Deep Orthogonal Representations: Fundamenta\ls Properties and Applications

Hsiang Hsu, Salman Salamatian, and Flavio P. Calmon

Abstract

Several representation learning and, more broadly, dimensionality reduction techniques seek to produce representations of the data that are orthogonal (uncorrelated). Examples include PCA, CCA, Kernel/Deep CCA, the ACE algorithm and correspondence analysis (CA). For a fixed data distribution, all finite variance representations belong to the same function space regardless of how they are derived. In this work, we present a theoretical framework for analyzing this function space, and demonstrate how a basis for this space can be found using neural networks. We show that this framework (i) underlies recent multi-view representation learning methods, (ii) enables classical exploratory statistical techniques such as CA to be scaled via neural networks, and (iii) can be used to derive new methods for comparing black-box models. We illustrate these applications empirically through different datasets.

Keywords: Representation learning, principal inertia components, canonical correlation analysis, correspondence analysis, multi-view learning, model comparison.

*Hsiang Hsu and Flavio P. Calmon are with John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA (e-mails: hsianghsu@g.harvard.edu, flavio@seas.harvard.edu).
†Salman Salamatian is with the Research Laboratory of Electronics at the Massachusetts Institute of Technology, Cambridge, MA (e-mail: salmansa@mit.edu).
1 Introduction

The performance of machine learning algorithms depends on the chosen representation of the train and test data. Consider data given by a sequence of \( n \) i.i.d. random variables \( \{(X_k, Y_k)\}_{k=1}^{n} \) with \( (X_k, Y_k) \sim P_{X,Y} \), where \( P_{X,Y} \) is a distribution defined over \( \mathcal{X} \times \mathcal{Y} \). Here, \( X_k \) represents a data sample, and \( Y_k \) represents a label to be predicted, a noisy “view” of \( X_k \) (Benton et al. 2017), or any other side information correlated with the original sample. A \( d \)-dimensional representation of the data is defined as a mapping

\[
X \mapsto (f_1(X), \ldots, f_d(X)) \triangleq f(X),
\]

where \( f_i : \mathcal{X} \to \mathbb{R}, i \in \{1, \ldots, d\} \). For example, \( X_k \in \mathcal{X} \) can be an image, and \( f(X_k) \) features extracted by an autoencoder (Bengio et al. 2013; Goodfellow et al. 2016), or the result of a kernel mapping (Shawe-Taylor and Cristianini 2004). Equivalently, a representation for \( Y_i \) is given by a mapping \( Y \mapsto (g_1(Y), \ldots, g_d(Y)) \triangleq g(Y) \) where \( g_i : \mathcal{Y} \to \mathbb{R}, i \in \{1, \ldots, d\} \).

Several dimensionality reduction and representation learning methods seek to produce representations that are orthogonal, i.e., satisfy

\[
E[f(X)f(X)^\top] = I_d,
\]

where \( I_d \in \mathbb{R}^{d \times d} \) is the identity matrix. When \( \mathcal{X} \subseteq \mathbb{R}^n \), perhaps the simplest such method is principal component analysis (PCA). When representations for \( X \) and \( Y \) are simultaneously produced, \( f(X) \) and \( g(Y) \) are often sought to be maximally correlated under constraints on \( f \) and \( g \). Examples include Canonical Correlation Analysis (CCA) (Hotelling 1936), where the representations are restricted to linear functions of \( X \) and \( Y \), Kernel CCA (Bach and Jordan 2002), where \( f \) and \( g \) belong to a reproducing kernel Hilbert space, or Deep CCA (Andrew et al. 2013), where \( f \) and \( g \) are the outputs of a pair of neural networks.

Our overarching goal is to study fundamental properties of representations of the form \( f(X) \) and \( g(Y) \). We bring to bear an information-theoretic tool called the principal inertia components (PICs) of a joint distribution \( P_{X,Y} \) (a formal definition is provided in the second half of this section). The PICs (under different guises) date back to the work of Hirschfeld (1935), Gebelein (1941) and Rényi (1959), and have been studied in the information theory and statistics literature by Witsenhausen (1961; Benton et al. 2017; Wang et al. 2016). Due to the aforementioned equivalence between CA, ACE and DCCA, both (i) the exploratory and visualization methods used in CA and (ii) the non-linear transformations of variables sought out by the (bivariate) ACE method can be scaled to
much larger datasets via deep learning models. Moreover, this connection also embeds DCCA with a greater theoretical significance beyond its current use in multi-view learning. Second, we introduce a method for comparing black box models using PICs. More specifically, we demonstrate how any classification model that outputs a likelihood across labels (e.g., neural network with a final softmax layer, logistic regression, random forests) can be decomposed using the PIC-based analysis presented here. We show how this decomposition can be used to produce new metrics for comparing models.

In the remainder of this section, we introduce the PICs and accompanying technical definitions. A framework for estimating the PICs via neural networks is derived in Section 2. We illustrate how the PICs can be used for comparing across black-box models in Section 3, and provide a brief discussion on how CA, ACE and DCCA are connected in Section 4. Finally, Section 5 presents experimental results on synthetic data, generated from the binary symmetric channels (Cover and Thomas, 2012) and Gaussian distributions, and on real-world data, including noisy MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky and Hinton, 2009) and the ProPublica COMPAS dataset (Angwin et al., 2016). Proofs, an additional discussion on related literature and additional experiments are provided in the Appendix.

1.1 Notation

Capital letters (e.g. X and Y) are used to denote random variables, and calligraphic letters (e.g. \( \mathcal{X} \) and \( \mathcal{F} \)) denote sets. We denote the probability measure of \( X \times Y \) by \( P_{X,Y} \), the conditional probability measure of \( Y \) given \( X \) by \( P_{Y|X} \), and the marginal probability measure of \( X \) and \( Y \) by \( P_X \) and \( P_Y \) respectively. We denote the fact that \( X \) is distributed according to \( P_X \) by \( X \sim P_X \). If \( X \) and \( Y \) have finite support sets \(|\mathcal{X}| < \infty \) and \(|\mathcal{Y}| < \infty \), then we denote the joint probability mass function (pmf) of \( X \) and \( Y \) as \( p_{X,Y} \), the conditional pmf of \( Y \) given \( X \) as \( p_{Y|X} \), and the marginal distributions of \( X \) and \( Y \) as \( p_X \) and \( p_Y \), respectively. Finally, a sample drawn from a probability distribution is denoted by lower-case letters (e.g. \( x \) and \( y \)).

Matrices are denoted in bold capital letters (e.g. \( X \)) and vectors in bold lower-case letters (e.g. \( x \)). The \((i,j)\) entry of a matrix \( X \) is given by \( [X]_{i,j} \). We denote the identity matrix of dimension \( d \) by \( I_d \). Given \( v \in \mathbb{R}^d \), we denote the matrix with diagonal entries equal to \( v \) by \( \text{diag}(v) \).

1.2 Function Spaces and the PICs

We provide next a geometric characterization of the space of possible representations of the form \( f(X) \) and \( g(Y) \). We aim at introducing these concepts in an intuitive way at the expense of some mathematical rigor; the results below hold under mild compactness assumptions on \( P_{X,Y} \). A more thorough introduction to PICs can be found in (Witsenhausen, 1975; Buja, 1990).

We assume throughout that the representations have finite variance i.e., \( \mathbb{E} \| f(X) \|^2 < \infty \) and \( \mathbb{E} \| g(Y) \|^2 < \infty \). For a random variable \( X \) over the alphabet \( \mathcal{X} \), we let \( \mathcal{L}_2(P_X) \) be the Hilbert Space of all functions from \( \mathcal{X} \to \mathbb{R} \) with finite variance with respect to \( P_X \), i.e.

\[
\mathcal{L}_2(P_X) \triangleq \{ f : \mathcal{X} \to \mathbb{R} \mid \mathbb{E}[f(X)^2] < \infty \}.
\]

All possible finite-variance representations of \( X \) lie within this space regardless of the representation learning method used.

For \( f_1, f_2 \in \mathcal{L}_2(P_X) \), this Hilbert space has an associated inner product given by \( \langle f_1, f_2 \rangle = \mathbb{E}[f_1(X) \cdot f_2(X)] \). As customary, this inner product induces a distance\(^1\) between two functions

\[
d(f,g) = \sqrt{\langle f - g, f - g \rangle},
\]

\(^1\)For convenience, and without loss of generality, we let the distance be defined as \( d(f,g) = \langle f - g, f - g \rangle \), as opposed to \( d(f,g) = \sqrt{\langle f - g, f - g \rangle} \).
\( f_1, f_2 \in \mathcal{L}_2(P_X) \), namely the Mean-Square-Error (MSE) distance given by
\[
d(f_1, f_2) = \mathbb{E} \left[ (f_1(X) - f_2(X))^2 \right].
\] (4)

For pairs of random variables \( (X, Y) \sim P_{X,Y} \) taking values in \( \mathcal{X} \times \mathcal{Y} \), we can similarly define the Hilbert Space \( \mathcal{L}_2(P_{X,Y}) \). Note that \( \mathcal{L}_2(P_X) \) and \( \mathcal{L}_2(P_Y) \) are subspaces of \( \mathcal{L}_2(P_{X,Y}) \) and, thus, one can construct the projection operator from \( \mathcal{L}_2(P_X) \) to \( \mathcal{L}_2(P_Y) \) given by
\[
\Pi_Y[f](y) \triangleq \arg\min_{g \in \mathcal{L}_2(P_Y)} \mathbb{E}_{X,Y} \left[ (f(X) - g(Y))^2 \right] = \mathbb{E}[f(X)|Y = y],
\] (5)

with adjoint operator \( \Pi_X[g](x) = \mathbb{E}[g(Y)|X = x] \) defined for \( g \in \mathcal{L}(P_Y) \). The projection operator describes the closest function, in terms of mean-square-error, to a given function \( f \) of the inputs. Since \( \mathcal{L}_2(P_X) \) is a Hilbert space, there exists a basis (in fact infinitely many) through which any function \( f \in \mathcal{L}_2(P_X) \) can be equivalently represented. However, at a high level, it is of interest to find a basis for \( \mathcal{L}_2(P_X) \), which diagonalizes the projection operator \( \Pi_Y \). This naturally leads to the following proposition.

**Proposition 1** (Witsenhausen (1975)). Without loss of generality, let \( |\mathcal{Y}| \leq |\mathcal{X}| \) and let \( \mathcal{d} \triangleq |\mathcal{Y}| - 1 \), or be infinity if both \( |\mathcal{X}| \) and \( |\mathcal{Y}| \) are unbounded. There exists two sets of functions of \( \mathcal{F} = \{f_0, f_1, \ldots, f_d\} \subseteq \mathcal{L}_2(P_X) \) and \( \mathcal{G} = \{g_0, g_1, \ldots, g_d\} \subseteq \mathcal{L}_2(P_Y) \), and a set \( \Lambda = \{1, \lambda_1, \ldots, \lambda_d\} \) such that:

- \( f_0(X) \) and \( g_0(Y) \) are constant function almost surely, \( \mathbb{E}[f_i(X) \cdot f_j(X)] = \delta_{i,j} \) and \( \mathbb{E}[g_i(Y) \cdot g_j(Y)] = \delta_{i,j} \) (orthornormality).

- \( \mathbb{E}[f_i(X)|Y = y] = \sqrt{\lambda_i}g_i(y) \), and \( \mathbb{E}[g_i(Y)|X = x] = \sqrt{\lambda_i}f_i(x) \) for all \( i = 1, \ldots, d \) (diagonalization).

- Any function \( g \in \mathcal{L}_2(P_Y) \) can be represented as a linear combination \( g(y) = \sum_{i=0}^{d} \beta_i g_i(y) \). Similarly, any function \( f \in \mathcal{L}_2(P_X) \) can be represented as a linear combination \( f(x) = f^\perp(x) + \sum_{i=0}^{d} \alpha_i f_i(x) \), where \( f^\perp \) is orthogonal to all \( f_i \) for all \( i = 0, 1, \ldots, d \) (basis).

Borrowing the terminology from CA (Greenacre 1984), the functions within the sets \( \mathcal{F} \) and \( \mathcal{G} \) are defined here as the Principal Functions of \( P_{X,Y} \), and the elements of \( \Lambda \) as the Principal Inertia Components (PICs) of \( P_{X,Y} \). Observe that \( 0 \leq \lambda_i \leq 1 \) and, without loss of generality, we let \( \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d \). We denote \( f_i \) and \( g_i \) the \( i^{th} \) principal functions, and \( \lambda_i \) the \( i^{th} \) PIC; moreover, the \( 0^{th} \) principal functions are the constant 1, and the resulting 0th PIC always has value 1.

The principal functions constitute a particularly useful set of non-linear orthonormal representations of \( X \) that can be reliably reconstructed from \( Y \) (and vice-versa). The decomposition in Prop. 1 allows minimum-MSE (MMSE) estimators to be easily cast in terms of the principal functions of \( X \). Consider the problem of estimating an arbitrary function \( g \) of the labels. Since \( \mathcal{G} \) forms a basis, \( g \) can be written as \( \sum_{i=0}^{d} \beta_i g_i(y) \), and thus, the best estimator of \( g \) from the features \( X \) is given by
\[
\arg\min_{f \in \mathcal{L}_2(P_X)} \mathbb{E} \left[ (f(X) - g(Y))^2 \right] = \sum_{i=0}^{d} \beta_i \sqrt{\lambda_i}f_i(x).
\] (6)

The \( i^{th} \) principal function \( f_i \) and \( g_i \), as well as the PIC \( \lambda_i \), have an equivalent recursive characterization given by (Witsenhausen 1975, Calmon et al. 2017):
\[
\lambda_i(X; Y) = \max_{f \in \mathcal{L}_2(P_X)} \{ \mathbb{E}[f(X)g(Y)]^2 | \mathbb{E}[f(X)f_j(X)] = 0, \mathbb{E}[g(Y)g_j(Y)] = 0, 0 \leq j \leq i - 1 \}. \] (7)
2 Estimating Principal Functions via Neural Networks

We show next how a subset of the principal functions or, equivalently, orthogonal representations of the data, can be directly estimated using deep learning models. More precisely, we will let the principal functions \( f_i \in \mathcal{F} \) and \( g_i \in \mathcal{G} \) be approximated by a pair of neural networks, called the F-Net and G-Net, respectively. The general architecture is depicted in Fig. 1.

For two random variables \((X,Y)\) (e.g., sample/label, view 1/view 2 of an image), we consider representations of the form

\[
\tilde{f}(X,\theta_F) \triangleq [\tilde{f}_1(X,\theta_F), \cdots, \tilde{f}_d(X,\theta_F)]^\top \in \mathbb{R}^{d \times 1},
\]

where each \( \tilde{f}_i : \mathcal{X} \times \Theta_F \to \mathbb{R} \) is an output neuron of the F-Net. Similarly, we define the representation of \( Y \) produced by the G-Net as \( \tilde{g}(Y,\theta_G) \triangleq [\tilde{g}_1(Y,\theta_G), \cdots, \tilde{g}_d(Y,\theta_G)]^\top \in \mathbb{R}^{d \times 1} \). Here, \( \Theta_F \) and \( \Theta_G \) represent model parameters (e.g., layer weights). When \( X \) or \( Y \) have finite support, it suffices to consider \( d = \min\{|\mathcal{X}|,|\mathcal{Y}|\} - 1 \) (Calmon et al. 2017).

Our goal is to extract \( d \) principal functions of \( P_{XY} \) that correspond to the \( d \) largest PICs. For the sake of derivation, we drop for now the parameters \( \Theta_F \) and \( \Theta_G \), and assume we can search over the entire space of representations in \( \mathcal{L}_2(P_X) \) and \( \mathcal{L}_2(P_Y) \). We also assume that \( P_{XY} \) is known (we return to the neural network, data-driven case towards the end of the section).

Under these assumptions, the solution of the optimization problem

\[
\min_{A \in \mathbb{R}^{d \times d}, \tilde{f}, \tilde{g}} \mathbb{E}\left[\|A\tilde{f}(X) - \tilde{g}(Y)\|_2^2\right] \quad \text{subject to} \quad \mathbb{E}\left[\tilde{A}	ilde{f}(X)(\tilde{A}	ilde{f}(X))^\top\right] = I_d
\]

has been proven to recover the \( d \) largest PICs (Rényi 1959). To see why this is the case, let

\[
f(X) = A\tilde{f}(X) = [f_1(X),\cdots,f_d(X)]^\top,
\]

and suppose that \( f,\tilde{g} \) and \( A \) achieve optimality in \(9\). Optimality under quadratic loss implies that \( \tilde{g}_i(y) = \mathbb{E}\{f_i(X) \mid Y = y\} \) for \( i \in \{1,\ldots,d\} \). Moreover, the orthogonality constraint assures that the entries of \( f(X) \) satisfy \( \mathbb{E}\{f_i(X)f_j(X)\} = \delta_{ij} \), and thus form a basis for a \( d \)-dimensional subspace of \( L^2(P_X) \). As discussed in Section 1 conditional expectation on \( Y \) is a (compact) operator from \( L^2(P_X) \to L^2(P_Y) \) (Eq. 5), and, from orthogonality of \( f(X) \), it follows directly from the Hilbert-Schmidt Theorem (Reed and Simon 1980) that the optimal value of \(9\) is \( \sum_{i=0}^{d-1} \lambda_i \), with \( f \) and \( \tilde{g} \) corresponding to the \( d \) largest principal functions.

We can further simplify the objective function in \(9\) using the following proposition.

**Proposition 2.** The minimization in \(9\) is equivalent to the following unconstrained optimization problem.

\[
\min_{\tilde{f}, \tilde{g}} \quad -2\|C_f^{-\frac{1}{2}}C_{fg}\|_d + \mathbb{E}[\|\tilde{g}(Y)\|_2^2],
\]

where \( C_f = \mathbb{E}[\tilde{f}(X)\tilde{f}(X)^\top] \), \( C_{fg} = \mathbb{E}[\tilde{f}(X)\tilde{g}(Y)^\top] \), and \( \|Z\|_d \) is the \( d \)-th Ky-Fan norm, defined as the sum of the singular values of \( Z \) (Horn et al. 1996, Eq. (7.4.8.1)). Denoting by \( A \) and \( B \) the whitening matrices\(^2\) for \( \tilde{f}(X) \) and \( \tilde{g}(Y) \), the principal functions are given by \( f(X) = [f_0(X),\cdots,f_d(X)]^\top = A\tilde{f}(X) \) and \( g(Y) = [g_0(Y),\cdots,g_d(Y)]^\top = B\tilde{g}(Y) \).

\(^2\)We call \( A \) and \( B \) the whitening matrices since in \(7\), it is clear that the covariance matrices of \( f(X) \) and \( g(Y) \) should be identity matrices.
When backpropagating the objective in (11), calculating the singular values of the matrix
\( \tilde{C} \) vectors to align with \( C \) by columns, which can always be done by subtracting the column-mean element-wise. Then Proposition 2. Without loss of generality, we will assume that \( C \) struct the whitening matrices unsuccessful matrix inversion.

Proposition 2. Without loss of generality, we will assume that \( C \) by columns, which can always be done by subtracting the column-mean element-wise. Then Proposition 2. Without loss of generality, we will assume that \( C \) struct the whitening matrices unsuccessful matrix inversion.

\[ \text{Proof. Please refer to Appendix A.1.} \]

\[ \text{Implementation. With the objective (11) in hand, we return to the neural network setting} \]
where \( f(X, \theta_F) \) and \( g(Y, \theta_G) \) are parametrically defined through the FG-Net architecture in Fig. 1.

Given \( n \) realizations (samples) \( \{x_k, y_k\}_{k=1}^n \) from \( P_{X,Y} \), we denote \( x_n \triangleq [x_1, \ldots, x_n], y_n \triangleq [y_1, \ldots, y_n] \). \( \bar{F}_n(x_n) = [f(x_1, \theta_F), \ldots, f(x_n, \theta_F)]^\top \in \mathbb{R}^{d \times n} \) and \( \bar{G}_n(y_n) = [g(y_1, \theta_G), \ldots, g(y_n, \theta_G)] \in \mathbb{R}^{d \times n} \). The empirical evaluations of the terms in (11) are

\[ C_f \approx \frac{1}{n} \bar{F}_n(x_n, \theta_F)\bar{F}_n(x_n, \theta_F)^\top, \]

\[ C_{fg} \approx \frac{1}{n} \bar{F}_n(x_n, \theta_F)\bar{G}_n(y_n, \theta_G)^\top, \]

\[ \mathbb{E}[\|g(Y)\|_2^2] \approx \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^d \tilde{g}_j(y_i, \theta_G)^2. \]

When backpropagating the objective in (11), calculating the singular values of \( C_f^{-1/2}C_{fg} \) is equivalent to calculating the eigenvalues of \( C_f^{-1/2}C_{fg} \). The latter expression can be directly cast and backpropagated in Tensorflow [Abadi et al., 2016] (by tf.self_adjoint_eig). To avoid numerical instability, we not only clip the outputs of the F-Net to the interval \([-10000, 10000]\), but also impose \( L^2 \) regularization [Goodfellow et al., 2016, Chap. 7.1.1] on \( C_f^{-1/2}C_{fg} \), i.e., we compute the eigenvalues of \( C_f^{-1/2}(C_f^{-1} + \epsilon I_d)C_{fg} \) instead, where \( \epsilon = 0.001 \) is the regularization parameter to avoid unsuccessful matrix inversion.

After extracting \( \bar{F}_n(x_n) \) and \( \bar{G}_n(y_n) \) from the F-Net and G-Net respectively, we need to reconstruct the whitening matrices \( A \) for \( f \), and \( B \) for \( g \) to obtain the principal functions, as suggested in Proposition 2. Without loss of generality, we will assume that \( \bar{F}_n(x_n) \) and \( \bar{G}_n(y_n) \) have zero-mean columns, which can always be done by subtracting the column-mean element-wise. Then \( A \) is given by \( A = U^\top C_f^{-1/2} \), with \( U \) the left singular vectors of the matrix

\[ L = \frac{1}{n}(C_f^{-1/2}\bar{F}_n(x_n))(C_g^{-1/2}\bar{G}_n(y_n))^\top. \]

The matrix \( C_f^{-1/2} \) guarantees that \( \bar{F}_n(x_n) \) has orthonormal columns, while \( U \) rotates the set of vectors to align with \( \bar{G}_n(y_n) \). By symmetry, \( B = V^\top C_g^{-1/2} \), where \( V \) are the right singular vectors of \( C_g^{-1/2} \).

\[ \text{Recall that the singular value of } C_f^{-1/2}C_{fg} \text{ is the square root of the eigenvalues of } (C_f^{-1/2}C_{fg})^\top(C_f^{-1/2}C_{fg}) = C_{fg}^\top C_f^{-1/2}C_{fg}. \]
of \( L \). The produced matrices \( F_n(x_n) = A \tilde{F}_n(x_n) \) and \( G_n(y_n) = B \tilde{G}_n(y_n) \) satisfy
\[
\frac{1}{n} F_n(x_n)^\top F_n(x_n) = \frac{1}{n} G_n(y_n)^\top G_n(y_n) = I_d, \tag{14}
\]
and \( \frac{1}{n} F_n(x_n)^\top G_n(y_n) = \Lambda \), the diagonal matrix with the estimated PICs. It should be emphasized that in the implementation and subsequent experiments, we estimate the whitening matrices \( A \) and \( B \) on the training set alone, and use them as such for test sets. For the sake of clarity, we summarize our previous discussion on whitening by Algorithm 1 in Appendix B.

3 Applications to Model Comparison

We illustrate next how the PIC-based analysis can be used to compare different models trained over the same dataset. As illustrated in (1), the principal functions precisely characterize the MSE-performance of estimating a function of \( X \) from an observation \( Y \) (and vice-versa). In fact, the PICs and principal functions can be used to reconstruct the joint distribution entirely. Assuming throughout that \( P_{X,Y} \) and \( P_X \times P_Y \) have density \( p_{X,Y} \) and \( p_X \times p_Y \) with regard to a common measure, we have (Buja, 1990, Sec. 3)
\[
p_{X,Y}(x,y) = \sum_{(f_i,g_i) \in F \times G} \sqrt{\lambda_i} f_i(x) g_i(y). \tag{15}
\]
Interestingly, the PICs and the principal functions can thus be used for classification purposes, since it follows from (15) that
\[
p_{Y|X}(y|x) = p_Y(y) \sum_{(f_i,g_i) \in F \times G} \sqrt{\lambda_i} f_i(x) g_i(y). \tag{16}
\]
This connection between PICs and classification is at the core of the application to model comparison discussed next.

3.1 Black-Box Inference of PICs of Trained Models

Let \( \mathcal{Y} \) be a finite set with cardinality \( |\mathcal{Y}| = d \) (e.g., image labels). Consider a classification model trained on i.i.d. samples drawn from \( P_{X,Y} \) that outputs a conditional distribution (belief) \( \hat{p}_{Y|X} \). This is the case, for example, of neural networks with a softmax output layer, logistic regression and random forests. The belief \( \hat{p}_{Y|X} \) can itself be viewed as a distribution, or a “noisy channel”: one would hope that \( \hat{p}_{Y|X} \approx p_{Y|X} \). This is indeed the desideratum, for example, of models trained under log-loss. We demonstrate next that the PICs of the “virtual” distribution \( p_X \hat{p}_{Y|X} \) can be estimated directly from data, and is useful for comparing black-box models.

Further assume we are given fresh input samples \( \{x_k\}_{k=1}^n \) generated i.i.d. from the input distribution \( p_X \). We do not have access to the model except via input/output values, i.e., we cannot look into the details of the algorithm. An FG-net can be trained by minimizing (11) using the output of a model \( \hat{p}_{Y|X} \) and input samples \( \{x_k\}_{k=1}^n \). The following identities allow us to estimate the objective (11) over \( p_X \hat{p}_{Y|X} \) and thus, use the FG-nets to reconstruct the PICs and the principal functions of a given model. Observe that the \((i,j)\)-th entry of \( C_f \), the cross-correlation matrix \( C_{fg} \)
and the norm of $\tilde{g}$ in (11) can be computed using $\hat{p}_{Y|X}$ by

$$
[C_{f}]_{i,j} = E \left[ \hat{f}_i(X) \hat{f}_j(X) \right],
$$

$$
[C_{f\hat{g}}]_{i,j} = E \left[ \sum_{y \in Y} \hat{p}_{Y|X}(y|X) \hat{f}_i(X) \tilde{g}_j(y) \right],
$$

$$
E[\|g(\hat{Y})\|_2^2] = E \left[ \sum_{y \in Y} \hat{p}_{Y|X}(y|X) \sum_{i=1}^d \tilde{g}_i(y)^2 \right].
$$

(17a) (17b) (17c)

Thus, by computing empirical estimates over the samples $\{x_k\}_{k=1}^n$, we can estimate the quantities in (17) and evaluate the objective in (11). We explore this procedure in experiments in Section 5.

### 3.2 Model Decomposition

Any two models $\hat{p}_{Y|X}^{(1)}$ and $\hat{p}_{Y|X}^{(2)}$ can be decomposed according to (15) by finding the principal functions and PICs through the procedure described in the previous subsection. As we discuss next, this decomposition is especially useful for comparing two trained (black-box) models. The principal functions of the predicted labels for each model, denoted by $\hat{g}_i^{(1)}$ and $\hat{g}_i^{(2)}$, are of particular interest, and have the following intuitive interpretation: Assume that for a given sample $x$ drawn from $p_X$, two values, $\hat{g}_i^{(1)}$ and $\hat{g}_i^{(2)}$ are drawn according to $\hat{p}_{Y|X}^{(1)}(\cdot|X = x)$ and $\hat{p}_{Y|X}^{(2)}(\cdot|X = x)$, respectively. Given only $x$, which functions of $\hat{g}_i^{(1)}$ and $\hat{g}_i^{(2)}$ can be reliably estimated (in an MSE-sense) from $x$? Is there a function of $h(\hat{g}_i^{(1)})$ that can be estimated with significantly lower MSE from $x$ and $h(\hat{g}_i^{(2)})$ (and vice-versa)? From the model decomposition given in (15), if both models are very similar, there will be no such function that can be estimated with significantly lower MSE; however, if they operate very differently, it is positive that such function exists. We formalize this reasoning next, deriving a new measure for comparing models decomposed using FG-Nets.

Let $G_{(1)} = [g_1^{(1)}, \ldots, g_d^{(1)}]$ and $G_{(2)} = [g_1^{(2)}, \ldots, g_d^{(2)}]$ be the matrix of principal functions obtained from $\hat{p}_{Y|X}^{(1)}$ and $\hat{p}_{Y|X}^{(2)}$, respectively, and let

$$
\Lambda_{(1)} = \text{diag}([\lambda_1^{(1)}, \ldots, \lambda_d^{(1)}]) \quad \text{and} \quad \Lambda_{(2)} = \text{diag}([\lambda_1^{(2)}, \ldots, \lambda_d^{(2)}])
$$

be the corresponding PIC values. For a given function $h : Y \rightarrow \mathbb{R}$ such that $E[h(Y)^2] = 1$, and model $\hat{p}_{Y|X}$, we let the accuracy of reconstructing this function (in terms of MSE) be denoted by $\text{Acc}(h|\hat{p}_{Y|X})$, i.e.,

$$
\text{Acc}(h|\hat{p}_{Y|X}) = 1 - \inf_{f \in \mathcal{L}(p_X)} \frac{1}{2} \cdot E_{p_X} \left[ E_{\hat{p}_{Y|X}} \left[ (f(X) - h(Y))^2 \right] \right]
$$

(19)

For functions $f$ and $h$ such that $E[f(X)^2] = E[h(Y)^2] = 1$, the accuracy can be rewritten as $\sup_{f \in \mathcal{L}(p_X)} E[f(X)h(Y)]$, and takes value between 0 and 1. This accuracy measure can be computed from the principal functions, and used as a tool to compare the performance of two models, as shown below.

---

4 The value 0 is achieved when $f$ is a constant function and $h$ is mean-zero. To achieve the value 1, it must be that there exist a function $f$ such that $f(X) = h(Y)$ almost surely with respect to $p_{X,Y}$ (Witsenhausen, 1975).
Proposition 3. Let \( p^{(1)}_{Y|X} \) and \( p^{(2)}_{Y|X} \) be two models, then we have
\[
\sup_{h \in \mathcal{G}} \frac{\text{Acc}(h|p^{(1)}_{Y|X})}{\text{Acc}(h|p^{(2)}_{Y|X})} = \sigma^2_1,
\]
where \( \sigma_1 \) is the largest singular value of the matrix \( M = \Lambda^{1/2}_1 G^\top_1 D_Y G_2 \Lambda_2^{-1/2} \), and \( D_Y = \text{diag}(p_Y) \).

Proof. Please refer to Appendix A.2.

The result above allows us to characterize a classification algorithm beyond solely its accuracy. Indeed, if the ratio in (20) is above 1, then the model \( p^{(1)}_{Y|X} \) reconstructs a direction of variation of the label \( Y \) not captured by \( p^{(2)}_{Y|X} \). As shown in the numerical experiments in Section 5.2, such discrepancies can appear when one model is trained using features not observed by the other.

4 Correspondence Analysis, Alternating Conditional Expectations, and Deep Canonical Correlation Analysis

We briefly discuss next how the methods that name this section are theoretically equivalent: they all seek to characterize the PICs, and lie squarely within the framework described in the previous section. The connection between ACE and CA was noted by Buja [1990] and dates back to the work of Hirschfeld [1935]. We review both methods, and show their relation to DCCA.

CA [Greenacre, 1984] is an exploratory statistical technique that converts a contingency table into a specific type of graphical display that can be used to analyze dependencies between observed variables. CA aims at extracting representations which simplify interpretation of moderate sized contingency tables, and has been used in psychology, epidemiology, and many other applications (Sourial et al., 2010). In its simplest form, the CA of a two-way contingency table \( \{X_k, Y_k\}_{k=1}^n \) consists of computing the singular value decomposition of \( Q = D_X^{-1/2} P_{X,Y} D_Y^{-1/2} \), where \( D_X \) (respectively \( D_Y \)) is a diagonal matrix with empirical frequencies of \( X \) (resp. \( Y \)) (Greenacre and Hastie, 1987, Eq. (2.4)). If the entries of \( P_{X,Y} \) are the true distribution \( p_{X,Y} \), and denoting \( Q = U \Sigma V^\top \), then the columns of \( D_X^{-1/2} U \) and \( D_Y^{-1/2} V \) are the principal functions of \( p_{X,Y} \) in \( F \) and \( G \) (Calmon et al., 2017). Clearly, contingency table-based CA is infeasible for non-discrete or high dimensional data. By using the FG-net architecture presented in Section 2, we demonstrate how correspondence analysis-flavored visualizations can be produced using neural networks in the next section (see Figure 6 for an example).

Bivariate ACE [Breiman and Friedman, 1985] is an iterative algorithm used to recover non-linear functions \( f(X) \) and \( g(Y) \) that are maximally correlated whilst having zero mean and unit variance. This representation is found by iteratively computing \( E[g(Y)|X] \) and \( E[f(X)|Y] \). ACE converges only to the first principal functions in \( F \) and \( G \) corresponding to \( \lambda_1 \), with resulting correlation \( \sqrt{\lambda_1} \). This follows directly from the characterization in (7), and was observed by Buja [1990]. For large datasets (e.g. \( X \) and \( Y \) both images), iteratively computing conditional expectations is computationally intractable, and thus the FG-net is a natural counterpart for achieving this objective.

\[\text{Proposition 3. Let } p_{Y|X}^{(1)} \text{ and } p_{Y|X}^{(2)} \text{ be two models, then we have}\]
\[\sup_{h \in \mathcal{G}} \frac{\text{Acc}(h|p_{Y|X}^{(1)})}{\text{Acc}(h|p_{Y|X}^{(2)})} = \sigma^2_1,\]
where \( \sigma_1 \) is the largest singular value of the matrix \( M = \Lambda^{1/2}_1 G^\top_1 D_Y G_2 \Lambda_2^{-1/2} \), and \( D_Y = \text{diag}(p_Y) \).

\textit{Proof.} Please refer to Appendix A.2.

The result above allows us to characterize a classification algorithm beyond solely its accuracy. Indeed, if the ratio in (20) is above 1, then the model \( p_{Y|X}^{(1)} \) reconstructs a direction of variation of the label \( Y \) not captured by \( p_{Y|X}^{(2)} \). As shown in the numerical experiments in Section 5.2, such discrepancies can appear when one model is trained using features not observed by the other.

4 Correspondence Analysis, Alternating Conditional Expectations, and Deep Canonical Correlation Analysis

We briefly discuss next how the methods that name this section are theoretically equivalent: they all seek to characterize the PICs, and lie squarely within the framework described in the previous section. The connection between ACE and CA was noted by Buja [1990] and dates back to the work of Hirschfeld [1935]. We review both methods, and show their relation to DCCA.

CA [Greenacre, 1984] is an exploratory statistical technique that converts a contingency table into a specific type of graphical display that can be used to analyze dependencies between observed variables. CA aims at extracting representations which simplify interpretation of moderate sized contingency tables, and has been used in psychology, epidemiology, and many other applications (Sourial et al., 2010). In its simplest form, the CA of a two-way contingency table \( \{X_k, Y_k\}_{k=1}^n \) consists of computing the singular value decomposition of \( Q = D_X^{-1/2} P_{X,Y} D_Y^{-1/2} \), where \( D_X \) (respectively \( D_Y \)) is a diagonal matrix with empirical frequencies of \( X \) (resp. \( Y \)) (Greenacre and Hastie, 1987, Eq. (2.4)). If the entries of \( P_{X,Y} \) are the true distribution \( p_{X,Y} \), and denoting \( Q = U \Sigma V^\top \), then the columns of \( D_X^{-1/2} U \) and \( D_Y^{-1/2} V \) are the principal functions of \( p_{X,Y} \) in \( F \) and \( G \) (Calmon et al., 2017). Clearly, contingency table-based CA is infeasible for non-discrete or high dimensional data. By using the FG-net architecture presented in Section 2, we demonstrate how correspondence analysis-flavored visualizations can be produced using neural networks in the next section (see Figure 6 for an example).

Bivariate ACE [Breiman and Friedman, 1985] is an iterative algorithm used to recover non-linear functions \( f(X) \) and \( g(Y) \) that are maximally correlated whilst having zero mean and unit variance. This representation is found by iteratively computing \( E[g(Y)|X] \) and \( E[f(X)|Y] \). ACE converges only to the first principal functions in \( F \) and \( G \) corresponding to \( \lambda_1 \), with resulting correlation \( \sqrt{\lambda_1} \). This follows directly from the characterization in (7), and was observed by Buja [1990]. For large datasets (e.g. \( X \) and \( Y \) both images), iteratively computing conditional expectations is computationally intractable, and thus the FG-net is a natural counterpart for achieving this objective.
DCCA, in turn, seeks to produce maximally correlated representations of two views \( X \) and \( Y \) of an underlying variable. The objective function in DCCA is equivalent to (Wang et al., 2015, Eq. 1)

\[
\max_{\theta_F, \theta_G, A, B} \mathbb{E} \left[ \text{tr} \left( A \tilde{f}(X, \theta_F)(B \tilde{g}(Y, \theta_G))^\top \right) \right],
\]

subject to

\[
\mathbb{E}[A \tilde{f}(X, \theta_F)(A \tilde{f}(X, \theta_F))^\top] = \mathbb{E}[B \tilde{g}(Y, \theta_G)(B \tilde{g}(Y, \theta_G))^\top] = I_d,
\]

where, as before, \( \tilde{f}(X, \theta_F) \) and \( \tilde{g}(Y, \theta_G) \) are the output layers of two neural networks with parameters \( \theta_F \) and \( \theta_G \). From the characterization of the principal functions in (7), the whitened functions \( A \tilde{f}(X, \theta_F) \) and \( B \tilde{g}(Y, \theta_G) \) are exactly the principal functions defined in (7), and this optimization is essentially equivalent to (11). We also note that the objective (21) is also related to the maximal correlated CCA (Feizi and Tse, 2017, Eq. 1.4), in which the objective is to maximize the sum of eigenvalues of the covariance matrix of the nonlinear transformation of \( X \) and \( Y \) without the orthonormal constraints.

5 Numerical Experiments

This section provides numerical examples and demonstrate that the PICs and principal functions can be computed using the FG-Nets introduced in Section 2, both on synthetic and non-synthetic datasets. We also put in practice the applications discussed in Section 3. Additional experiments and implementation details are provided in the Appendix D.

5.1 Synthetic Data

The principal functions and the PICs can be expressed in closed-form for certain distributions \( P_{X,Y} \). In this section, we illustrate two cases where the PICs and the principal functions can be theoretically characterized, and compare them with the results obtained from the FG-Nets. The first case is when \( X \) and \( Y \) is discrete, and \( p_{Y|X} \) follows a binary symmetric channel (BSC) (Cover and Thomas, 2012, Chap. 1), and the second case is when \( X \) and \( P_{Y|X} \) follow Gaussian distributions.

5.1.1 Discrete Case: Binary Symmetric Channels

When \( p_{Y|X} \) is given by a discrete memoryless BSC, the PICs can be derived exactly (O’Donnell, 2014, Section 2.4) or (Calmon et al., 2017, Section 3.5). Let \( X \) be a binary string of length \( n \), and consider a binary string \( Y \) of the same length, where each bit is flipped independently with probability \( \delta \). The parameter \( \delta \), called the crossover probability, captures the noise in the mapping.
The F- and G-Nets are simple neural nets with two hidden layers with ReLU activation.

More precisely, the \( x \)-value of the bits are flipped. In other words, for \( \delta \)
PICs. We also show the CA-style plots under different crossover probability
in Figure 2 as well as the theoretical values, verifying that the proposed FG-Nets approximate the
each hidden layer. We train the FG-Nets with standard gradient descent (learning rate 0.01) across
the entire training set for 2000 epochs. The estimated PICs for the training and test set are shown
in Figure 2 as well as the theoretical values, verifying that the proposed FG-Nets approximate the
PICs. We also show the CA-style plots under different crossover probability \( \delta \) in Figure 3. When
\( \delta = 0.1 \), the noise is small, and thus \( X \) and \( Y \) are likely to be identical, while when \( \delta = 0.9 \),
most of the bits are flipped. In other words, for \( \delta = 0.1 \), the FG-Nets learn to map \( X \) to the identical
\( Y \), while for \( \delta = 0.9 \), the FG-Nets learn to map \( X \) to the \( Y \) where all bits are flipped. For a more
in-depth discussion on principal functions in this setting, please refer to Appendix D.1

5.1.2 Continuous Case: Gaussian Variables and Hermite Polynomials

When \( \mathcal{X} = \mathcal{Y} = \mathbb{R} \), \( X \sim \mathcal{N}(0, \sigma_1) \), \( Z \sim \mathcal{N}(0, \sigma_2) \) and \( Y = X + Z \), the set of functions \( \mathcal{F} \) and \( \mathcal{G} \) that
give the PICs are the Hermite polynomials (Abbe and Zheng 2012), where for \( x \in \mathbb{R} \), the Hermite
polynomial \( H_i(x) \) of degree \( i \geq 0 \) is defined as

\[
H_i(x) = (-1)^i e^{x^2} \frac{d^i}{dx^i} e^{-x^2}.
\]

More precisely, the \( i \)-th principal functions \( f_i \) and \( g_i \) are \( H_i^{(\sigma_1)} \) and \( H_i^{(\sigma_1+\sigma_2)} \) respectively, where
\( H_i^{(r)} \) denotes the generalized Hermite polynomial, defined as \( H_i^{(r)}(x) = \frac{1}{\sqrt{r}} H_i \left( \frac{x}{\sqrt{r}} \right) \), of degree \( i \) with
Figure 4: Hermite polynomials of degree 0 to 4 and outputs of the FG-Nets that approximate 0th to 4th principal functions.

Table 1: The MSE when using the FG-Net to approximate the principal functions (Hermite polynomials)

| Hermite | 0th  | 1st  | 2nd  | 3rd  | 4th  |
|---------|------|------|------|------|------|
| MSE$_f$ | 0.0000 | 0.0001 | 0.0042 | 0.0213 | 0.0522 |
| MSE$_g$ | 0.0000 | 0.0053 | 0.0197 | 0.0238 | 0.0583 |
| Theoretical PICs | 0.9984 | 0.6977 | 0.4675 | 0.2979 | 0.2113 |
| Estimated PICs | 0.9984 | 0.7007 | 0.4938 | 0.3376 | 0.2037 |

respect to the Gaussian distribution $\mathcal{N}(0, r)$, for $r \in (0, \infty)$. The PICs will then be given by the associated inner product $\mathbb{E}[H_i^{(\sigma_1)}(X)H_i^{(\sigma_1+\sigma_2)}(Y)]$.

We pick $\sigma_1 = \sigma_2 = 1$, and generate 5000 training samples for $X$ and $Y$ according to the Gaussian distribution and 1000 test samples. The FG-Nets are composed by two hidden layers with hyperbolic tangent activation, 30 units per layer. We train the FG-Nets with standard gradient descent (learning rate 0.01) across the entire training set for 8000 epochs.

In Figure 4, we show the Hermite polynomials of degrees 0 to 4 and the outputs of the FG-Nets that approximate the 0th to 4th principal functions. The output of the FG-Nets closely recover the Hermite polynomials; this can be further verified by computing the mean square difference between the approximated principal functions and the Hermite polynomials, i.e.

$$\text{MSE}_f \triangleq \mathbb{E}[(f_i(X) - H_i^{(\sigma_1)}(X))^2], \quad \text{MSE}_g \triangleq \mathbb{E}[(g_i(Y) - H_i^{(\sigma_1+\sigma_2)}(Y))^2], \quad (23)$$

Table 1 provides the mean square difference, the theoretical and estimated PICs. Since the FG-Nets approximate the Hermite polynomials, the estimated PICs are also close to its theoretical values.

5.2 Multi-view Representation Learning

In multi-view representation learning [Chaudhuri et al., 2009; Arora and Livescu, 2013], $X$ and $Y$ represent two unlabeled views of an information source. A representation for each view is produced, which is then (often) used in clustering [Wang et al., 2015]. We show next that the FG-Net architecture and the ensuing PIC analysis can be used to determine the number of clusters in this last step. We illustrate this in two datasets: noisy MNIST [Wang et al., 2015] and CIFAR-10 [Krizhevsky and Hinton, 2009]. In both examples, we extract 50 principal functions.
**Noisy MNIST.** The dataset consists of pairs (views) of $28 \times 28$ grayscale handwritten digits, with 60K/10K images for training/testing. For view 1 ($X$), each image is rotated at angles uniformly sampled from $[-\pi/4, \pi/4]$, and for view 2 ($Y$), each image is randomly rotated and random noise uniformly sampled from $[0, 1]$ is added. The network is trained for 50k epochs on the training set, with a batch size of 2048 for estimating the correlation/cross correlation matrix. It is reasonable to expect 10 significant PICs in this case (due to 10 underlying classes). As seen in Figure 5a, there is a phase transition in the PICs values precisely at 10, indicating that the dataset indeed has 10 significant (nonlinear) components.

**CIFAR-10.** This dataset contains $32 \times 32$ colored images in 10 categories, with 50K/10K images for training/testing. We sample view 1 ($X$) and view 2 ($Y$) by randomly selecting pairs from a randomly chosen category in each batch. Results are displayed in Fig. 5b. Once again we observe a transition of the PICs at around $m = 13$, indicating 13 directions that maximally preserve correlation after training. Note that these directions do not necessarily correspond to the underlying labels (e.g., may distinguish between images with dark/light background as opposed to dog/cat).

### 5.3 Large Scale Correspondence Analysis Visualizations

We use the CIFAR-10 dataset to demonstrate that CA-flavored visualizations can be produced at scale using the FG-net architecture. Here, $X$ is an image and $Y$ the corresponding label. The FG-net is trained using a similar training procedure to the multi-view case. The resulting CA plot is displayed in Fig. 6 with only the first 3 principal functions shown. Here, each image ($X$) is represented by a colored dot, with the color chosen depending on the true class of the image. For visualization, the outputs of $g_t(Y)$ is also superimposed on the corresponding plots as black dots. We also reconstruct $p_{Y|X}(Y|X)$ using (15), and obtain an accuracy on training and test set of 87.52% and 85.23%, respectively. This can be visualized by the “spread” of points in the test set which is much larger. We highlight that our objective was not to fine tune classification performance, but to illustrate the visualization capabilities of the FG-net.

### 5.4 Model Decomposition

Finally, we apply the results from Section 3 to compare different models trained on ProPublica’s COMPAS recidivism data (Angwin et al., 2016). This dataset contains information on the criminal history and demographic makeup of prisoners in Brower County, Florida from 2013–2014. From the COMPAS dataset (7214 instances), we select severity of charge, number of prior crimes, and age category to be the variables ($X$). We then consider four different models: the COMPAS algorithm,
and three models trained on the data to predict recidivism from $X$, namely random forest (RF), logistic regression (LR), and a simple neural network (NN). We note that the COMPAS algorithm is a “black box” — we do not have direct access to the underlying classifier.

For each model, we quantized the decile scores, which are integers from 0 to 9 indicating the likelihood of recidivism, into three classes: “low” (0 to 2), “medium” (3 to 6) and “high” (7 to 9), corresponding to the variable $Y (m = 3)$. Quantization was chosen to ensure that the four models output similar distributions over the three classes. The principal functions $f_i(X)$ extracted from the model are displayed in Fig. 7a. The greater “spread” of the points observed in the COMPAS dataset may be due to the fact that the COMPAS prediction instrument uses input factors that are not available in the dataset.

In Fig. 7b, we show the Acc ratio defined in (3), with $p_{Y|X}^{(1)}$ on the vertical axis $p_{Y|X}^{(2)}$ on the horizontal axis. When $p_{Y|X}^{(1)}$ and $p_{Y|X}^{(2)}$ are given by the same model, the Acc ratio is 1. We can observe that when $p_{Y|X}^{(1)}$ is the COMPAS algorithm, the Acc ratio is small, indicating there are functions of $Y$ that can be more accurately estimated by an RF, LR, or NN model, than by COMPAS algorithm. In other words, the COMPAS model appears to be a more “noisy channel”. Moreover, when comparing the three models RF, LR and NN models, the NN has largest Acc ratios, meaning the NN is able to express more functions of $Y$ that can be accurately estimated.

6 Final Remarks

We discussed a theoretical framework for analyzing finite-variance representations based on the PICs. This framework underlies not only classical statistical techniques such as ACE and CA, but recent multi-view representation learning methods such as DCCA. The PIC-based analysis is not limited
to these applications, and can potentially be used to characterize and compare representations learned by different models, as well as the evolution of models during training. Moreover, the PIC representation of conditional distributions in \cite{16} enables decomposition of a black-box model based only on the evaluation of the former on a reference dataset. The accuracy metric presented in \cite{16} is one out of many potential metrics that can be derived from this decomposition. We hope that the framework laid out here can serve as a useful tool to analyze representations across different learning models.

Acknowledgments

The authors gratefully acknowledge Mario Diaz (Arizona State University) for his helpful and insightful discussions and feedback on the results contained in this paper.

7 References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In OSDI, volume 16, pages 265–283.

Abbe, E. and Zheng, L. (2012). A coordinate system for gaussian networks. IEEE Transactions on Information Theory, 58(2):721–733.

Andrew, G., Arora, R., Bilmes, J., and Livescu, K. (2013). Deep canonical correlation analysis. In International Conference on Machine Learning, pages 1247–1255.

Angwin, J., Larson, J., Mattu, S., and Kirchner, L. (2016). Machine bias. ProPublica, May 23.

Arora, R. and Livescu, K. (2013). Multi-view cca-based acoustic features for phonetic recognition across speakers and domains. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, pages 7135–7139. IEEE.

Bach, F. R. and Jordan, M. I. (2002). Kernel independent component analysis. Journal of machine learning research, 3(Jul):1–48.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence, 35(8):1798–1828.

Benton, A., Khayrallah, H., Gujral, B., Reisinger, D., Zhang, S., and Arora, R. (2017). Deep generalized canonical correlation analysis. arXiv preprint arXiv:1702.02519.

Breiman, L. and Friedman, J. H. (1985). Estimating optimal transformations for multiple regression and correlation. Journal of the American statistical Association, 80(391):580–598.

Buja, A. (1990). Remarks on functional canonical variates, alternating least squares methods and ace. The Annals of Statistics, pages 1032–1069.

Calmon, F. P., Makhdoumi, A., Médard, M., Varia, M., Christiansen, M., and Duffy, K. R. (2017). Principal inertia components and applications. IEEE Transactions on Information Theory, 63(8):5011–5038.
Chandar, S., Khapra, M. M., Larochelle, H., and Ravindran, B. (2016). Correlational neural networks. *Neural computation*, 28(2):257–285.

Chaudhuri, K., Kakade, S. M., Livescu, K., and Sridharan, K. (2009). Multi-view clustering via canonical correlation analysis. In *Proceedings of the 26th annual international conference on machine learning*, pages 129–136. ACM.

Cover, T. M. and Thomas, J. A. (2012). *Elements of information theory*. John Wiley & Sons.

Feizi, S. and Tse, D. (2017). Maximally correlated principle component analysis. *arXiv preprint arXiv:1702.05471*.

Gebelein, H. (1941). Das statistische problem der korrelation als variations-und eigenwertproblem und sein zusammenhang mit der ausgleichsrechnung. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 21(6):364–379.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

Gower, J. C. and Dijksterhuis, G. B. (2004). *Procrustes problems*, volume 30. Oxford University Press on Demand.

Greenacre, M. and Hastie, T. (1987). The geometric interpretation of correspondence analysis. *Journal of the American statistical association*, 82(398):437–447.

Greenacre, M. J. (1984). Theory and applications of correspondence analysis.

Hirschfeld, H. O. (1935). A connection between correlation and contingency. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 31, pages 520–524. Cambridge University Press.

Horn, R. A., Horn, R. A., and Johnson, C. R. (1990). *Matrix analysis*. Cambridge university press.

Horst, P. (1961). Generalized canonical correlations and their applications to experimental data. *Journal of Clinical Psychology*, 17(4):331–347.

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

Huang, S.-L., Makur, A., Zheng, L., and Wornell, G. W. (2017). An information-theoretic approach to universal feature selection in high-dimensional inference. In *Information Theory (ISIT), 2017 IEEE International Symposium on*, pages 1336–1340. IEEE.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Makur, A. and Zheng, L. (2015). Bounds between contraction coefficients. In *Communication, Control, and Computing (Allerton), 2015 53rd Annual Allerton Conference on*, pages 1422–1429. IEEE.

Mirsky, L. (1975). A trace inequality of john von neumann. *Monatshefte für mathematik*, 79(4):303–306.
O’Donnell, R. (2014). *Analysis of boolean functions*. Cambridge University Press.

Reed, M. and Simon, B. (1980). Functional analysis.

Rényi, A. (1959). On measures of dependence. *Acta mathematica hungarica*, 10(3-4):441–451.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge university press.

Sourial, N., Wolfson, C., Zhu, B., Quail, J., Fletcher, J., Karunaninanthan, S., Bandeen-Roche, K., Béland, F., and Bergman, H. (2010). Correspondence analysis is a useful tool to uncover the relationships among categorical variables. *Journal of clinical epidemiology*, 63(6):638–646.

Tishby, N., Pereira, F. C., and Bialek, W. (2000). The information bottleneck method. *arXiv preprint physics/0004057*.

Wang, W., Arora, R., Livescu, K., and Bilmes, J. (2015). On deep multi-view representation learning. In *International Conference on Machine Learning*, pages 1083–1092.

Wang, W., Yan, X., Lee, H., and Livescu, K. (2016). Deep variational canonical correlation analysis. *arXiv preprint arXiv:1610.03454*.

Witsenhausen, H. S. (1975). On sequences of pairs of dependent random variables. *SIAM Journal on Applied Mathematics*, 28(1):100–113.
A Proofs

A.1 Proposition 2
Since $\mathbb{E}[\|\mathbf{A} \tilde{f}(X) - \tilde{g}(Y)\|_2^2] = \text{tr} \left( \mathbf{A} \mathbb{E} [\tilde{f}(X)\tilde{f}(X)^T] \mathbf{A}^T \right) - 2\text{tr} \left( \mathbf{A} \mathbb{E} [\tilde{f}(X)\tilde{g}(Y)^T] \right) + \left( \mathbb{E}[\|\tilde{g}(Y)\|_2^2] \right)$, we have
\begin{equation}
\mathbb{E}[\|\mathbf{A} \tilde{f}(X) - \tilde{g}(Y)\|_2^2] = d - 2\text{tr} \left( \mathbf{A} \mathbf{C}_f \right) + \mathbb{E}[\|\tilde{g}(Y)\|_2^2],
\end{equation}
where the last equation comes from the fact that $\text{tr} \left( \mathbf{A} \mathbb{E} [\tilde{f}(X)\tilde{f}(X)^T] \mathbf{A}^T \right) = \text{tr}(I_d) = d$. Since $\mathbf{C}_f$ is positive-definite, $\mathbf{C}_f^{-\frac{1}{2}}$ exists, and so does $\mathbf{A} = \mathbf{A} \mathbf{C}_f^{-\frac{1}{2}}$, and (9) can be alternatively expressed as
\begin{equation}
\min_{\mathbf{A} \in \mathbb{R}^{d \times d}} -2\text{tr} (\mathbf{A} \mathbf{B}) + \mathbb{E}[\|\tilde{g}(Y)\|_2^2], \quad \text{subject to } \mathbf{A}^T \mathbf{A} = I_d,
\end{equation}
where $\mathbf{B} = \mathbf{C}_f^{-\frac{1}{2}} \mathbf{C}_g$. The term $\text{tr}(\mathbf{A} \mathbf{B})$ can be upper bounded by the Von Neumann’s trace inequality (Mirsky 1975),
\begin{equation}
\text{tr}(\mathbf{A} \mathbf{B}) \leq \sum_{i=1}^{d} \sigma_{\mathbf{A},i} \sigma_{\mathbf{B},i},
\end{equation}
where $\sigma_{\mathbf{A},i}$’s and $\sigma_{\mathbf{B},i}$’s are the singular values for $\mathbf{A}$ and $\mathbf{B}$ respectively. Moreover, the upper bound can be achieved by solving the orthogonal Procrustes problem (Gower and Dijksterhuis 2004), and the optimizer is $\mathbf{A}^* = \mathbf{VU}^T$, where $\mathbf{V}$ and $\mathbf{U}$ are given by the SVD of $\mathbf{B} = \mathbf{U} \Sigma \mathbf{V}^T$. Therefore,
\begin{equation}
\text{tr}(\mathbf{A}^* \mathbf{B}) = \text{tr}(\mathbf{VU}^T \Sigma \mathbf{V}^T) = \sum_{i=1}^{d} \sigma_{\mathbf{B},i}
\end{equation}
which is the $d$-th Ky-Fan norm of $\mathbf{B}$. The desired result then follows by simple substitution.

A.2 Proposition 3
Recall that the accuracy of any given function $h(Y)$ can be computed in terms of its decomposition in the basis given by the principal functions. In particular, let
\begin{equation}
h(Y) = \sum_{i=1}^{d} \alpha_i g^{(1)}_i(Y)
\end{equation}
be the decomposition of $h$ in the basis $\mathbf{G}_{(1)}$, then its accuracy is given by $\text{Acc}(h|p_{Y|X}^{(1)}) = \sum_{i=1}^{d} \alpha_i^2 \lambda_i$. Rewriting this in matrix form, we have $\mathbf{h} = \mathbf{G}_{(1)} \mathbf{a}$, then
\begin{equation}
\text{Acc}(h|p_{Y|X}^{(1)}) = \mathbf{a}^T \mathbf{A}_{(1)} \mathbf{a}.
\end{equation}
Similarly, $h = \mathbf{G}_{(2)} \mathbf{b}$, and $\text{Acc}(h|p_{Y|X}^{(2)})$. Then, we have,
\begin{equation}
\frac{\text{Acc}(h|p_{Y|X}^{(1)})}{\text{Acc}(h|p_{Y|X}^{(2)})} = \frac{\mathbf{a}^T \mathbf{A}_{(1)} \mathbf{a}}{\mathbf{b}^T \mathbf{A}_{(2)} \mathbf{b}}
\end{equation}
\begin{equation}
= \frac{\mathbf{b}^T \mathbf{G}_{(2)}^T \mathbf{D}_y \mathbf{G}_{(1)} \mathbf{A}_{(1)} \mathbf{G}_{(2)} \mathbf{D}_y \mathbf{G}_{(1)}^T}{\mathbf{b}^T \mathbf{A}_{(2)} \mathbf{b}}
\end{equation}
\begin{equation}
= \frac{\mathbf{c}^T \mathbf{A}_{(2)}^{-1/2} \mathbf{G}_{(2)}^T \mathbf{D}_y \mathbf{G}_{(1)} \mathbf{A}_{(1)} \mathbf{G}_{(2)} \mathbf{D}_y \mathbf{G}_{(1)}^T \mathbf{A}_{(2)}^{-1/2} \mathbf{c}}{\mathbf{c}^T \mathbf{c}},
\end{equation}
where $\mathbf{c} = \mathbf{a}$.
where (31) follows by noting that \( a = G_{(2)} D Y G_{(1)}^{\top} b \), and (32) follows by the change of variable \( c = \Lambda_{(2)}^{1/2} b \). The desired result then follows by noting that (32) is the of \( M^{\top} M \) with \( M = \Lambda_{(1)}^{1/2} G_{(2)} D Y G_{(1)}^{\top} \Lambda_{(2)}^{(-1)/2} \), which is also a variational definition of the top singular value of \( M \).

### B Recovering the Principal Functions from the Outputs of FG-Nets

**Algorithm 1** Recovering \( F_n(x_n) \) and \( G_n(y_n) \) from \( \tilde{F}_n(x_n) \) and \( \tilde{G}_n(y_n) \), the output of the FG-Nets.

**Input:** \( \tilde{F}_n(x_n) \) and \( \tilde{G}_n(y_n) \)

**Output:** \( A \) and \( B \)

1. \( \tilde{F}_n(x_n) \leftarrow \tilde{F}_n(x_n) − E\left[\tilde{F}_n(x_n)\right] \)
   \( \tilde{G}_n(y_n) \leftarrow \tilde{G}_n(y_n) − E\left[\tilde{G}_n(y_n)\right] \)
   \( \triangleright \) (Remove mean)

2. \( U_f, S_f, V_f \leftrightarrow \text{SVD of } \frac{1}{n} \tilde{F}_n(x_n) \tilde{F}_n(x_n)^{\top} \)
   \( \quad U_g, S_g, V_g \leftrightarrow \text{SVD of } \frac{1}{n} \tilde{G}_n(y_n) \tilde{G}_n(y_n)^{\top} \)

3. \( C^{-1/2}_f \leftarrow U_f S^{-1/2}_f V_f^{\top} \)
   \( C^{-1/2}_g \leftarrow U_g S^{-1/2}_g V_g^{\top} \)
   \( \triangleright \) (Find inverse)

4. \( L = \frac{1}{n} (C^{-1/2}_f \tilde{F}_n(x_n)) (C^{-1/2}_g \tilde{G}_n(x_n))^{\top} \)

5. \( U, S, V \leftrightarrow \text{SVD of } L \)
   \( \triangleright \) (Find singular vectors)

6. **return** \( A = U^{\top} C^{-1/2}_f \)
   \( B = V^{\top} C^{-1/2}_g \)

### C More on Related Work

In representation learning, a wide range of visualization and analysis have been inspired by Canonical Correlation Analysis (CCA) (Hotelling, 1936). CCA is a statistical method that finds linear projections to make two random vectors maximally correlated. CCA suffers from two drawbacks: (i) it fails to capture non-linear correlation and (ii) it is limited to two random vectors.

To address the linearity limitation of CCA, kernel CCA (KCCA) (Bach and Jordan, 2002) was proposed as a way to extract non-linear correlation given a fixed kernel in reduced Hilbert spaces. Later, neural network-based approaches for CCA were proposed by Chandar et al. (2016) and Andrew et al. (2013). These methods relax the fixed kernel assumption, enabling more flexibility in finding good nonlinear representations via one and two deep networks, respectively. To remove the limitation on the number random vectors, generalized CCA (GCCA) (Horst, 1961) was introduced as a way to learn multiple linear mappings to a shared space. Most recently, deep generalized CCA (DGCCA) (Benton et al., 2017) and deep variational CCA (VCCA) (Wang et al., 2016) have combined the nonlinear and multi-view extensions of CCA. Similar methods to extend Principal Component Analysis to nonlinear settings are also used in (Peizi and Tse, 2017).

Variational autoencoders seek to learn representation of data, and are often used as regularization terms in CCA or DCCA (Chandar et al., 2016; Wang et al., 2015). The trade-off between maximizing the (nonlinear) correlation and minimizing the reconstruction error adheres to the same principle as the information bottleneck method (Tishby et al., 2000), and proves useful in practice.
## D Additional Experiment Results

### D.1 More Discussion on the Principal Functions in Section 5.1.1

The principal functions in the BSC example are actually given by the parity check functions \((\text{Calmon et al., } 2017, \text{Section } 3.5)\). Given a binary string \(x = x_1x_2 \cdots x_n\) of size \(n\) and a subset \(S \subseteq \{1, 2, \cdots, n\}\), the parity check function \(\chi_S(x) : [0, 1]^n \to \{-1, 1\}\) is defined as

\[
\chi_S(x) = \prod_{k \in S} (2x_k - 1).
\]  

(33)

To be more specific, the first five principal functions, \(f_1(\cdot), g_1(\cdot), \cdots, f_5(\cdot), g_5(\cdot)\), are \(\chi_S(\cdot)\) with \(S\) being a singleton (e.g. \(S = \{1\}\)), and the correlation between the parity check functions and principal functions give the PICs \((1 - 2\delta)^{|S|}\). Since in our case \(n = 5\) and \(\delta = 0.1\), there are \(^5\!\!C_1 = 5\) singletons that \(S\) can be, corresponding to 5 PICs of value \((1 - 0.2)^1 = 0.8\). Similarly, the next ten principal functions, \(f_6(\cdot), g_6(\cdot), \cdots, f_{15}(\cdot), g_{15}(\cdot)\), are \(\chi_S(\cdot)\) when \(S\) is a doubleton. Since there are \(^5\!\!C_2 = 10\) doubletons, there will be 10 PICs of value \((1 - 2\delta)^2 = 0.64\). Continuing this process, the rest of the principal functions can be found in the same way.

In Figure 8 we show the correlation matrix between the parity function with \(S\) as all singletons and doubletons, and the first 15 principal functions. The correlation matrix has two sub-matrices corresponding to singletons and doubletons, and the parity function of singletons and doubletons span a subspace with same PICs which can be obtained by the principal functions.

![Correlation Matrix](image_url)

Figure 8: Correlation between the parity and principal functions.
D.2 MNIST Handwritten Digits

We take $X$ as MNIST handwritten digits (LeCun et al., 1998) and $Y$ as labels from 0 to 9. The dataset has 60000 images for training and 10000 for test, where each image has $28 \times 28$ pixels in gray-scale. Since the inputs of the F-Net are images, we use two convolutional layers with output sizes 32 and 64 with filter dimension $5 \times 5$ and max pooling, a fully-connected layer with 1,024 units, and a readout layer with output size $d$. For the G-Net, the inputs are the one-hot encoded labels, and we use two hidden layers with output size 128 and 64, respectively, and a readout layer with output size $d$. We adopt ReLU activation for both FG-Nets, and train for 20k epochs with batch size equal to 128. The optimizer we use is the GradientDescentOptimizer with learning rate 0.01.

We reconstruct $p_{Y|X}(Y|X)$ from the principal functions according to (16), and achieve 99.90% in training accuracy and 99.08% in test accuracy. The PICs are reported in Table 2, and the CA-style plots of the nine principal functions extracted from training and test set are shown in Figure 9 and 10 respectively, where each the colored points corresponds to images ($X$) separated by color for each digit, and the black dots corresponds to the labels ($Y$).

D.3 Noisy MNIST Dataset

As we have seen in Section 5.2, the noisy MNIST dataset contains rotated and noisy handwritten digits. In this experiment, let $X$ be these images and $Y$ be the true labels. The network structure of the FG-Nets is exactly the same as in Section 5.2 however, we train the FG-Net with batch size 2048 and 50k epochs, and use AdagradOptimizer with learning rate 0.01. The training and test accuracy are 99.76% and 96.77% respectively. The PICs are reported in Table 3, and the CA-style plots of the nine principal functions extracted from training and test set are shown in Figure 11 and 12 respectively.
Table 2: The PICs of training and test sets for MNIST.

| PICs | 0th  | 1st  | 2nd  | 3rd  | 4th  | 5th  | 6th  | 7th  | 8th  | 9th  |
|------|------|------|------|------|------|------|------|------|------|------|
| Training | 1.000 | 0.994 | 0.993 | 0.992 | 0.991 | 0.990 | 0.989 | 0.986 | 0.985 | 0.984 |
| Test   | 1.000 | 0.998 | 0.992 | 0.985 | 0.979 | 0.970 | 0.967 | 0.956 | 0.953 | 0.953 |

Figure 9: CA-style plots of MNIST on training set.

Figure 10: CA-style plots of MNIST on test set.
Table 3: The PICs of training and test sets for noisy MNIST.

| PICs | 0th | 1st | 2nd | 3rd | 4th | 5th | 6th | 7th | 8th | 9th |
|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Training | 1.000 | 0.989 | 0.987 | 0.987 | 0.985 | 0.982 | 0.981 | 0.979 | 0.978 | 0.976 |
| Test | 1.000 | 0.957 | 0.945 | 0.944 | 0.927 | 0.925 | 0.924 | 0.921 | 0.917 | 0.903 |

Figure 11: CA-style plots of noisy MNIST on training set.

Figure 12: CA-style plots of noisy MNIST on test set.
D.4 CIFAR-10 Images

The CIFAR-10 dataset contains 50000 colored images for training and 10000 for test in 10 categories (e.g. dog, cat, truck, etc.), where each images has $32 \times 32$ pixels and three channels representing the RGB color model. We let $X$ be the images and $Y$ be the labels. For the F-Net, we use five convolutional layers with max pooling, two fully-connected layers, and a readout layer. The convolutional layers have output size 128, and the filter dimension is $3 \times 3$; the two fully-connected layers have output sizes 384 and 192. The G-Net has the same architecture as the one we use for training MNIST (Section D.2). We train with batch size 256 and epochs 10,000, and adopt GradientDescentOptimizer with learning rate 0.01. The training and test accuracy are 87.52% and 85.23% respectively. The PICs are reported in Table 4 and the CA-style plots of the nine principal functions extracted from training and test set are shown in Figure 13 and 14 respectively, where again each colored point corresponds to an image ($X$) differentiated by color for each class, and the black dots correspond to the labels ($Y$).

D.5 The ProPublica COMPAS Dataset

The description of the COMPAS dataset was provided in Section 5.4. We use neural networks with only one hidden layer of 20 units for both F-Net and G-Net, and train with a batch size of 256 for 20000 epochs using AdamGradient optimizer with learning rate 0.01. In Figure 15, we show CA-style plots evaluated on the training set for different models described in the paper, which give similar observations as Figure 7 in the main paper. Moreover, we report the PICs and accuracy (Acc.) of training and test sets of all models in Table 5. In Figures 16 we filter out African-American with and without recidivism (a label collected in the COMPAS dataset), and plot the CA-style plot. The plot shows that for African-American, the predicted decile scores align with the likelihood of recidivism. However, in Figure 17 we filter out Caucasian with and without recidivism, and it indicates that the predicted decile scores of Caucasian do not match with the likelihood of recidivism of them.
Table 4: The PICs of training and test sets for CIFAR-10.

| PICs | 0th | 1st  | 2nd  | 3rd  | 4th  | 5th  | 6th  | 7th  | 8th  | 9th  |
|------|-----|------|------|------|------|------|------|------|------|------|
| Training | 1.000 | 0.996 | 0.996 | 0.995 | 0.994 | 0.994 | 0.994 | 0.993 |      |      |
| Test   | 1.000 | 0.837 | 0.800 | 0.752 | 0.746 | 0.739 | 0.722 | 0.584 | 0.562 | 0.487 |

Figure 13: CA-style plots of CIFAR-10 on training set.

Figure 14: CA-style plots of CIFAR-10 on test set.
Table 5: PICs and accuracy of training and test sets of different models for COMPAS.

|       | 0th PIC | 1st PIC | 2nd PIC | Acc.   | 0th PIC | 1st PIC | 2nd PIC | Acc.   |
|-------|---------|---------|---------|--------|---------|---------|---------|--------|
| COMPAS| 1.000   | 0.689   | 0.374   | 65.32% | 1.000   | 0.692   | 0.342   | 63.23% |
| RF    | 1.000   | 0.897   | 0.704   | 72.38% | 1.000   | 0.914   | 0.733   | 68.52% |
| LR    | 1.000   | 0.978   | 0.931   | 68.67% | 1.000   | 0.986   | 0.918   | 69.54% |
| NN    | 1.000   | 0.954   | 0.872   | 68.34% | 1.000   | 0.966   | 0.870   | 68.88% |

(a) Correspondence analysis on COMPAS dataset

(b) Acc ratio

Figure 15: The COMPAS dataset on different models for training set.

(a) No recidivism

(b) Recidivism

Figure 16: CA-style plot of Recidivism of African-American.

(a) No recidivism

(b) Recidivism

Figure 17: CA-style plot of Recidivism of Caucasian.