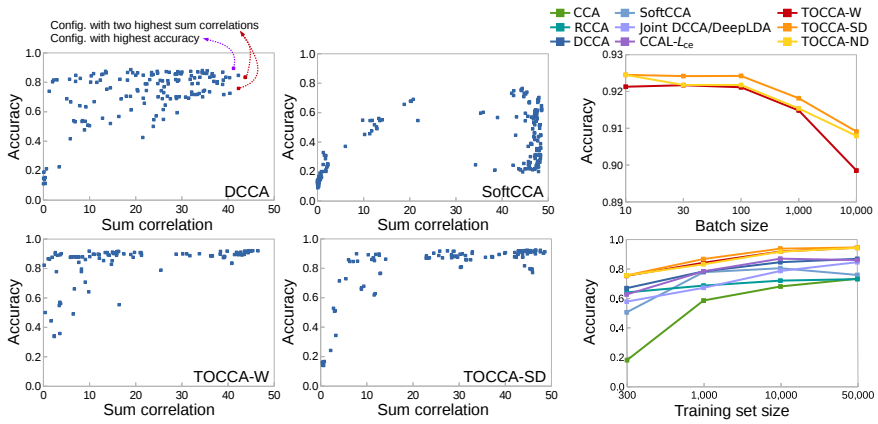

Figure 2: *Left*: Sum correlation vs. cross-view classification accuracy (on MNIST) across different hyperparameter settings on a training set size of 10,000 for DCCA [3], SoftCCA [8], `TOCCA-W`, and `TOCCA-SD`. For unsupervised methods (DCCA and SoftCCA), large correlations do not necessarily imply good accuracy. *Right*: The effect of batch size on classification accuracy for each `TOCCA` method on MNIST (training set size of 10,000), and the effect of training set size on classification accuracy for each method. Our `TOCCA` variants out-performed all others across all training set sizes.

We evaluated cross-view classification accuracy by first computing the projection for each view, then we trained a linear SVM on one view's projection, and finally we used the other view's projection at test time. While the task-driven methods presented in this work learn a classifier within the model, this test setup enables a fair comparison with the unsupervised CCA variants and validates the discriminativity of the features learned. Notably, using the built-in softmax classifier performed similarly to the SVM (not shown), as much of the power of our methods comes from the representation learning part. We do not compare with a simple supervised NN because this setup does not learn the shared space necessary for cross-view classification. We report results averaged over five randomly selected training/validation sets; the test set always remained the same.

Correlation vs. classification accuracy We first demonstrate the importance of adding a task-driven component to DCCA by showing that maximizing the sum correlation between views is not sufficient. Fig. 2 (*left*) shows the sum correlation vs. cross-view classification accuracy across many different hyperparameter settings for DCCA [3], SoftCCA [8], and `TOCCA`. We used 50 components for each; thus, the maximum sum correlation was 50. The sum correlation was measured after applying linear CCA to ensure that components were independent. With DCCA a larger correlation tended to produce a larger classification accuracy, but there was still a large variance in classification accuracy amongst hyperparameter settings that produced a similar sum correlation. For example, with the two farthest right points in the plot (colored red), their classification accuracy differs by 10%, and they are not even the points with the best classification accuracy (colored purple). The pattern is different for SoftCCA. There was an increase in classification accuracy as sum correlation increased but only up to a point. For higher sum correlations, the classification accuracy varied even more from 20% to 80%. Further experiments (not shown) have indicated that when the sole objective is correlation, some of the projection directions are simply not discriminative, particularly when there are a large number of classes. Hence, optimizing for sum correlation alone does not guarantee a discriminative model. `TOCCA-W` and `TOCCA-SD` show a much greater classification accuracy across a wide range of correlations and, overall, the best accuracy when correlation is greatest.

Effect of batch size. Fig. 2 (*right*) plots the batch size vs. classification accuracy for a training set size of $10,000$. We tested batch sizes from 10 to 10,000; a batch size of 10 or 30 was best for all three variations of `TOCCA`. This is in line with previous work that found the best performance with a batch size between 2 and 32 [25]. We used a batch size of 32 in the remaining experiments on MNIST.

Effect of training set size. We manipulated the training set size in order to study the robustness of our methods. In particular, Fig. 2 (*right*) shows the cross-view classification accuracy for training set sizes from $n = 300$ to $50{,}000$. While we expected that performance would decrease for smaller training set sizes, some methods were more susceptible to this degradation than others. The classification accuracy with CCA dropped significantly for $n = 300$ and $1{,}000$, due to overfitting and instability issues related to the covariance and cross-covariance matrices. SoftCCA shows similar behavior (prior work [8] on this method did not test such small training set sizes).

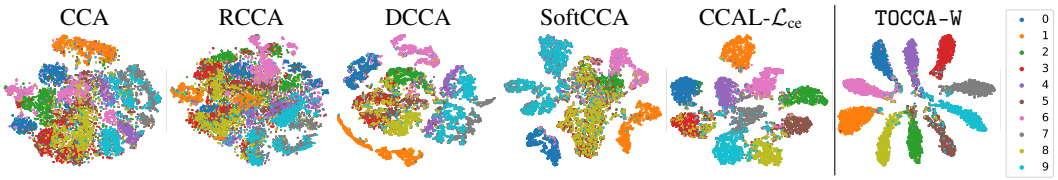

Figure 3: t-SNE plots for CCA methods on our variation of MNIST. Each method was used to compute projections for the two views (left and right sides of the images) using 10,000 training examples. The plots show a visualization of the projection for the left view with each digit colored differently. TOCCA-SD and TOCCA-ND (not shown) produced similar results to TOCCA-W.

Table 1: Classification accuracy for different methods of predicting Basal genomic subtype from images or grade from gene expression. Linear SVM and DNN were trained on a single view, while all other methods were trained with both views. By regularizing with the second view during training, all TOCCA variants improved classification accuracy. The standard error is in parentheses.

Method	Training data	Test data	Task	Accuracy
Linear SVM	Image only	Image	Basal	0.777 (0.003)
NN	Image only	Image	Basal	0.808 (0.006)
CCAL-$\mathcal{L}_{ce}$	Image+GE	Image	Basal	0.807 (0.008)
TOCCA-W	Image+GE	Image	Basal	**0.830 (0.006)**
TOCCA-SD	Image+GE	Image	Basal	0.818 (0.006)
TOCCA-ND	Image+GE	Image	Basal	0.816 (0.004)

Method	Training data	Test data	Task	Accuracy
Linear SVM	GE only	GE	Grade	0.832 (0.012)
NN	GE only	GE	Grade	0.830 (0.012)
CCAL-$\mathcal{L}_{ce}$	GE+image	GE	Grade	0.804 (0.022)
TOCCA-W	GE+image	GE	Grade	**0.862 (0.013)**
TOCCA-SD	GE+image	GE	Grade	0.856 (0.011)
TOCCA-ND	GE+image	GE	Grade	0.856 (0.011)

Across all training set sizes, our TOCCA variations consistently exhibited good performance, e.g., increasing classification accuracy from 78.3% to 86.7% for $n = 1{,}000$ with TOCCA-SD. Increases in accuracy over TOCCA-ND were small, indicating that the different decorrelation schemes have only a small effect on this data set; the task-driven component is the main reason for the success of our method. In particular, the classification accuracy with $n = 1{,}000$ did better than the unsupervised DCCA method on $n = 10{,}000$. Further, TOCCA with $n = 300$ did better than linear methods on $n = 50{,}000$, clearly showing the benefits of the proposed formulation. We also examined the CCA projections qualitatively via a 2D t-SNE embedding [32]. Fig. 3 shows the CCA projection of the left view for each method. As expected, the task-driven variant produced more clearly separated classes.

4.2 Regularization for cancer classification

In this experiment, we address the following question: Given two views available for training but only one at test time, does the additional view help to regularize the model?

We study this question using 1,003 patient samples with image and genomic data from CBCS[3] [33]. Images consisted of four cores per patient from a tissue microarray that was stained with hematoxylin and eosin. Image features were extracted using a VGG16 backbone [34], pre-trained on ImageNet, by taking the mean of the 512D output of the fourth set of conv. layers across the tissue region and further averaging across all core images for the same patient. For gene expression (GE), we used the set of 50 genes in the PAM50 array [35]. The data set was randomly split into half for training and one quarter for validation/testing; we report the mean over eight cross-validation runs. Classification tasks included predicting (1) Basal vs. non-Basal genomic subtype using images, which is typically done from GE, and (2) predicting grade 1 vs. 3 from GE, typically done from images. This is not a multi-task classification setup; it is a means for one view to stabilize the representation of the other.

We tested different classifier training methods when only one view was available at test time: a) a linear SVM trained on one view, b) a deep NN trained on one view using the same architecture as the lower layers of TOCCA, c) CCAL-$\mathcal{L}_{ce}$ trained on both views, d) TOCCA trained on both views. Table 1 lists the classification accuracy for each method and task. When predicting genomic subtype Basal from images, all our methods showed an improvement in classification accuracy; the best result was with TOCCA-W, which produced a 2.2% improvement. For predicting grade from GE, all our methods again improved the accuracy – by up to 3.2% with TOCCA-W. These results show that having additional information during training can boost performance at test time. Notably, this experiment used a static set of pre-trained VGG16 image features in order to assess the utility of the method. The

[3] http://cbcs.web.unc.edu/for-researchers/

network itself could be fine-tuned end-to-end with our TOCCA model, providing an easy opportunity for data augmentation and likely further improvements in classification accuracy.

4.3 Semi-supervised learning for speech recognition

Our final experiments use speech data from XRMB, consisting of simultaneously recorded acoustic and articulatory measurements. Prior work has shown that CCA-based algorithms can improve phonetic recognition [36, 4, 5, 10]. The 45 speakers were split into 35 for training, 2 for validation, and 8 for testing – a total of 1,429,236 samples for training, 85,297 for validation, and 111,314 for testing.[4] The acoustic features are 112D and the articulatory ones are 273D. We removed the per-speaker mean & variance for both views. Samples are annotated with one of 38 phonetic labels.

Our task on this data set was representation learning for multi-view prediction – that is, using both views of data to learn a shared discriminative representation. We trained each model using both views and their labels. To test each CCA model, we followed prior work and concatenated the original input features from both views with the projections from both views. Due to the large training set size, we used a Linear Discriminant Analysis (LDA) classifier for efficiency. The same construction was used at test time. This setup was used to assess whether a task-optimal DCCA model can improve discriminative power. We tested TOCCA with a task-driven loss of LDA [28] or softmax to demonstrate the flexibility of our model.

Table 4: XRMB classification results.

Method	Task	Accuracy
Baseline	-	0.591
CCA	-	0.589
RCCA	-	0.588
DCCA	-	0.620
SoftCCA	-	0.635
Joint DCCA/DeepLDA	LDA	0.633
CCAL-$\mathcal{L}_{\text{ce}}$	Softmax	0.642
TOCCA-W	LDA	0.710
TOCCA-SD	LDA	0.677
TOCCA-ND	LDA	0.677
TOCCA-W	Softmax	**0.795**
TOCCA-SD	Softmax	0.785
TOCCA-ND	Softmax	0.785

We compared the discriminability of a variety of methods to learn a shared latent representation. Table 4 lists the classification results with a baseline that used only the original input features for LDA. Although deep methods, i.e., DCCA and SoftCCA, improved upon the linear methods, all TOCCA variations significantly outperformed previous state-of-the-art techniques. Using softmax consistently beat LDA by a large margin. TOCCA-SD and TOCCA-ND produced equivalent results as a weight of 0 on the decorrelation term performed best. However, TOCCA-W showed the best result with an improvement of 15% over the best alternative method.

TOCCA can also be used in a *semi-supervised* manner when labels are available for only some samples. Table 5 lists the results for TOCCA-W in this setting. With 0% labeled data, the result would be similar to DCCA. Notably, a large improvement over the unsupervised results in Table 4 is seen even with labels for only 10% of the training samples.

Table 5: Semi-supervised classification results on XRMB using TOCCA-W.

Labeled data	Accuracy
100%	0.795
30%	0.762
10%	0.745
3%	0.684
1%	0.637

5 Discussion

We proposed a method to find a shared latent space that is also discriminative by adding a task-driven component to deep CCA while enabling end-to-end training. This was accomplished by replacing the CCA projection with ℓ_2 distance minimization and orthogonality constraints on the activations, and was implemented in three different ways. TOCCA-W or TOCCA-SD performed the best, dependent on the data set – both of which include some means of decorrelation to provide an extra regularizing effect to the model and thereby outperforming TOCCA-ND.

TOCCA showed large improvements over state-of-the-art in cross-view classification accuracy on MNIST and significantly increased robustness when the training set size was small. On CBCS, TOCCA provided a regularizing effect when both views were available for training but only one at test time. TOCCA also produced a large increase over state-of-the-art for multi-view representation learning on a much larger data set, XRMB. On this data set we also demonstrated a semi-supervised approach to get a large increase in classification accuracy with only a small proportion of the labels. Using a similar technique, our method could be applied when some samples are missing a second view.

[4] http://ttic.uchicago.edu/~klivescu/XRMB_data/full/README

Classification tasks using a softmax operation or LDA were explored in this work; however, the formulation presented can also be used with other tasks such as regression or clustering. Another possible avenue for future work entails extracting components shared by both views as well as individual components. This approach has been developed for dictionary learning [37–39] but could be extended to deep CCA-based methods. Finally, we have yet to apply data augmentation to the proposed framework; this could provide a significant benefit for small training sets.

References

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

[2] Tijl De Bie, Nello Cristianini, and Roman Rosipal. Eigenproblems in pattern recognition. In *Handbook of Geometric Computing*, pages 129–167. Springer Berlin Heidelberg, 2005.

[3] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep Canonical Correlation Analysis. In *Proc. ICML*, 2013.

[4] Weiran Wang, Raman Arora, Karen Livescu, and Jeff Bilmes. On deep multi-view representation learning. In *Proc. ICML*, 2015.

[5] Weiran Wang, Raman Arora, Karen Livescu, and Nathan Srebro. Stochastic optimization for deep CCA via nonlinear orthogonal iterations. In *Proc. Allerton Conference on Communication, Control, and Computing*, 2016.

[6] Meina Kan, Shiguang Shan, Haihong Zhang, Shihong Lao, and Xilin Chen. Multi-view Discriminant Analysis. *IEEE PAMI*, 2015.

[7] Sarath Chandar, Mitesh M. Khapra, Hugo Larochelle, and Balaraman Ravindran. Correlational Neural Networks. *Neural Computation*, 28(2):257–285, feb 2016.

[8] Xiaobin Chang, Tao Xiang, and Timothy M. Hospedales. Scalable and Effective Deep CCA via Soft Decorrelation. In *Proc. CVPR*, 2018.

[9] Mehmet Emre Sargin, Yücel Yemez, Engin Erzin, and A Murat Tekalp. Audiovisual synchronization and fusion using canonical correlation analysis. *IEEE Transactions on Multimedia*, 9(7):1396–1403, 2007.

[10] Matthias Dorfer, Gerhard Widmer, and Gerhard Widmerajku At. Towards Deep and Discriminative Canonical Correlation Analysis. In *Proc. ICML Workshop on Multi-view Representaiton Learning*, 2016.

[11] Raman Arora and Karen Livescu. Kernel cca for multi-view learning of acoustic features using articulatory measurements. In *Symposium on Machine Learning in Speech and Language Processing*, 2012.

[12] George Lee, Asha Singanamalli, Haibo Wang, Michael D Feldman, Stephen R Master, Natalie N C Shih, Elaine Spangler, Timothy Rebbeck, John E Tomaszewski, and Anant Madabhushi. Supervised multi-view canonical correlation analysis (sMVCCA): integrating histologic and proteomic features for predicting recurrent prostate cancer. *IEEE Transactions on Medical Imaging*, 34(1):284–97, jan 2015.

[13] Asha Singanamalli, Haibo Wang, George Lee, Natalie Shih, Mark Rosen, Stephen Master, John Tomaszewski, Michael Feldman, and Anant Madabhushi. Supervised multi-view canonical correlation analysis: fused multimodal prediction of disease diagnosis and prognosis. In *Proc. SPIE Medical Imaging*, 2014.

[14] Kanghong Duan, Hongxin Zhang, and Jim Jing Yan Wang. Joint learning of cross-modal classifier and factor analysis for multimedia data classification. *Neural Computing and Applications*, 27(2):459–468, feb 2016.

[15] Matthias Dorfer, Jan Schlüter, Andreu Vall, Filip Korzeniowski, and Gerhard Widmer. End-to-end cross-modality retrieval with CCA projections and pairwise ranking loss. *International Journal of Multimedia Information Retrieval*, 7(2):117–128, jun 2018.

[16] Sumit Shekhar, Vishal M Patel, Nasser M Nasrabadi, and Rama Chellappa. Joint sparse representation for robust multimodal biometrics recognition. *IEEE PAMI*, 36(1):113–26, jan 2014.

[17] Xing Xu, Atsushi Shimada, Rin-ichiro Taniguchi, and Li He. Coupled dictionary learning and feature mapping for cross-modal retrieval. In *Proc. International Conference on Multimedia and Expo*, 2015.

[18] Miriam Cha, Youngjune Gwon, and H. T. Kung. Multimodal sparse representation learning and applications. *arXiv preprint: 1511.06238*, 2015.

[19] Soheil Bahrampour, Nasser M. Nasrabadi, Asok Ray, and W. Kenneth Jenkins. Multimodal Task-Driven Dictionary Learning for Image Classification. *arXiv preprint: 1502.01094*, 2015.

[20] Gaurav Bhatt, Piyush Jha, and Balasubramanian Raman. Common Representation Learning Using Step-based Correlation Multi-Modal CNN. *arXiv preprint: 1711.00003*, 2017.

[21] Lei Huang, Dawei Yang, Bo Lang, and Jia Deng. Decorrelated Batch Normalization. In *Proc. CVPR*, 2018.

[22] Yann LeCun. The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*, 1998.

[23] Natalia Y. Bilenko and Jack L. Gallant. Pyrcca: regularized kernel canonical correlation analysis in Python and its applications to neuroimaging. *Frontiers in Neuroinformatics*, 10, nov 2016.

[24] Dongge Li, Nevenka Dimitrova, Mingkun Li, and Ishwar K. Sethi. Multimedia content processing through cross-modal association. In *Proc. ACM International Conference on Multimedia*, 2003.

[25] Dominic Masters and Carlo Luschi. Revisiting Small Batch Training for Deep Neural Networks. *arxiv preprint: 1804.07612*, 2018.

[26] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proc. ICML*, 2015.

[27] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1106–1114, 2012.

[28] Matthias Dorfer, Rainer Kelz, and Gerhard Widmer. Deep linear discriminant analysis. In *Proc. ICLR*, 2016.

[29] Jared Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang, and Yuval Kluger. Deep Survival: A Deep Cox Proportional Hazards Network. *arxiv preprint: 1606.00931*, 2016.

[30] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proc. ECCV*, 2018.

[31] Agnan Kessy, Alex Lewin, and Korbinian Strimmer. Optimal whitening and decorrelation. *arXiv preprint: 1512.00809*, 2015.

[32] L Van Der Maaten and G Hinton. Visualizing high-dimensional data using t-sne. journal of machine learning research. *Journal of Machine Learning Research*, 9:26, 2008.

[33] MA Troester, Xuezheng Sun, Emma H. Allott, Joseph Geradts, Stephanie M Cohen, Chui Kit Tse, Erin L. Kirk, Leigh B Thorne, Michelle Matthews, Yan Li, Zhiyuan Hu, Whitney R. Robinson, Katherine A. Hoadley, Olufunmilayo I. Olopade, Katherine E. Reeder-Hayes, H. Shelton Earp, Andrew F. Olshan, LA Carey, and Charles M. Perou. Racial differences in PAM50 subtypes in the Carolina Breast Cancer Study. *Journal of the National Cancer Institute*, 2018.

[34] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proc. ICLR*, 2015.

[35] Joel S Parker, Michael Mullins, Maggie CU Cheang, Samuel Leung, David Voduc, Tammi Vickery, Sherri Davies, Christiane Fauron, Xiaping He, et al. Supervised risk predictor of breast cancer based on intrinsic subtypes. *Journal of Clinical Oncology*, 27(8):1160–1167, 2009.

[36] Weiran Wang, Raman Arora, Karen Livescu, and Jeff A. Bilmes. Unsupervised learning of acoustic features via deep canonical correlation analysis. In *Proc. ICASSP*, 2015.

[37] Eric F Lock, Katherine A Hoadley, J S Marron, and Andrew B Nobel. Joint and Individual Variation Explained (JIVE) for Integrated Analysis of Multiple Data Types. *The Annals of Applied Statistics*, 7(1):523–542, mar 2013.

[38] Priyadip Ray, Lingling Zheng, Joseph Lucas, and Lawrence Carin. Bayesian joint analysis of heterogeneous genomics data. *Bioinformatics*, 30(10):1370–6, may 2014.

[39] Qing Feng, Meilei Jiang, Jan Hannig, and JS Marron. Angle-based joint and individual variation explained. *Journal of Multivariate Analysis*, 166:241–265, 2018.

Supplementary Material

This supplementary material includes additional details on our TOCCA algorithm and experiments, including 1) a comparison of our formulation with other related CCA approaches, 2) pseudocode for the ZCA whitening algorithm used by TOCCA-W, 3) details on hyperparameter selection, and 4) training runtime experiments.

Comparison of TOCCA with related algorithms

Table S1 compares our three TOCCA formulations with other related linear and deep CCA methods.

Table S1: A comparison of our proposed task-optimal deep CCA methods with other related ones from the literature: DCCA [3], SoftCCA [8], CCAL-$\mathcal{L}_{\text{rank}}$ [15]. CCAL-$\mathcal{L}_{\text{rank}}$ uses a pairwise ranking loss with cosine similarity to identify matching and non-matching samples for image retrieval – not classification. A_1 and A_2 are mean centered outputs from two feed-forward networks. $\Sigma = A^T A$ is computed from a single (large) batch (used in DCCA); $\hat{\Sigma}$ is computed as a running mean over batches (for all other methods). $f_{\text{task}}(A; \theta_{\text{task}})$ is a task-specific function with parameters θ_{task}, e.g., a softmax operation for classification.

Method	**Objective**	
CCA	$-\text{tr}(W_1^T \Sigma_{12} W_2)$	s.t. $W_1^T \Sigma_1 W_1 = W_2^T \Sigma_2 W^2 = I$
DCCA	$-\|\|\Sigma_1^{-1/2} \Sigma_{12} \Sigma_2^{-1/2}\|\|_{\text{tr}}$	where $\|\|T\|\|_{\text{tr}} = \text{tr}(T^T T)^{1/2}$ (TNO, equivalent to CCA objective)
		$\text{CCA}(W_1^T A_1, W_2^T A_2)$ computed after optimization complete
SoftCCA	$\mathcal{L}_{\ell_2 \text{ dist}}(A_1, A_2) + \lambda \left(\mathcal{L}_{\text{Decorr}}(A_1) + \mathcal{L}_{\text{Decorr}}(A_2)\right)$	
CCAL-$\mathcal{L}_{\text{rank}}$	$\mathcal{L}_{\text{rank}}(B_1, B_2)$	where $B_1, B_2 = \text{CCA}(A_1, A_2)$, $\mathcal{L}_{\text{rank}}$ is pairwise ranking loss
TOCCA-W	$\text{Task}(B_1, B_2, Y) + \lambda \mathcal{L}_{\ell_2 \text{ dist}}(B_1, B_2)$	where $B_1 = U_1 A_1, B_2 = U_2 A_2$ s.t. $B_1^T B_1 = B_2^T B_2 = I$
TOCCA-SD	$\text{Task}(A_1, A_2, Y) + \lambda_1 \mathcal{L}_{\ell_2 \text{ dist}}(A_1, A_2) + \lambda_2 \left(\mathcal{L}_{\text{Decorr}}(A_1) + \mathcal{L}_{\text{Decorr}}(A_2)\right)$	Whitening
TOCCA-ND	$\text{Task}(A_1, A_2, Y) + \lambda \mathcal{L}_{\ell_2 \text{ dist}}(A_1, A_2)$	
Loss functions		
ℓ_2 dist	$\mathcal{L}_{\ell_2 \text{ dist}}(A_1, A_2) = \|\|A_1 - A_2\|\|_F^2$	
Decorr	$\mathcal{L}_{\text{Decorr}}(A) = \sum_{i \neq j} \lvert\hat{\Sigma}_{i,j}\rvert$	where $\hat{\Sigma}$ is running mean across batches of $\Sigma = A^T A$
Task	$\text{Task}(A_1, A_2, Y) = \mathcal{L}_{\text{task}}(f_{\text{task}}(A_1; \theta_{\text{task}}), Y) + \mathcal{L}_{\text{task}}(f_{\text{task}}(A_2; \theta_{\text{task}}), Y)$	where $\mathcal{L}_{task}$ can be cross-entropy or any other task-driven loss

Algorithm for whitening

Pseudocode for ZCA whitening used to achieve orthogonality in our TOCCA-W implementation is shown in Algorithm 1.

Algorithm 1 Whitening layer for orthogonality.

Input: activations $A \epsilon \mathbb{R}^{d_o \times n}$
Hyperparameters: batch size m, momentum α
Parameters of layer: mean μ, covariance Σ
if training **then**
  $\mu \leftarrow \alpha\mu + (1-\alpha)\frac{1}{m} A\, 1_{n\times 1}$ {Update mean}
  $\bar{A} = A - \mu$ {Mean center data}
  $\Sigma \leftarrow \alpha\Sigma + (1-\alpha)\frac{1}{m-1}\bar{A}_1 \bar{A}_2^T$ {Update covariance}
  $\hat{\Sigma} \leftarrow \Sigma + \epsilon I$ {Add ϵI for numerical stability}
  $\Lambda, V \leftarrow \text{eig}(\hat{\Sigma})$ {Compute eigendecomposition}
  $U \leftarrow V \Lambda^{-1/2} V^T$ {Compute transformation matrix}
else
  $\bar{A} \leftarrow A - \mu$ {Mean center data}
end if
$B \leftarrow U\bar{A}$ {Apply ZCA whitening transform}
return B

Implementation details: hyperparameters

A random search over hyperparameters was used to train our methods. The hyperparameter settings and ranges for each data set are provided in Table S2.

Table S2: Hyperparameter settings and search ranges for the experiments on each data set.

Hyperparameter	**MNIST**	**CBCS**	**XRMB**
Hidden layers	4	[0,4]	4
Hidden layer size	500	200	1,000
Output layer size	50	50	112
Loss function weight λ	$[10^0, 10^{-4}]$	$[10^1, 10^{-5}]$	$[10^1, 10^{-5}]$
Momentum α	0.99	0.99	0.99
ℓ_2 regularizer	$[10^{-3}, 10^{-6}]$, 0	$[10^{-2}, 10^{-5}]$, 0	$[10^{-3}, 10^{-7}]$, 0
Soft decorrelation regularizer	$[10^0, 10^{-5}]$	$[10^0, 10^{-5}]$	$[10^0, 10^{-5}]$
Batch size	32	100	50,000
Learning rate	$[10^{-2}, 10^{-4}]$	$[10^{-1}, 10^{-3}]$	$[10^0, 10^{-4}]$
Epochs	200	400	100

Runtime experiments

The computational complexity of `TOCCA-W` is greater than that of `TOCCA-SD` due to the eigendecomposition operation (see §3 in the main article); however, this extra computation is only carried out once per batch. A runtime comparison of the two methods on all three data sets is provided in Table S3.

Table S3: Training runtime for each data set.

Data set	**Batch size**	**Epochs**	`TOCCA-W`	`TOCCA-SD`
MNIST	100	200	488 s	418 s
MNIST	30	200	1071 s	1036 s
CBCS	100	400	103 s	104 s
XRMB	50,000	100	3056 s	3446 s