Towards Scalable, Flexible, and Interpretable Self-Supervised Learning for Multiview Biomedical Data

by

James Chapman

July 2024

A dissertation in partial fulfilment

of the requirements for the degree of

Doctor of Philosophy

i4health CDT
University College London

Declaration

I, James Chapman, confirm that the work presented in my thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Imagine a world where images of you, data from your smartwatch, and your electronic health records could seamlessly integrate to paint a comprehensive picture of your health. Now, envision this on a global scale, where vast amounts of diverse biomedical data are harnessed to target drug trials, personalize treatment, and improve the lives of millions. This is the promise of multiview learning in the era of big data, but it comes with a significant challenge: How can we effectively integrate and analyze these complex, heterogeneous data sources?

The need for novel methods to tackle this challenge is paramount. Traditional approaches often struggle with the sheer scale and intricacy of modern biomedical datasets, limiting our ability to uncover crucial insights and advance personalized medicine. This thesis addresses this critical need by developing cutting-edge machine learning techniques that leverage the power of self-supervised and multiview learning, focusing on improving Canonical Correlation Analysis (CCA) for enormous datasets with rich structure and complex, possibly non-linear relationships.

The primary contributions of this thesis are fourfold. First, a framework for regularised CCA using structured priors is developed, enhancing the interpretability of the results. Second, simulated data generation methods for CCA are unified under a latent variable model perspective, improving our understanding of the relationship between loadings and weights in CCA. Third, a new gradient descent approach for CCA and other generalised eigenvalue problems is formulated, tailored for large datasets. Finally, this gradient descent approach is extended to Deep CCA and Joint Embedding Self-Supervised Learning, enabling the integration of diverse data sources using modern deep learning techniques. Finally, we make all of our code and simulated data publicly available, ensuring that our research is reproducible and accessible to the wider scientific community.

Impact Statement

The research presented in this thesis has the potential to significantly advance the field of representation learning, particularly in the context of integrating diverse, large-scale biomedical data. By developing novel methods for canonical correlation analysis (CCA) and its extensions, this work addresses the critical challenge of uncovering meaningful patterns and relationships in complex, high-dimensional datasets.

Within academia, the theoretical contributions of this thesis will enable researchers to scale CCA methods to much larger datasets, a crucial development as access to extensive biomedical data becomes increasingly common. This will facilitate the discovery of new insights and knowledge across various domains, from neuroscience to genomics, ultimately leading to a deeper understanding of human health and disease.

Beyond the academic sphere, the impact of this research extends to numerous real-world applications. The high-quality, open-source implementations of several CCA methods developed as part of this thesis will promote reproducible research and widespread adoption by the Python community, which has become the de facto standard for data science and machine learning. By providing accessible tools and frameworks, this work democratizes the use of advanced representation learning techniques, allowing practitioners and researchers from diverse backgrounds to harness the power of these methods in their own domains.

Through this mechanism, the work presented in this thesis has already demonstrated impact in fields as varied as process monitoring, geothermal flow, and medical imaging. As more researchers and practitioners adopt these tools and techniques, we anticipate far-reaching implications for industries such as healthcare, where improved integration and analysis of biomedical data could lead to earlier disease detection, personalized treatment plans, and enhanced patient outcomes.

In summary, by pushing the boundaries of representation learning and providing practical, open-source tools for the research community, this thesis has the potential to accelerate discovery and innovation across a wide range of domains, with particularly profound implications for advancing our understanding of human health and well-being.

Acknowledgements

Thanks to my supervisors, Professor Janaina Mourao-Miranda and Professor John Shawe-Taylor, for their contributions. I am very grateful to the EPSRC UCL Centre for Doctoral Training (CDT) in Intelligent, Integrated Imaging in Healthcare (i4Health) and NIHR UCLH Biomedical Research Centre for funding this research. Thanks to G-Research for funding my trip to NeurIPS 2023 to present my work.

To my incredible friends and coauthors, Ana and Lennie, I couldn't have done this without you. Ana, you kept our paper alive when I had given up on it (as well as the PhD). Lennie, your brutal honesty about the work being rubbish made it infinitely better. Florence, I was honored to be included on Fusili and for the hard lessons you taught me in marketing.

To members of the Machine Learning for Neuroimaging group at UCL, the Centre for Medical Imaging Computing, and all of the friends from 90 High Holborn. Cemre, I wish we had more opportunities to collaborate, but I am grateful for your words of advice before you left. Agoston, I was inspired by your immense knowledge of the field. Rick, you were everpresent in the office whenever the pandemic allowed, and I appreciated your offers to get lunch together. To the Mojo Dojo Casa House, thank you for making my NeurIPS 2023 experience unforgettable.

To the boat clubs of University College London, University of London, and Vesta Rowing Club who have provided an outlet that gave me a sense of progress, purpose, and community, even when academia had me down. Thanks also to all of my friends who have listened to me complain about my PhD. And thanks to the birds outside the window for the entertainment you provided during breaks from the desk while working from home.

Thanks to my mum and dad. Obviously this has been a bit of a rollercoaster, but I am grateful to you for supporting me through it. I will be proud to (just about) be the second Dr Chapman in the family.

And finally, with lots of love to Rebecca. On about our third date, you came to visit Leamington Spa to scout out PhDs. You have experienced every good and bad moment. You have been proud of me. We got through this together and we will get through the next thing together too.

List of Publications

First Author Peer Reviewed Conference Proceedings

Chapman, J., L. Wells, and A. L. Aguila (2024). *Unconstrained Stochastic CCA: Unifying Multiview and Self-Supervised Learning*.

First Author Peer Reviewed Conference Workshop and Abstract

Chapman, J. and L. Wells (2023). “CCA with Shared Weights for Self-Supervised Learning”. In: *NeurIPS 2023 Workshop: Self-Supervised Learning - Theory and Practice*. URL: `https://openreview.net/forum?id=7rYseRZ7Z3`.

James Chapman Janaina Mourao-Miranda, J. S.-T. (2023). *A Framework for Regularised Canonical Correlation Analysis by Alternating Least Squares*.

First Author Pre-Print

Chapman, J., A. L. Aguila, and L. Wells (2022). “A Generalized EigenGame with Extensions to Multiview Representation Learning”. In: *arXiv preprint arXiv:2211.11323*.

Co-Authored Peer Reviewed Journal

Adams, R. A. et al. (2024). “Voxel-wise multivariate analysis of brain-psychosocial associations in adolescents reveals six latent dimensions of cognition and psychopathology”. In: *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*. ISSN: 2451-9022. DOI: `https://doi.org/10.1016/j.bpsc.2024.03.`

006. URL: https://www.sciencedirect.com/science/article/pii/S2451902224000855.

Mihalik, A., J. Chapman, et al. (2022). “Canonical correlation analysis and partial least squares for identifying brain-behaviour associations: a tutorial and a comparative study”. In: *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*.

Co-Authored Peer Reviewed Conference Proceedings

Lawry Aguila, A., J. Chapman, and A. Altmann (2023). “Multi-modal Variational Autoencoders for Normative Modelling Across Multiple Imaging Modalities”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer Nature Switzerland Cham, pp. 425–434.

Lawry Aguila, A., J. Chapman, M. Janahi, et al. (2022). “Conditional VAEs for Confound Removal and Normative Modelling of Neurodegenerative Diseases”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer Nature Switzerland Cham, pp. 430–440.

Software

Mihalik, A., N. Winter, et al. (Oct. 2022). *CCA/PLS Toolkit*. Version 1.0.0. URL: https://github.com/anaston/cca_pls_toolkit.

Townsend, F., J. Chapman, and J. Cole (Nov. 2023). *florencejt/fusilli: Fusilli v1.0.0*. Version v1.0.0. DOI: 10.5281/zenodo.10228564. URL: https://doi.org/10.5281/zenodo.10228564.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Acronyms

ABCD Adolescent Brain Cognitive Development.

ABIDE Autism Brain Imaging Data Exchange.

ADNI Alzheimer’s Disease Neuroimaging Initiative. 10, , 55–57, 69, 72, 73, 77, 78, 83, 214

API Application Programming Interface.

CCA Canonical Correlation Analysis. , 28, 41–44, 46, 47, 53, 58, 97, 103, 104, 107, 113, 114

CPU Central Processing Unit.

DCCA Deep Canonical Correlation Analysis. , 43

ELBO Evidence Lower Bound.

EY Eckart-Young.

fMRI functional Magnetic Resonance Imaging.

FRALS Flexible Regularised Alternating Least Squares. , 68, 69

GCD Greatest Common Divisor.

GEP Generalized Eigenvalue Problem. , 35, 36, 41, 44, 45

GFA Group Factor Analysis. , 103, 104

GPU Graphics Processing Unit.

HCP Human Connectome Project. 10, , 55–57, 69, 71, 73, 77, 78, 83, 211

ICA Independent Component Analysis.

KCCA Kernel Canonical Correlation Analysis.

LCM Least Common Multiple.

LDA Linear Discriminant Analysis.

MCCA Multiset Canonical Correlation Analysis. , 44

ML Machine Learning.

MRI Magnetic Resonance Imaging. , 67

PCA Principal Component Analysis. , 36–39, 41

PLS Partial Least Squares. , 39–44, 114, 115

RCCA Regularized Canonical Correlation Analysis.

rs-fMRI Resting-State Functional Magnetic Resonance Imaging.

SGD Stochastic Gradient Descent.

SGHA Stochastic Generalized Hebbian Algorithm.

sMRI Structural Magnetic Resonance Imaging.

sPLS Sparse Partial Least Squares.

SSL Self-Supervised Learning.

SVD Singular Value Decomposition.

SVM Support Vector Machine.

UKBB UK Biobank.

VAE Variational Autoencoder.

Symbols List

$W^{(i)}$ The matrix of loadings for the i-th view. The jk-th element of this matrix is given by $w_{jk}^{(i)}$..

$X^{(i)}$ The ith view of the data, represented as a matrix of random variables. $X^{(i)} \in \mathbb{R}^{D_i}$ where D_i is the dimensionality of the ith view..

$Z^{(i)}$ The learned K-dimensional representation for the ith view of the data. $Z^{(i)} = f^{(i)}(X^{(i)}; \theta^{(i)})$ where $f^{(i)}$ is a function parameterized by $\theta^{(i)}$..

$\mathrm{CCA}_K(X^{(1)}, X^{(2)})$ The vector of the top K canonical correlations obtained from Canonical Correlation Analysis (CCA) applied to views $X^{(1)}$ and $X^{(2)}$. It is defined as $\mathrm{CCA}_K(X^{(1)}, X^{(2)}) := (\rho_k)_{k=1}^{K}$, where ρ_k is the kth canonical correlation..

$\mathrm{MCCA}_K(X^{(1)}, \ldots, X^{(I)})$ The vector of the top K generalized eigenvalues obtained from Multiview Canonical Correlation Analysis (MCCA) applied to views $X^{(1)}, \ldots, X^{(I)}$. It is defined as $\mathrm{MCCA}_K(X^{(1)}, \ldots, X^{(I)}) = (\lambda_1, \ldots, \lambda_K)$, where λ_k is the kth generalized eigenvalue..

$\mathrm{PLS}_K(X^{(1)}, X^{(2)})$ The vector of the top K singular values obtained from Partial Least Squares (PLS) applied to views $X^{(1)}$ and $X^{(2)}$. It represents the covariances between the learned latent variables..

Σ_{ii} The population covariance matrix of the random variables associated with view i. $\Sigma_{ii} = \mathrm{Cov}(X^{(i)})$..

Σ_{ij} The population cross-covariance matrix between the random variables associated with view i and view j. $\Sigma_{ij} = \mathrm{Cov}(X^{(i)}, X^{(j)})$..

λ The generalized eigenvalues obtained when solving the generalized eigenvalue problems that arise in PLS and CCA. In CCA, these are known as the canonical correlations..

ρ_k The kth canonical correlation obtained from CCA. ρ_k is the kth element of the vector $\mathrm{CCA}_K(X^{(1)}, X^{(2)})$..

$\theta^{(i)}$ The parameters of the function $f^{(i)}$ used to learn the representation $Z^{(i)}$ from the ith view $X^{(i)}$..

$u_j^{(i)}$ The weight of the j-th feature in the i-th view for a latent variable..

$w_j^{(i)}$ The loading of the j-th feature in the i-th view on the k-th latent variable..

Definitions

$U^{(i)}$ The matrix of weights for the i-th view. The jk-th element of this matrix is given by $u^{(i)}_{jk}$..

canonical correlations In the context of CCA, the generalized eigenvalues λ are referred to as canonical correlations. They represent the strength of the linear relationship between the learned representations of the two views. The goal of CCA is to find the weights that maximize these canonical correlations.

covariance matrix A covariance matrix captures the relationships between variables in a dataset. The notation Σ_{ij} represents the population covariance matrix between the variables in view i and view j. Each element of this matrix, $\Sigma_{ij}(a, b)$, measures how much the a-th variable in view i and the b-th variable in view j change together. A positive covariance indicates that the variables tend to increase or decrease together, while a negative covariance indicates that they tend to move in opposite directions. Σ_{ii} represents the covariance matrix within view i, capturing the relationships between variables in the same view. These matrices are essential for understanding the structure of the data and are used in subspace learning algorithms like CCA and PLS to find common patterns across views. , 35, 37, 38, 43–46

generalized eigenvalue problem A Generalized Eigenvalue Problem (GEP) is defined by two symmetric matrices $A, B \in \mathbb{R}^{D\times D}$ and is characterized by the set of solutions to the equation $Au = \lambda Bu$, where $\lambda \in \mathbb{R}$ and $u \in \mathbb{R}^D$ are called generalized eigenvalue and generalized eigenvector, respectively. Many classical subspace learning algorithms, including CCA and PLS, can be formulated as GEPs constructed from covariance matrices.

latent variables Latent variables, also referred to as representations, are the low-dimensional variables Z_k computed as linear transformations of the input variables using the weights, i.e., $Z_k = X_k u_k$. They aim to capture meaningful structures in the data that are not directly observed. , 35

loadings The Pearson correlation between a feature and a latent variable, given by $\mathrm{Corr}(X_j^{(i)}, Z_k)$. It measures the strength of the linear relationship between a feature and a latent variable, with higher absolute values indicating a stronger relationship. Loadings are invariant to certain transformations of the data matrix, such as scaling, duplication, and linear combinations of columns.. 13, 15, 35, 113, 114, 117–122, 124, 125, 212–215

norm A norm is a function that assigns a non-negative length or size to a vector in a vector space. Common norms include the L1 norm (sum of absolute values) and the L2 norm (Euclidean norm)..

representations The representations or latent variables Z_k are computed as linear transformations of the input variables using the weights, i.e., $Z_k = X_k u_k$. They aim to capture meaningful low-dimensional structures in the data. In the CCA literature, they are sometimes referred to as canonical variables. 12, 29–31, 34–37, 39, 77, 79, 90, 142, 144, 145

sample covariance matrix In practice, we often don't have access to the true population covariance matrices, which would require knowing the data distribution. Instead, we estimate these matrices from the available data samples. The sample covariance matrix, denoted as $\hat{\Sigma}^{(ij)}$, is calculated by averaging the products of the centered data points (i.e., data points with the mean subtracted) across all samples. These sample covariance matrices serve as approximations of the true population covariance matrices and are used in place of them when applying subspace learning algorithms like CCA and PLS to real-world datasets. , 47

views Views refer to the different sets of variables or modalities that describe the same underlying phenomena or objects. In the context of multiview learning, methods like PLS and CCA aim to find common latent structures that explain the relationships between these views. The term "view" is used to emphasize the distinct nature of these variable sets, which can come from different data sources or represent different aspects of the data. , 29–31, 33, 38, 39, 41, 44, 48, 53

weights When the functions f are linear, the weights u_k are used to compute the representations or latent variables as $Z_k = X_k u_k$. The weights define the linear transformation from the input variables to the latent space. 12, 15, 16, 35, 41, 44, 58, 59, 66, 75, 77–79, 88, 90, 125, 144, 145, 214, 215

Chapter I

Introduction

It was June 2021, and I had self-referred to the Community Living Well service in London, UK, seeking help for my mental health. Each week, I met with my therapist and dutifully filled out the questionnaires, rating my mood and answering questions about my well-being. Yet, I couldn't shake the feeling that these snapshots were inadequate in capturing the complexity of my mental state. As a keen sportsperson, I relied on my Garmin watch to track my heart rate, sleep, and activity levels, providing a continuous stream of biometric data that painted a more nuanced picture of my physical health. Moreover, as a type 1 diabetic, my continuous glucose monitor offered real-time insights into my blood sugar levels, helping me fine-tune my insulin management. These diverse data streams, each offering unique perspectives on my overall health, highlight the potential of learning meaningful representations from disparate data sources to gain a more comprehensive understanding of an individual's well-being.

In biomedical research, there is a growing need to develop methods that can effectively combine and analyze data from various sources, such as electronic health records, imaging data, and patient-reported outcomes. By leveraging the power of self-supervised learning, a machine learning approach that learns from unlabeled data, we can potentially uncover hidden patterns and relationships in these complex datasets. Self-supervised learning is particularly well-suited for this task, as it can learn robust and generalizable representations from vast amounts of unlabeled data, which is abundant in the biomedical domain.

This thesis focuses on developing and applying novel machine learning methods to address the challenge of integrating diverse health metrics through representation learning; that is, learning meaningful low-dimensional representations from complex, high-dimensional, and potentially multimodal data sources. A key approach explored in this work is Canonical Correlation Analysis (CCA), a powerful multiview learning technique that aims to find linear transformations of two or more datasets such that the transformed variables are maximally correlated. By learning these transformations, CCA can uncover latent structures and relationships between disparate data sources, making it a valuable tool for representation learning in the biomedical domain. Through improved methods for multiview and self-supervised learning, particularly centered around CCA, we hope to improve the analysis and comprehension of biomedical data, ultimately enhancing our ability to understand and manage personal health. We will focus our attention in general on the two-view case, as it is the most common scenario in practice, but the methods developed in this thesis can be extended to multiple views.

The main contributions of this thesis are fourfold:

1. Developing a framework for regularized Canonical Correlation Analysis using structured priors to learn more interpretable and biologically meaningful representations;
2. Unifying simulated data generation methods for CCA under a latent variable model perspective to facilitate the evaluation and comparison of representation learning algorithms;
3. Formulating a new gradient descent approach for CCA and other generalized eigenvalue problems, tailored for learning representations from large datasets;
4. Extending the gradient descent approach to Deep CCA and Joint Embedding Self-Supervised Learning to learn more complex representations from complex, high-dimensional data;
5. Developing CCA-Zoo, an open-source Python package for Canonical Correlation Analysis, which provides a unified interface for various CCA methods and facilitates their application in representation learning.

These contributions have significant practical implications, from aiding in the diagnosis and treatment of mental health and neurological disorders to enabling efficient analysis of extensive health databases like the UK Biobank (Biobank, 2014).

Thesis Structure

The thesis is structured as follows:

- **Chapter II** reviews multiview and self-supervised learning techniques, focusing on their application in learning meaningful representations from biomedical data.
- **Chapter III** introduces a method to regularize CCA using structured priors, demonstrated with Human Connectome Project and Alzheimer's Disease Neuroimaging Initiative data to showcase the potential of learning structured representations.
- **Chapter IV** examines the relationship between loadings and weights in CCA, using simulated data to show the advantages of loadings for interpreting learned representations.
- **Chapter V** presents a new gradient descent algorithm for generalized eigenvalue problems, tailored for learning representations from large datasets, demonstrated with Multiview CCA and PLS. We show how our algorithm can be applied to large datasets, using the UK Biobank as an example.
- **Chapter VI** extends the algorithm from Chapter V to deep learning, showing its application in scaling deep CCA to learn hierarchical representations from complex, high-dimensional data. We demonstrate state-of-the-art results on CIFAR-10 and CIFAR-100 benchmarks, illustrating the potential of Deep CCA in Self-Supervised Learning.
- **Chapter VII** introduces CCA-Zoo, a Python package implementing the methodologies of this thesis, and discusses its role in the Python ecosystem and biomedical research, particularly in facilitating representation learning.
- **Chapter VIII** discusses the implications, challenges, and future directions for the research presented in this thesis.

Through this thesis, we aspire to bridge the gap between the potential of biomedical data and the current capabilities of analytical methods. By developing novel, scalable, and interpretable machine learning approaches for representation learning, we aim to unlock the full potential of diverse health metrics, paving the way for advancements in biomedical research and personalized healthcare.

Chapter II

Background: Multiview Machine Learning: Concepts, Methods, and Limitations

> Principal Component Analysis is a dimensionally invalid method that gives people a delusion that they are doing something useful with their data. If you change the units that one of the variables is measured in, it will change all the "principal components"! It's for that reason that I made no mention of PCA in my book.
>
> Professor David MacKay

Contents

1 Introduction

This chapter provides the foundational knowledge needed to understand the thesis as a whole, while the individual chapters will provide more specific background information as needed.

2 Machine Learning and Multiview Learning

Machine learning encompasses methods that enable models to learn patterns and make decisions from data. Machine learning methods are typically categorized by a training set of data, which is used to learn a model, and a test set of data, which is used to evaluate the model.

Arguably the most common machine learning paradigm is supervised learning, where the training data consists of pairs of inputs and outputs, and the model learns to predict a function that maps the inputs to the outputs. This function is then used to predict the outputs for new inputs. The goal of supervised learning is to learn a function that generalizes well to new data, i.e., to make accurate predictions on unseen data. At the heart of many machine learning algorithms lies the concept of representation learning - the process of automatically discovering and extracting meaningful features or representations from raw data. Representation learning aims to transform high-dimensional, complex data into a lower-dimensional space that captures the most salient aspects of the original data. This process not only reduces computational complexity but also uncovers hidden structures and relationships within the data, which can then be leveraged for various tasks such as classification, regression, or clustering.

Unsupervised and self-supervised learning are common machine learning paradigms that often involve learning low-dimensional *latent* representations of the input data. In these paradigms, the training data consists of inputs only, and the model learns to find patterns or structure in the data without explicit output labels. *Downstream* models can use these learned representations as their inputs rather than the original data.

While the distinction between unsupervised and self-supervised learning is sometimes blurred, unsupervised learning has typically been used to describe dimensionality reduction, clustering, and generative modeling algorithms. Self-supervised learning (SSL) describes a special case of unsupervised learning where the system derives *labels* from the data itself (Balestriero, Ibrahim, et al., 2023). The cornerstone of SSL is the concept of a *pretext task*, a learning task created from the data that trains the model to capture useful features or representations. Most famously, SSL is the backbone to the success of Large Language Models (Vaswani et al., 2017) and in particular ChatGPT (OpenAI, 2021), a language model trained on a pretext task of predicting masked words in a sentence. SSL methods have also recently been shown to outperform supervised methods for certain computer vision tasks for large datasets (Goyal et al., 2019).

2.1 Multiview Machine Learning

This thesis is focussed on multiview machine learning. Practically, our results will mostly focus on the two-view case, but the methods we develop are generalizable to more than two views. Here, data from different sources or modalities, referred to as views, such as neuroimaging, genomics, and clinical records, are analyzed collectively to unveil underlying patterns. Multiview machine learning encompasses a variety of techniques aimed at learning from data that have multiple sources or modalities, also known as views. These techniques broadly fall into categories of supervised and self-supervised multiview learning, with some algorithms straddling

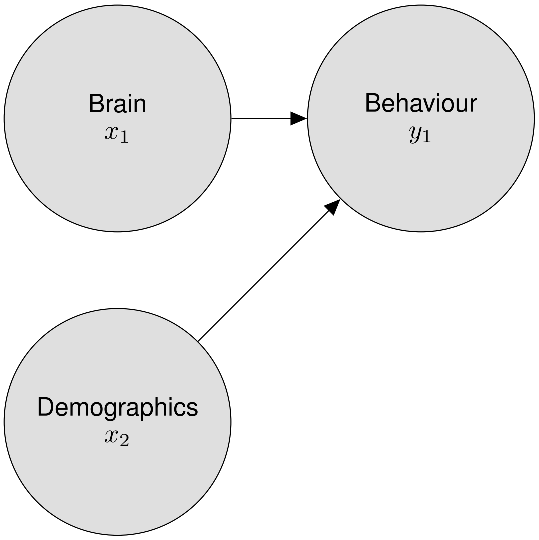


Figure II.1: ***Supervised multiview learning in mental health:*** An example of how different views (brain activity and demographics) can be fused to predict a target variable (behavior) in a supervised learning setting.

the boundary between the two.

2.1.1 Supervised Multiview Learning

In supervised multiview learning, the goal is to fuse information from multiple distinct views or feature sets to improve the predictive performance of a model. This approach involves integrating the various views, often using one view as the target variable and the others as predictors. The model learns to combine the information from the different views to make more accurate predictions than would be possible using any single view alone.

For instance, in the context of mental health, we can consider behavioral data as a dependent variable influenced by multiple independent variables like brain activity and demographics. Figure II.1 illustrates this concept, where behavioral patterns (y_1) are predicted based on features from brain activity (x_1) and demographic information (x_2). The model learns to fuse the information from the brain and demographic views to form a more comprehensive understanding of the factors influencing behavior.

Multiple Kernel Learning (MKL) (Gonen and Alpaydin, 2011) is a prominent example of supervised multiview learning, where the algorithm learns to combine kernel representations of the different views. By fusing the information from the various kernels, MKL enhances the model's predictive capabilities compared to using a single kernel.

With the advent of deep learning, the underlying concept of MKL has been extended to deep learning architectures. These architectures enable the model to learn and fuse representations from various views more effectively (Guo, J. Wang, and S. Wang, 2019). The deep learning models can automatically learn the most informative features from each view and combine them in non-linear ways to make predictions.

I contributed to this line of research through the software package Fusili (Townsend, Chapman, and Cole, 2023), which implements a number of deep-learning based multi-modal data fusion methods for supervised learning. Fusili provides a flexible framework for building models that can fuse information from various data modalities, such as images, text, and structured data, to make more accurate predictions.

In summary, supervised multiview learning involves fusing information from multiple views to improve predictive performance. By combining the unique perspectives offered by each view, these methods can form a more comprehensive understanding of the problem and make more accurate predictions than would be possible using any single view alone.

2.1.2 Self-Supervised Multiview Learning

In contrast to supervised multiview learning, where explicit labels guide the learning process, self-supervised multiview learning operates under the hypothesis that different views are manifestations of a shared, yet hidden, latent variable (Zong, Mac Aodha, and T. Hospedales, 2023). This approach, as evidenced in the latent variable model of mental health illustrated in Figure II.2, suggests that both neuroimaging and behavioural data are influenced by an underlying factor, such as the severity of a mental health condition, which remains unobserved.

A key challenge in self-supervised learning is designing pretext tasks to estimate this latent source from the available views. A common approach is to estimate a common low-dimensional representation of the variance in the data from both views. In most objectives of this form, this ammounts to identifying the mutual information between the views. These representations may be informative for their own sake, identifying common factors between the views, or they may be used as inputs to a downstream task, such as classification or regression.

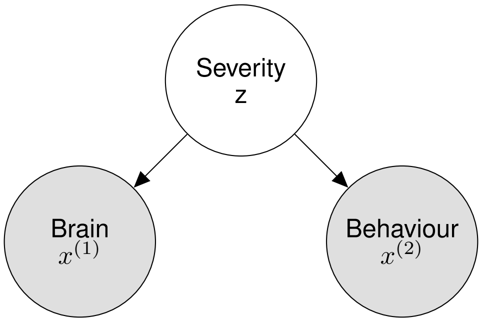


Figure II.2: ***Latent Variable Model of Mental Health:*** From this perspective the neuroimaging modality and behavioural data are both considered to have been generated with distributions conditioned on the severity of a mental health condition

2.2 Conditional Independence, Causality, and Multiview Learning

The graphical model in Figure II.1 represents the assumption that the brain and demographics are independent variables, and that the behaviour is a conditional variable, dependent on both the brain and demographics.

On the other hand, the graphical model in Figure II.2 represents the assumption that the brain and behaviour are conditionally independent given the severity of an unobserved latent mental health condition.

Reichenbach (1956) introduced the epoynmous Reichenbach's principle, which states that if two variables are correlated, then either one causes the other, or both are caused by a third variable. While the relationship between conditional independence and causality is nuanced (Pearl, 2009), it is clear that our assumptions about the causal structure of the data can inform our choice of multiview learning algorithm. In particular, we could envision a number of causal structures that could give rise to the observed data in Figure II.2:

- direct causation (brain influencing behavior or vice versa or even both)
- both being influenced by a common, possibly unobserved, cause
- no direct causal link between them

In the first case, we might be more inclined to use a supervised multiview learning algorithm to predict one view from the other. In the second case, we might be more inclined to use a self-supervised multiview learning algorithm to estimate the latent variable.

2.2.1 Complementary and Redundant Information

The nature of the information provided by different views (such as neuroimaging and behavioral data) is important for understanding multiview learning models. A particularly useful distinction is between complementary and redundant information (Nguyen and D. Wang, 2020; Lyu et al., 2021; M.-S. Chen et al., 2022). The complementary information in views offers unique insights into different aspects of the same subject. For instance, in mental health studies, neuroimaging might reveal structural changes in the brain that are not (yet) present in presented behavioural phenotypes, while behavioral data could be influenced by demographic factors that do not present as structural differences in the brain. Both these views together provide a more holistic understanding of a mental health condition. Conversely, redundant information in different views refers to overlapping or similar data presented from various angles. For instance, a specific mental health condition may manifest in both observable behavioral changes and detectable neuroimaging markers. While each view alone could suggest the presence of the condition, their combination, due to redundancy, can enhance the reliability of the diagnosis. This redundancy is not merely repetitive; it plays a crucial role in denoising and validating findings. In essence, if both neuroimaging and behavioral data independently point to the same diagnosis, the confidence in this diagnosis increases. The *Wisdom of Crowds* phenomenon, where the collective average of multiple estimates tends to be more accurate than individual estimates, exemplifies the strength of redundant information (Galton, 1907), as illustrated in Figure II.3. This principle is akin to the redundancy in multiview data, where multiple views converge to a more accurate or robust conclusion than any single view alone.

In this thesis, we will explore Canonical Correlation Analysis, a multiview learning method predicated on the assumption that different views provide complementary information about latent variables. The following sections will establish a formal framework for representation learning and motivate the use of Canonical Correlation Analysis in harnessing complementary information from multiview data.

Figure II.3: ***The Wisdom of Crowds:*** The average of multiple noisy estimates of the weight of a cow is more accurate than any individual estimate

3 Learning Representations: Definitions and Notation

Suppose we have a sequence of vector-valued random variables $X^{(i)} \in \mathbb{R}^{D_i}$ for $i \in \{1, \ldots, I\}$ We want to learn meaningful K-dimensional representations

$$Z^{(i)} = f^{(i)}(X^{(i)}; \theta^{(i)}). \tag{II.1}$$

For convenience, define $D = \sum_{i=1}^{I} D_i$ and $\theta = \left(\theta^{(i)}\right)_{i=1}^{I}$. Without loss of generality take $D_1 \geq D_2 \geq \cdots \geq D_I$. We will consistently use the superscript (i) to denote the i-th view and not as an exponentiation operation. $d \in [D_i]$ for dimensions of input variables; and $l, k \in [K]$ for dimensions of representations - i.e. to subscript dimensions of $Z^{(i)}, f^{(i)}$. Later on we will introduce total number of samples N.

We denote the inner product between two vectors a and b as $\langle a, b \rangle$, which is defined as:

$$\langle a, b \rangle = a^\top b = \sum_{i=1}^{n} a_i b_i \tag{II.2}$$

where a_i and b_i are the i-th elements of vectors a and b, respectively, and n is the dimension of the vectors.

The inner product is a measure of similarity between two vectors, with larger values indicating higher similarity. It is also related to the angle θ between the vectors, as shown in the following equation:

$$\langle a, b \rangle = |a||b| \cos(\theta) \tag{II.3}$$

where $|a|$ and $|b|$ are the Euclidean norms (lengths) of vectors a and b, respectively.

In this report, when the functions f are linear, we will typically refer to u_k as weights, $Z_k = X_k u_k$ as representations or latent variables (noting that in the CCA literature they are sometimes referred to as canonical variables (Borga, 1998) or scores (Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al., 2022)), depending on the context. We will sometimes consider a matrix $U = (u_1, \dots, u_K) \in \mathbb{R}^{D \times K}$ of weights, and a matrix $Z = (Z_1, \dots, Z_K) \in \mathbb{R}^{N \times K}$ of representations. We will refer to the Pearson correlation between features and their respective latent variable $\mathrm{Corr}(X_j^{(i)}, Z_k)$ as the loadings of $X_j^{(i)}$ on Z_k (Rosipal and Krämer, 2005; Alpert and R. A. Peterson, 1972; Borga, 1998), noting that the same concept has also been referred to as structure correlations (Meredith, 1964).

We will use the notation $\Sigma_{ij} = \mathrm{Cov}(X^{(i)}, X^{(j)})$ for the population covariance matrix between the random variables associated with view i and j. This covariance matrix captures the relationships between variables from different views. Each element of this matrix, $\Sigma_{ij}(a, b)$, measures how much the a-th variable in view i and the b-th variable in view j change together, even though they belong to different views. A positive covariance indicates that the variables from different views tend to increase or decrease together, while a negative covariance indicates that they tend to move in opposite directions. These covariance matrices play a crucial role in multiview learning algorithms as they capture the inter-view relationships that the algorithms aim to leverage.

We will also use $\Sigma_{ii} = \mathrm{Cov}(X^{(i)})$ for the population covariance matrix of the random variables associated with view i with each other. This covariance matrix captures the relationships between variables within the same view. Each element of this matrix, $\Sigma_{ii}(a, b)$, measures how much the a-th and b-th variables in view i change together. A positive covariance indicates that the variables within the same view tend to increase or decrease together, while a negative covariance indicates that they tend to move in opposite directions. These within-view covariance matrices are essential for understanding the structure of the data in each view and are used in conjunction with the inter-view covariance matrices in multiview learning algorithms.

Many classical subspace learning algorithms can be formulated as Generalized Eigenvalue Problems GEPs constructed from covariance matrices. In the following subsection, we introduce the concept of GEPs and discuss their properties, which will be useful for understanding the optimization problems and solutions of these algorithms.

3.1 Generalized Eigenvalue Problems in linear algebra

A Generalized Eigenvalue Problem is defined by two symmetric matrices $A, B \in \mathbb{R}^{D \times D}$ (**stewart_matrix_1990**)[1]. They are usually characterized by the set of solutions to the equation:

$$Au = \lambda Bu \tag{II.4}$$

with $\lambda \in \mathbb{R}, u \in \mathbb{R}^D$, called (generalized) eigenvalue and (generalized) eigenvector respectively. When B is positive definite, then the GEP becomes equivalent to an eigen-decomposition of the symmetric matrix $B^{-1/2}AB^{-1/2}$ (Ghojogh, Karray, and Crowley, 2019). In addition, one can find a basis of eigenvectors spanning $\mathbb{R}^D$. We define a top-K subspace to be one spanned by some set of eigenvectors $u_1, \ldots, u_K$ with the top-K associated eigenvalues $\lambda_1 \geq \cdots \geq \lambda_K$. We say a matrix $U \in \mathbb{R}^{D \times K}$ defines a top-K subspace if its columns span one.

Uniqueness In GEPs, the eigenvectors u are not in general unique, but the generalized eigenvalues $1 \geq \lambda_1 \geq \lambda_2 \geq \cdots \geq 0$ are unique (Mills-Curran, 1988).

4 Classical Subspace Learning Algorithms

4.1 Principal Components Analysis

Principal Components Analysis (Hotelling, 1933) (PCA) is a classical method in unsupervised machine learning for representation learning. It is widely used for dimensionality reduction and feature extraction. The primary goal of PCA is to transform the original high-dimensional data into a new coordinate system defined by orthogonal axes, capturing the most relevant aspects of the data.

In PCA, the representations are constrained to be linear transformations of the form:

$$Z_k = Xu_k, \tag{II.5}$$

[1] more generally, A, B can be Hermitian, but we are only interested in the real case

where u_k are orthonormal basis vectors such that:

$$u_k^\top u_k = 1, \quad u_k^\top u_l = 0 \text{ for } k \neq l. \tag{II.6}$$

The primary goal of PCA is to maximize the variance of the representations Z_k, finding the directions of maximal variance in the data.

4.1.1 Optimization and Solution

Mathematically, for the first principal component, this can be formulated as:

$$u_{\text{opt}} = \underset{u}{\text{argmax}} \left(u^\top \Sigma u\right) \tag{II.7}$$

$$\text{subject to:}$$

$$u^\top u = 1$$

Where $\Sigma = \mathbb{E}[X^\top X]$ is the population covariance matrix of the single view data X.

To solve this constrained optimization problem, we can use the method of Lagrange multipliers. The key idea behind Lagrange multipliers is to transform a constrained optimization problem into an unconstrained one by incorporating the constraints into the objective function. The Lagrange multiplier, denoted by λ, can be thought of as a penalty for violating the constraint. By setting the Lagrange multiplier to a large enough value, we ensure that the optimal solution satisfies the constraint.

The Lagrangian function for the PCA optimization problem is constructed by adding the constraint multiplied by the Lagrange multiplier to the original objective function:

$$f(u, \lambda) = u^\top \Sigma u + \lambda(1 - u^\top u), \tag{II.8}$$

where λ is the Lagrange multiplier.

Intuitively, the first term in the Lagrangian represents the objective function (maximizing the variance), while the second term penalizes solutions that violate the constraint (unit norm). By finding the stationary points of the Lagrangian with respect to both u and λ, we obtain the optimal solution that maximizes the variance while satisfying the unit norm constraint.

Differentiating the Lagrangian with respect to u and setting it to zero yields the first-order conditions (which are necessary for optimality for the optimal u):

$$\Sigma u = \lambda u, \tag{II.9}$$

$$u^\top u = 1. \tag{II.10}$$

Eigenvalue Problem This transforms the problem into an eigenvalue equation for the covariance matrix Σ, which can be efficiently solved using standard libraries such as scikit-learn (Pedregosa et al., 2011).

The first principal component therefore corresponds to the eigenvector associated with the largest eigenvalue λ. Subsequent components are the remaining eigenvectors ordered by their corresponding eigenvalues.

4.1.2 Limitations

There are three major limitations of PCA that are relevant to this thesis.

1. **Scale Invariance**: as highlighted in the epigraph to this chapter, PCA is not scale invariant, meaning that the importance of a principal component can be disproportionately affected by the scale of the variables in the data. Variables measured at larger scales can dominate over those measured at smaller scales unless the data is normalized. This sensitivity to the scale of the data can lead to misleading directions that do not necessarily capture the most meaningful underlying structures.

2. **Sparsity and Interpretability**: Although PCA reduces dimensionality by projecting the data onto new axes, the resulting principal components are linear combinations of *all* the original features. This complex aggregation can make it difficult to interpret the components, especially when each component is influenced by many original variables. For this reason, sparse variants of PCA have been developed (Zou, Hastie, and Tibshirani, 2006; Zou and Xue, 2018), which aim to find sparse linear combinations of the original features; interpretable as a subset of the original features contributing to a significant proportion of the variance in the data.

3. **Multiview Data**: PCA is primarily designed for analyzing a single dataset and does not naturally accommodate multiview data, where multiple independent sets of variables (views) describe the data. While it is possible to concatenate these views into a single dataset prior to analysis, this approach does not take advantage of the potential interactions and complementary information across

the views, which can be critical for more insightful analysis in applications such as image processing, bioinformatics, and social sciences.

Nevertheless, PCA remains a popular tool in practice (Greenacre et al., 2022) and is a useful baseline for multiview learning methods, and we will use it as a point of comparison in this thesis.

4.2 Partial Least Squares

Given the inherent limitations of PCA, especially in handling multiview datasets where capturing interactive and complementary information between different data sources is crucial, Partial Least Squares (PLS) emerges as a potent alternative. PLS extends the principles of PCA to analyze two correlated views simultaneously, optimizing for the shared covariance rather than variance within a single dataset. This approach makes PLS particularly valuable in multiview settings where the goal is to uncover the latent structures that explain the relationships between views.

Partial Least Squares (Wold, 1975) aims to maximize the shared covariance between two paired sets of data, referred to as views. PLS can be seen as a generalization of PCA, where PCA becomes a special case when the two views are identical.

PLS optimizes for the dot product between the representations of two views, a measure of similarity.

$$\langle X^{(1)}u^{(1)}, X^{(2)}u^{(2)}\rangle = |X^{(1)}u^{(1)}||X^{(2)}u^{(2)}|\cos(\theta) = u^{(1)\,T}\Sigma_{12}u^{(2)} \tag{II.11}$$

Where θ is the angle between the two representations. Much like for PCA, the representations are constrained to be linear transformations of the form:

$$Z^{(i)} = X^{(i)}u^{(i)} \tag{II.12}$$

Where $u^{(i)}$ are orthonormal basis vectors such that:

$$u^{(i)\,T}u^{(i)} = 1 \tag{II.13}$$

$$u^{(i)\,T}u^{(j)} = 0 \text{ for } i \neq j \tag{II.14}$$

4.2.1 Optimization and Solution

The constrained optimization problem for PLS can therefore be formulated as:

$$u^{(1)}_{\text{opt}} = \underset{u^{(1)}}{\mathrm{argmax}}\{u^{(1)\,T}\Sigma_{12}u^{(2)}\} \tag{II.15}$$

subject to:

$$u^{(1)\,T}u^{(1)} = 1$$

$$u^{(2)\,T}u^{(2)} = 1$$

The Lagrangian for this optimization problem once again integrates the constraints as penalties:

$$f(u^{(1)},\lambda) = u^{(1)\,T}\Sigma_{12}u^{(2)} + \lambda_1(1 - u^{(1)\,T}u^{(1)}) + \lambda_2(1 - u^{(2)\,T}u^{(2)}) \tag{II.16}$$

Upon deriving the first order conditions, we get:

$$\Sigma_{21}u^{(1)} = \lambda_2 u^{(2)} \tag{II.17}$$

$$\Sigma_{12}u^{(2)} = \lambda_1 u^{(1)} \tag{II.18}$$

$$u^{(1)\,T}u^{(1)} = 1 \tag{II.19}$$

$$u^{(2)\,T}u^{(2)} = 1 \tag{II.20}$$

By substituting the constraint conditions into these equations, we find that $\lambda_1 = \lambda_2 = \lambda$ by symmetry. Further simplification yields:

$$\Sigma_{21}\Sigma_{12}u^{(2)} = \lambda^2 u^{(2)} \tag{II.21}$$

$$\Sigma_{12}\Sigma_{21}u^{(1)} = \lambda^2 u^{(1)} \tag{II.22}$$

Eigenvalue Problem Once again, we see that solving these equations will yield the $u^{(1)}$ and $u^{(2)}$ vectors as eigenvectors, this time of $\Sigma_{12}\Sigma_{21}$ and $\Sigma_{21}\Sigma_{12}$, respectively (Höskuldsson, 1988).

Generalized Eigenvalue Problem We can also represent the system of equations in matrix form as follows:

$$\begin{pmatrix} 0 & \Sigma_{12} \\ \Sigma_{21} & 0 \end{pmatrix} \begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix} = \lambda I \begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix} \tag{II.23}$$

Which is of the form $Av = \lambda Bv$. PLS is therefore also defined by the solution to a single generalized eigenvalue problem.

Given the notions of uniqueness in GEPs, the weights u are not in general unique but we can write the vector of generalized eigenvalues $(\lambda_1, \ldots, \lambda_K)$ representing covariances as:

$$\mathrm{PLS}_K(X^{(1)}, X^{(2)}) := (\lambda_k)_{k=1}^K \tag{II.24}$$

4.2.2 Limitations

Despite the advantages of PLS over PCA in handling multiview datasets, PLS has its own limitations that can impact its effectiveness in certain applications:

1. **Scale Invariance**: Similar to PCA, PLS is not scale invariant. This means that the model's outcomes are affected by the scale of the features, potentially leading to biased weights towards features with larger scale unless data is normalized.

2. **Sparsity and Interpretability**: PLS does not inherently produce sparse models. The components derived from PLS are linear combinations of all input features, which can make the model difficult to interpret, particularly in high-dimensional contexts such as genomics or text processing. Sparse PLS has also been an active area of research (Chun and Keleş, 2010; Witten, Tibshirani, and Hastie, 2009).

4.3 Canonical Correlation Analysis

Canonical Correlation Analysis (CCA) is a close relative of PLS, designed to maximize the correlation instead of covariance between representations. This focus on correlation allows for a more nuanced understanding of the relationships between views, making CCA particularly useful in scenarios where the goal is to explore how views relate on a normalized scale.

CCA achieves this by optimizing the cosine of the angle between the representations, thus normalizing the effect of the scale of the data:

$$\cos(\theta) = \frac{\langle X^{(1)}u^{(1)}, X^{(2)}u^{(2)}\rangle}{|X^{(1)}u^{(1)}||X^{(2)}u^{(2)}|} = \frac{u^{(1)\,T}\Sigma_{12}u^{(2)}}{|X^{(1)}u^{(1)}||X^{(2)}u^{(2)}|} \tag{II.25}$$

By focusing on correlation, CCA normalizes the contributions of each variable, ensuring that the analysis is not unduly influenced by the magnitude of the data. This normalization is particularly valuable in multiview settings where the scales of the data sources may differ significantly.

4.3.1 Optimization and Solution

If we constrain the norms of the representations to be equal to 1, i.e., $|X^{(1)}u^{(1)}| = |X^{(2)}u^{(2)}| = 1$, then maximizing the cosine similarity is equivalent to maximizing the numerator $u^{(1)\,T}\Sigma_{12}u^{(2)}$. This leads to the following constrained optimization problem:

$$\begin{aligned} &u_{\text{opt}} = \underset{u}{\operatorname{argmax}}\{u^{(1)\,T}X^{(1)\,T}X^{(2)}u^{(2)}\} \\ &\text{subject to:} \\ &u^{(1)\,T}\Sigma_{11}u^{(1)} = 1 \\ &u^{(2)\,T}\Sigma_{22}u^{(2)} = 1 \end{aligned} \tag{II.26}$$

By focusing on correlation and imposing unit norm constraints on the representations, CCA normalizes the contributions of each variable, ensuring that the analysis is not unduly influenced by the magnitude of the data. This normalization is particularly valuable in multiview settings where the scales of the data sources may differ significantly.

Although non-convex, numerous methods exist for solving the CCA problem, including eigendecomposition and generalized eigendecomposition solvers (Uurtio et al., 2017) and block coordinate descent via alternating least squares regressions (Golub and Zha, 1995; Sun, Ji, and Ye, 2008).

The first-order conditions derived in the same manner as the PLS case are:

$$\Sigma_{21}u^{(1)} = \lambda^{(2)}\Sigma_{22}u^{(2)} \tag{II.27}$$

$$\Sigma_{12}u^{(2)} = \lambda^{(1)}\Sigma_{11}u^{(1)} \tag{II.28}$$

$$u^{(1)\,T}\Sigma_{11}u^{(1)} = 1 \tag{II.29}$$

$$u^{(2)\,T}\Sigma_{22}u^{(2)} = 1 \tag{II.30}$$

Eigenvalue Problems Substituting the second two conditions into the first two, we get $\lambda^{(1)} = \lambda^{(2)} = \lambda$. Finally, substituting the first two conditions into each other, we find the eigenvalue problems:

$$\Sigma_{11}^{-1}\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}u^{(1)} = \lambda^2 u^{(1)} \tag{II.31}$$

$$\Sigma_{22}^{-1}\Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}u^{(2)} = \lambda^2 u^{(2)} \tag{II.32}$$

An alternative form of the CCA problem can be developed by reparameterizing $u^{(i*)} = \Sigma_{ii}^{-\frac{1}{2}}u^{(i)}$. The optimization problem then becomes:

$$u_{\text{opt}} = \underset{u}{\operatorname{argmax}}\{u^{(1)\,T}\Sigma_{11}^{-\frac{1}{2}}\Sigma_{12}\Sigma_{22}^{-\frac{1}{2}}u^{(2)}\} \tag{II.33}$$

subject to:

$$u^{(1)\,T}u^{(1)} = 1$$

$$u^{(2)\,T}u^{(2)} = 1$$

This reparameterized form will later underpin Deep Canonical Correlation Analysis (DCCA) through the matrix $T = \Sigma_{11}^{-\frac{1}{2}}\Sigma_{12}\Sigma_{22}^{-\frac{1}{2}}$. This form also shows that PLS and CCA can be made equivalent by whitening the data matrices before constructing the covariance matrices. When the number of features exceeds the number of samples ($p > n$), CCA becomes degenerate because the within-view covariance matrices cannot be inverted—contrasting with PLS, which is always computable.

Generalized Eigenvalue Problem We can also represent the system of equations in equation II.27 as a matrix equation:

$$\begin{pmatrix} 0 & \Sigma_{12} \\ \Sigma_{21} & 0 \end{pmatrix} \begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix} = \lambda \begin{pmatrix} \Sigma_{11} & 0 \\ 0 & \Sigma_{22} \end{pmatrix} \begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix} \tag{II.34}$$

Which is once again of the form $Au = \lambda Bu$. CCA, like PLS, is therefore also defined by the solution to a single generalized eigenvalue problem.

Canonical Correlations In the case of CCA, the generalized eigenvalues λ are generally called canonical correlations (Hotelling, 1935; Hotelling, 1992). Given the notions of uniqueness in GEPs, the weights u are not in general unique but we can write the vector of generalized eigenvalues or canonical correlations as:

$$\mathrm{CCA}_K(X^{(1)}, X^{(2)}) := (\rho_k)_{k=1}^K \tag{II.35}$$

4.3.2 Limitations

A major limitation of CCA is revealed by the forms in equations II.31 and equation II.33; CCA in general requires the inversion of covariance matrices, which is computationally expensive, potentially numerically unstable, and impossible when the number of features exceeds the number of samples such that the covariance matrices are not full rank.

4.4 Multiview CCA

Multiview CCA (MCCA) is an extension of CCA that handles more than two views simultaneously. Given I views $X^{(1)}, \ldots, X^{(i)}$, the goal of MCCA is to find a set of directions $u^{(1)}, \ldots, u^{(i)}$ that maximize the sum of pairwise correlations between the projections of the views onto these directions.

4.4.1 Optimization and Solution

The optimization problem for MCCA can be formulated as follows:

$$u_{\text{opt}} = \underset{u}{\operatorname{argmax}} \sum_{i=1}^{I} \sum_{j=1, j \neq i}^{I} u^{(i)\,T} \Sigma_{ij} u^{(j)} \tag{II.36}$$

subject to:

$$\sum_{i=1}^{I} u^{(i)\,T} \Sigma_{ii} u^{(i)} = 1$$

Generalized Eigenvalue Problem The MCCA optimization problem can be solved by formulating it as a generalized eigenvalue problem. The GEP for MCCA can be written in matrix form as follows:

$$\underbrace{\begin{pmatrix} 0 & \Sigma_{12} & \cdots & \Sigma_{1I} \\ \Sigma_{21} & 0 & \cdots & \Sigma_{2I} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{I1} & \Sigma_{I2} & \cdots & 0 \end{pmatrix}}_{A} \begin{pmatrix} u^{(1)} \\ u^{(2)} \\ \vdots \\ u^{(I)} \end{pmatrix} = \lambda \underbrace{\begin{pmatrix} \Sigma_{11} & 0 & \cdots & 0 \\ 0 & \Sigma_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma_{II} \end{pmatrix}}_{B} \begin{pmatrix} u^{(1)} \\ u^{(2)} \\ \vdots \\ u^{(I)} \end{pmatrix}. \tag{II.37}$$

The matrix A contains the cross-covariance matrices between the views, while the matrix B is a block diagonal matrix containing the within-view covariance matrices. The solution to the GEP gives the optimal directions $u^{(1)}, \ldots, u^{(I)}$ and the corresponding generalized eigenvalues λ.

Unified Framework The GEP formulation of MCCA can be further generalized to include ridge regularization, which helps stabilize the solution when the covariance matrices are ill-conditioned. This leads to a unified framework that encompasses both CCA and its ridge-regularized extensions.

Let $A, B_\alpha \in \mathbb{R}^{D\times D}$ be block matrices defined as follows:

$$A_{ij} = \mathrm{Cov}(X^{(i)}, X^{(j)}) \text{ for } i \neq j, \tag{II.38}$$

$$B_{\alpha,ii} = \alpha_i I_{D^{(i)}} + (1-\alpha_i)\mathrm{Var}(X^{(i)}), \tag{II.39}$$

where $\alpha \in [0,1]^I$ is a vector of ridge penalty parameters. Setting $\alpha_i = 0\,\forall i$ recovers the standard CCA, while $\alpha_i = 1\,\forall i$ yields the PLS solution.

In the case of standard CCA (i.e., $\alpha = 0$), we can define the MCCA correlation vector as:

$$\mathrm{MCCA}_K(X^{(1)}, \ldots, X^{(I)}) = (\lambda_1, \ldots, \lambda_K), \tag{II.40}$$

where $\lambda_1, \ldots, \lambda_K$ are the top-K generalized eigenvalues. These eigenvalues represent the average of the top-K correlations between each pair of views.

4.5 Linear Discriminant Analysis LDA

Linear Discriminant Analysis (LDA) can be viewed as a special case of Canonical Correlation Analysis where $X^{(2)}$ is a one-hot encoded matrix representing the class labels. This allows us to draw a connection between the unsupervised learning framework of CCA and the supervised framework of LDA(Balakrishnama and Ganapathiraju, 1998; Riffenburgh, 1957), thus expanding the understanding of both algorithms.

Intuition: In LDA, the aim is to find a lower-dimensional subspace where the classes are maximally separated. This objective can be viewed through the lens of CCA, where the optimal directions $u^{(1)}$ and $u^{(2)}$ in the original and one-hot encoded spaces aim to maximize correlation. In the LDA context, $u^{(1)}$ would maximize the separation between classes.

4.5.1 Optimization and Solution

Mathematically, LDA is reduced to solving a generalized eigenvalue problem involving the between-class scatter matrix S_B and the within-class scatter matrix S_W:

$$\hat{S_B} = \sum_{i=1}^{c} n_i(\mu_i - \mu)(\mu_i - \mu)\top$$

$$\hat{S_W} = \sum_{i=1}^{c} \sum_{x \in X_i} (x - \mu_i)(x - \mu_i)\top$$

Connection to CCA: When $X^{(2)}$ is the one-hot encoded matrix of class labels, the CCA problem effectively tries to maximize the correlation between the feature vectors and their corresponding labels. This turns out to be equivalent to maximizing the between-class variance in LDA while minimizing the within-class variance. Thus, LDA can be thought of as a constrained form of CCA, tailored to classification tasks.

This perspective unifies the two algorithms and shows that the core objective—finding meaningful relationships or directions in the data—is shared between both CCA and LDA.

4.6 Sample Covariance and Population Covariance

In the previous sections, the methods were described in terms of population covariance matrices such as $\Sigma_{11} = \mathbb{E}[X^{(1)\,T}X^{(1)}]$, $\Sigma_{22} = \mathbb{E}[X^{(2)\,T}X^{(2)}]$, and

$\Sigma_{12} = \mathbb{E}[X^{(1)\,T} X^{(2)}]$. These population covariances assume an underlying probability distribution from which the data are drawn.

Sample Covariance: In practical settings, we often do not have access to the entire population but only to a sample. Hence, we can use the Sample Average Approximation to estimate these covariances:

$$\hat{\Sigma}^{(12)} = \frac{1}{b-1}\mathbf{X}^{\bar{(1)}}\mathbf{X}^{\bar{(2)}}\top$$

Here, b denotes the size of the minibatch, and $\mathbf{X}^{(1)} \in \mathbb{R}^{D_1 \times b}$ and $\mathbf{X}^{(2)} \in \mathbb{R}^{D_2 \times b}$ are the data matrices for the samples from $X^{(1)}$ and $X^{(2)}$, respectively. The bar over $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ signifies that these are centered versions of the matrices, i.e., the mean has been subtracted from each column. For the ease of both reader and writer, we will drop the bars for the remainder of the thesis and assume that all data matrices are centered without loss of generality.

Practical Implications: Using sample covariance matrices introduces some estimation error but allows us to apply the methods in real-world scenarios where population-level data are unattainable. Additionally, the use of minibatches (chunks of data) in later chapters provides a computationally efficient way to estimate these covariances in large-scale problems, at the cost of some additional statistical noise.

Connection to Previous Methods: The use of sample covariance matrices is directly applicable to algorithms like CCA and LDA. When replacing the population covariances $\Sigma^{(ij)}$ with sample estimates, the optimization problems remain structurally similar but are solved using the sample data.

This dual perspective—considering both population and sample covariance matrices—enables a more robust and flexible approach to the methods discussed, bridging the gap between theoretical analysis and practical application. It will be particularly useful in the context of chapter IV where we will use population variables as ground truth while estimating the models using sample data.

5 Practical Frameworks for Evaluating Multiview Learning Methods

At this point, we have introduced the theoretical foundations of multiview learning, and a number of classical representation learning algorithms including CCA and its variants. However, it is not yet clear how we should evaluate these methods in practice. In this section, we compare the machine learning and the statistical

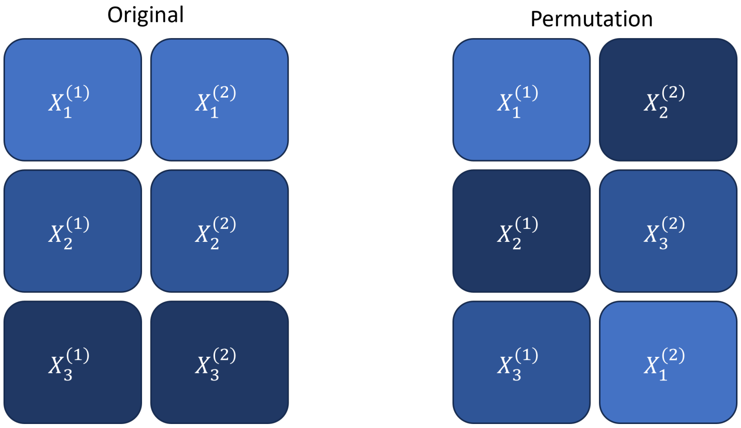


Figure II.4: Schematic of the permutation testing procedure. The original data are randomly shuffled, and the model is retrained on the shuffled data. This process is repeated multiple times, and the model's performance on the original data is compared to the distribution of performances on the shuffled data.

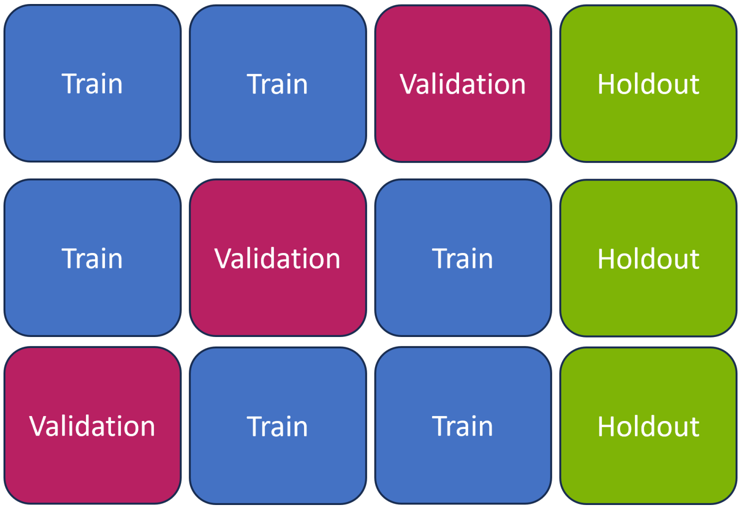
approach of permutation testing. These two approaches are not mutually exclusive, and statistical learning theory has emerged as a unifying framework for both perspectives (Vapnik, 1999; Hastie et al., 2009).

5.1 Permutation Testing

Permutation testing offers a robust way to evaluate the significance of the results obtained by multiview learning methods and, for a single component, is a relatively simple process (Winkler et al., 2020). As illustrated in Figure II.4, the views are randomly and separately shuffled, and the model is then trained and tested on this permuted data. This process is repeated multiple times, generating a distribution of performance metrics under the null hypothesis, where there is no relationship between the views. The actual performance of the model on the unshuffled data is then compared to this distribution. If the actual performance is significantly better than the permuted performance, it suggests that the model is capturing meaningful relationships in the data.

5.2 Machine Learning

The machine learning approach to evaluating multiview learning methods is to use a holdout or test set to estimate the out-of-sample performance of the model.



Train	Train	Validation	Holdout
Train	Validation	Train	Holdout
Validation	Train	Train	Holdout

Figure II.5: Schematic of the cross-validation procedure. The original data are partitioned into training and test sets. In cross-validation, the training set is further partitioned into training and validation sets. The model is trained on the training set and evaluated on the validation set for different parameter values. The parameter value with the best performance on the validation set is selected, and the model is retrained on the entire training set. The final model is evaluated on the single test or holdout set.

Within the training set, where necessary, cross-validation is used to select the best model hyperparameters. Cross-validation involves partitioning the training set into training and validation sets, training the model on the training set, and evaluating the model on the validation set. When this is performed for multiple subsets of the training set, it is referred to as k-fold cross-validation as illustrated in figure II.5. The model hyperparameters are then selected based on the performance across the validation sets. The model is then retrained on the entire training set using the best hyperparameters, and evaluated on the test set.

In this thesis, we will use the machine learning approach throughout. This is because in scaling up to large datasets, permutation testing becomes computationally intractable. This is because permutation testing requires retraining the model multiple times on the permuted data. This comes at the cost of only being able to evaluate models with a point estimate of performance, rather than a distribution.

5.3 Components and Subspaces in CCA

5.3.1 Eigenvalue Problems in CCA

While our focus so far has primarily been on the top-1 eigenvector-eigenvalue pair, it's important to note that the methodology also extends to the top-k subspace problem. This broader approach involves identifying the top-k eigenvectors and their corresponding eigenvalues.

5.3.2 Addressing the Top-k Problem

Transitioning from a focus on the top-1 component to exploring the top-k subspace introduces additional complexities. One common method to solve the top-k problem is to identify the top-1 component and then apply a deflation process to find subsequent orthogonal components. Deflation involves removing the top-1 component from the data and then repeating the process to find the next top-1 component. This process is repeated until the desired number of components is found. For instance, Hotelling's Deflation (Hotelling, 1933) involves removing the top-1 component from the data, while Projection Deflation (Mackey, 2008) involves projecting the data onto the orthogonal complement of the top-1 component. Different deflation methods enforce different forms of orthogonality, which can impact the resulting components and their interpretation, particularly when the first component is not a true eigenvector.

5.3.3 Non-Uniqueness of Components

Furthermore, non-uniqueness is a significant challenge in representation learning, particularly when eigenvectors have repeated eigenvalues. Imagine a scenario where the top-1 eigenvalue is repeated k times. In this case, there are k possible eigenvectors that can be associated with the top-1 eigenvalue. While this is unlikely to occur in practice, the eigenvalues can in practice be very close to each other, leading to numerical instability and non-uniqueness in the components. Particularly true in cross-validation settings, this non-uniqueness can lead to instability in the components, complicating their interpretation and comparison. For example, the top-1 component in one analysis might be the second component in another analysis, making it difficult to compare the results.

This non-uniqueness also has a grounding in the probabilistic perspectives on PCA and CCA (introduced in chapter IV), where the latent variables are considered unique only up to a rotation. This perspective further reinforces the subspace approach, emphasizing the identification of a subspace rather than specific directions within it.

Thesis Approach: Concentrating on the Top-1 Component In this thesis, we focus on the top-1 component in CCA to align with and facilitate comparison with typical componentwise studies in brain-behavior research. This choice is driven by the complexity associated with the top-k problem and the variety of methods available to address it. Under the assumption of a significant eigengap[2], the first component can be considered equivalent to the top-1 subspace. This equivalence allows for a clear and interpretable analysis, making the top-1 subspace a straightforward and reliable choice for studying multivariate data. It is important to note that while we focus on the top-1 component, the later sections of the thesis introduce a method for simultaneously solving the complete subspace, addressing broader subspace analyses.

6 Multiview Learning in Neuroimaging

Finally, we review important applications of multiview learning from the literature in neuroimaging, which will be our reference in chapters III and IV.

[2] An ‘eigengap’ refers to the difference in magnitude between consecutive eigenvalues in an eigenvalue problem. A significant eigengap between the first and second eigenvalues suggests that the first eigenvalue (and its corresponding eigenvector) is distinctly more significant than the next, lending credence to its uniqueness and importance.

6.1 Multiview Data in Neuroscience and Genetics

In neuroscience and genetics, two specific types of multiview studies are particularly relevant to this thesis: brain-behavior studies and imaging-genetics. Both involve the integration of data from multiple sources, offering rich insights into complex phenomena.

Brain-behavior studies typically involve pairing neuroimaging data, such as that obtained from Structural MRI (sMRI) or Functional MRI (fMRI), with non-imaging data like responses from questionnaires, cognitive test results, and other behavioral assessments. sMRI provides detailed anatomical brain images, essential for understanding brain structure and neurological disorders (Kanai and Rees, 2011), while fMRI focuses on brain function by mapping activity during cognitive tasks (Miranda et al., 2021).

In addition to task-based fMRI, resting-state fMRI (rs-fMRI) is frequently used to extract functional connectivity, reflecting the brain's functional organization during rest. This approach has been widely applied in Canonical Correlation Analysis and Partial Least Squares studies. For instance, S. M. Smith, Nichols, et al. (2015) applied CCA to rs-fMRI data alongside non-imaging features, demonstrating the utility of rs-fMRI in understanding intrinsic brain connectivity.

The integration of these imaging techniques with behavioral data offers a comprehensive view of how brain structures and functions correlate with behavioral and cognitive patterns (Rypma and D'Esposito, 2001; Genon, Eickhoff, and Kharabian, 2022).

Imaging-Genetics, another critical multiview approach, combines neuroimaging data with genetics and omics information (Lê Cao et al., 2008). This interdisciplinary field seeks to understand the genetic influences on brain structure and function, thereby illuminating the genetic basis of neuropsychiatric disorders and cognitive traits (R. Bogdan et al., 2017). Studies in this area can explore how specific genetic variations correlate with differences in brain morphology or activity patterns observed in neuroimaging (J. Liu and Calhoun, 2014).

Together, these multiview approaches are fundamental in advancing our understanding of the brain's structure, function, and its interactions with genetic and behavioral factors. They represent key applications of SSL in neuroscience and genetics, providing comprehensive insights that underpin developments in these fields.

6.2 Applications of Multiview Learning in Neuroimaging

There have been a number of applications of CCA and related methods to multiview problems in neuroimaging. Using resting state fMRI data, modes of correlation have been found that relate to differences in sex and age relating to drug and alcohol abuse, depression and self harm (Mihalik, Ferreira, Rosa, et al., 2019). A similar mode relating to 'positive-negative' wellbeing has been found across studies (S. M. Smith, Nichols, et al., 2015) suggesting that mental wellbeing has a relationship (though not necessarily causally) with functional connectivity between networks in the brain. Later in this dissertation we will replicate and build on the findings from this paper by using regularised and non-linear CCA methods. Owing to the high dimensionality of neuroimaging data, regularisation has been a particular focus of multiview learning in neuroimaging. Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al. (2022) reviews the application of CCA to neuroimaging data and highlights the importance of regularisation in this context. Bilenko and Gallant (2016) CCA has also been used as a preprocessing step in order to identify groups of subjects in the latent variable space.

In particular, CCA and clustering have been used to identify depression using fMRI data (Dinga et al., 2019; Drysdale et al., 2017). CCA has also been used in the manner we described to denoise two views of a dataset such as separate measures of neuroimaging data (Zhuang, Yang, and Cordes, 2020) to remove artefacts. Deep CCA has recently been used to extract features for the diagnosis of schizophrenia(Qi and Tejedor, 2016).

7 Conclusion

In this chapter, we have provided a comprehensive overview of multiview learning, with a particular focus on its applications in neuroimaging and genetics. We have discussed the fundamental concepts and methods in multiview learning, such as Canonical Correlation Analysis, Partial Least Squares, and their variants, highlighting their strengths and limitations.

The review has emphasized the importance of regularization techniques in high-dimensional settings, as well as the challenges associated with interpreting the resulting components. We have also touched upon the need for efficient algorithms that can handle large-scale datasets, and the potential of non-linear extensions of CCA and joint embedding self-supervised learning approaches.

Furthermore, we have discussed the practical frameworks for evaluating multiview learning methods, comparing the traditional statistical approach of permutation testing with the machine learning approach of cross-validation and holdout testing. We have also considered the complexities of identifying and interpreting top-k subspaces in CCA, and the reasons for focusing on the top-1 component in this thesis.

The applications of multiview learning in neuroimaging and genetics have been highlighted, with a particular emphasis on brain-behavior studies and imaging-genetics. These studies have demonstrated the potential of multiview learning in uncovering the complex relationships between brain structure, function, genetics, and behavior, thereby advancing our understanding of neurological disorders and cognitive traits.

Despite the significant progress made in multiview learning, several challenges remain. These include the need for more interpretable and regularized methods, particularly in high-dimensional settings, the development of efficient algorithms for handling large-scale datasets, and the extension of CCA to non-linear and deep learning-based approaches.

In the following chapters, this thesis aims to address these challenges by proposing novel methods and techniques for multiview learning. We will explore regularized and interpretable extensions of CCA, develop efficient algorithms for high-dimensional data, and investigate the potential of deep learning-based approaches for multiview learning. By tackling these challenges, we hope to contribute to the advancement of multiview learning and its applications in neuroimaging, genetics, and beyond.

Chapter III

Regularisation of CCA Models: A Flexible Framework based on Alternating Least Squares

Contents

Preface

In this chapter, I build upon work presented at the OHBM conference (James Chapman, 2023) and the insights gained from a tutorial paper I co-authored, which included a series of simulations (Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al., 2022).

1 Introduction

This chapter introduces a novel approach for analyzing large-scale neuroimaging datasets, such as the Human Connectome Project (HCP (Van Essen et al., 2013)) and Alzheimer's Disease Neuroimaging Initiative (ADNI), to understand the relationship between brain structure, function, and behavior (S. M. Smith and Nichols, 2018; Bzdok and Yeo, 2017; H.-T. Wang et al., 2020). These datasets are characterized by a disproportion between the number of subjects and the volume of features, posing a challenge for Canonical Correlation Analysis (CCA) models due to the risk of overfitting and spurious correlations (H.-T. Wang et al., 2018). For example, the HCP dataset used in this chapter contains 1003 subjects and 300 features derived from resting-state functional MRI (rs-fMRI), while the ADNI dataset contains 592 subjects and 168,130 features in the structural MRI (sMRI) view alone.

In response to the reproducibility crisis in neuroscience (Button et al., 2013), this chapter focuses on enhancing the generalizability of CCA models through regularization, a technique that introduces a bias towards more interpretable and generalizable models (Engl, Hanke, and Neubauer, 1996; Bzdok, Nichols, and S. M. Smith, 2019). Existing regularization methods in CCA, such as 'sparse CCA' with Partial Least Squares (PLS) objectives (Lê Cao et al., 2008; Witten, Tibshirani, and Hastie, 2009; Lindenbaum et al., 2021), are limited by their inherent bias towards the largest principal components (Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, and Mourão-Miranda, 2022).

Sparse CCA methods, particularly the one proposed by Witten, Tibshirani, and Hastie (2009), often referred to as sCCA, are commonly applied in neuroimaging studies. However, these methods face significant challenges when applied to very high-dimensional data, such as structural MRI (sMRI) data. The primary constraint of sCCA is the assumption of identity covariance, which may not hold in practice and limits its applicability to complex neuroimaging data.

To overcome these limitations, we propose the Flexible Regularised Alternating Least Squares (FRALS) framework for CCA based on the Alternating Least Squares form of CCA (Golub and Zha, 1995). FRALS allows for the integration of various regularized least squares solvers, particularly emphasizing the elastic net penalty, which combines L2 and L1 penalties. This method controls bias and promotes sparsity in model weights, advancing beyond previous sparse Brain-Behavior analysis methods.

Our application of the FRALS framework with Elastic Net regularization to the HCP and ADNI datasets showcases its effectiveness in enhancing out-of-sample canonical correlation compared to traditional CCA models. Additionally, FRALS uncovers new modes of variation in brain-behavior relationships.

In essence, this chapter presents FRALS as a robust, innovative solution for the analysis of high-dimensional neuroimaging datasets, significantly improving the reliability and interpretability of Brain-Behavior correlations.

2 Background: Regularisation for High-Dimensional and Structured Data

In this section, we review a number of regularisation techniques that have been applied to CCA and related methods.

2.1 The Bias-Variance Tradeoff

A key principle in machine learning is the bias-variance tradeoff (Curth, Jeffares, and Schaar, 2023; Hastie et al., 2009). This concept posits that a tradeoff exists between the bias and variance of a model: high-bias models typically exhibit low variance, and vice versa. High-bias models are generally simpler and more stable, but they might oversimplify the problem, leading to underfitting. Conversely, low-bias, complex models are sensitive to data changes and prone to overfitting. As the number of features increases, there are more parameters to estimate, and models tend to become more complex, leading to higher variance and lower bias. This relationship highlights the importance of balancing model complexity to avoid overfitting, particularly in high-dimensional scenarios with a low signal-to-noise ratio (McIntosh, 2021)[1]. Regularisation can be understood as a method for reducing the variance of

[1] It's worth noting that the number of model parameters, often used as a proxy for complexity, does not always directly correlate with model behavior, as illustrated by the 'double descent' phenomenon.

a model by introducing a bias towards simpler models. This means regularisation can improve the generalizability of models in high-dimensional settings.

2.1.1 Implicit and Explicit Regularisation

We can implement regularisation in two different ways. Explicit regularisation is achieved by adding a penalty term to the objective function. This weights the objective function against a term that penalises complexity.

Implicit regularisation is achieved by changing the optimisation algorithm and can include dimensionality reduction, as well as certain optimisation procedures like using stochastic gradient descent in place of gradient descent (Ali, Dobriban, and Tibshirani, 2020), and early stopping of optimization routines (Yao, Rosasco, and Caponnetto, 2007)

2.2 Shrinkage Regularisation

Shrinkage regularisation is a form of regularisation that penalises the magnitude of the model parameters. This technique is particularly effective in enhancing the performance of linear models in situations characterised by high dimensionality, multicollinearity, or low signal-to-noise ratios.

In high-dimensional situations where the number of features exceeds the number of observations in either view, like Linear Regression, Canonical Correlation Analysis is non-identifiable, meaning there is no unique solution. This is because we can find perfectly correlated latent variables using a linear combination of the features, but there are many different linear combinations that will achieve this. While in principle cross-validation could be used to search among these solutions, the infinite space of perfectly correlated combinations makes this impractical without additional constraints on the solution space[2].

Even in low-dimensional situations, if features exhibit multicollinearity, they can also be non-identifiable or, at best, estimates of the parameters are unstable. Mathematically, this is because in both cases the covariance matrix of the features is not full rank and therefore is not invertible (non-identifiable) or ill-conditioned (matrix inversion is unstable). To capture this intuition, if two features are perfectly correlated, the model is not identifiable (has no unique solution) because we can arbitrarily swap the weights between the two features without changing the latent variables (CCA) or the predictions (regression). In practice, features are rarely perfectly correlated,

[2] Regularisation provides such constraints, effectively making the problem identifiable by restricting the space of possible solutions.

but even when features are highly correlated, the model can be unstable (Mihalik, Ferreira, Moutoussis, et al., 2020), and small changes in the data can lead to large changes in the model parameters. Once again, some of these linear combinations will generalize better than others, but we might expect a model to generalize better if it spreads the weights across the correlated features rather than concentrating them on a single feature.

Finally, even in low-dimensional settings with little multicollinearity, the model parameters can be sensitive to noise in the data, and once again small changes in the data can lead to large changes in the model parameters. For example, parameters associated with noisy features might 'cancel out' in the training set, but not in the test set, leading to poor generalisation.

The premise of shrinkage regularisation in all these cases is that the latent variables or predictions are too sensitive to small changes in the data because the model parameters are too large. Shrinkage regularisation works by shrinking the model parameters towards zero, so that small changes in the data do not lead to large changes in the model estimates.

2.2.1 PLS as Shrinkage Regularisation

PLS can be interpreted as a form of shrinkage regularisation applied to CCA. We can explain this by considering an analogy between CCA and Linear Regression[3].

In Linear Regression, the ridge regression solution is given by:

$$\hat{\beta}_{\text{ridge}} = ((1-c)\Sigma_{X,X} + cI)^{-1}\Sigma_{X,y} \tag{III.1}$$

Where c is the regularisation parameter between 0 and 1[4]. The ridge penalty acts in three important ways:

- It shrinks the weights towards zero.
- It shrinks the weights of correlated features towards each other.
- It biases the solution to high covariance directions rather than high correlation directions.

As c becomes large, $\lim_{c\to 1}(\Sigma_{X,X} + cI)^{-1} = (cI)^{-1}$, so that $\hat{\beta}_{\text{ridge}} = \frac{\Sigma_{X,y}}{c}$, which is precisely the covariance of the features of X with Y scaled by c (and

[3] indeed Linear Regression is a special case of CCA where $X^{(2)}$ has one feature

[4] It is more common to see $(\Sigma_{X,X} + cI)^{-1}\Sigma_{X,y}$ but these are equivalent up to a scalar factor and this form helps us later on

shrunk towards zero for $c \geq 1$). Notice that the ridge regression solution is no longer sensitive to the correlation of features in X. Additionally, notice that for sufficiently large c, $(\Sigma_{X,X} + cI)$ is invertible even if $\Sigma_{X,X}$ is not invertible, so that ridge regression is always identifiable even when the number of features exceeds the number of observations.

Now consider the CCA problem. Firstly, recall that PLS and CCA are equivalent up to a scaling when the covariance matrices are identity matrices, a similar relationship to the relationship between Linear and Ridge Regression. Consider the well-known form of CCA given in equation III.2(Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al., 2022) (formed by reparameterizing $u^{(i)} = (\Sigma_{ii})^{-\frac{1}{2}}u^{(i)}$):

$$\begin{aligned}
&u_{\text{opt}} = \underset{u}{\operatorname{argmax}}\{u^{(1)\,T}(\Sigma_{11} + cI)^{-\frac{1}{2}}\Sigma_{12}(\Sigma_{22} + cI)^{-\frac{1}{2}}u^{(2)}\} \qquad \text{(III.2)}\\
&\text{subject to:}\\
&u^{(1)\,T}u^{(1)} = 1, u^{(2)\,T}u^{(2)} = 1
\end{aligned}$$

As we increase c, $\lim_{c\to\infty}(\Sigma_{ii} + cI)^{-\frac{1}{2}} = (cI)^{-1}$ so that the objective approaches:

$$\begin{aligned}
&u_{\text{opt}} = \underset{u}{\operatorname{argmax}}\{u^{(1)\,T}(cI)^{-1}\Sigma_{12}(cI)^{-1}u^{(2)}\} \qquad \text{(III.3)}\\
&\text{subject to:}\\
&u^{(1)\,T}u^{(1)} = 1, u^{(2)\,T}u^{(1)} = 1
\end{aligned}$$

Which is precisely the PLS objective and constraints with an arbitrary scaling of the covariance matrix Σ_{12} by $\frac{1}{c^2}$. For this reason, we can consider PLS as an explicit shrinkage method for CCA, equivalent to adding a maximal ridge regularisation term. The downside of using PLS as a regularised CCA is precisely its very high bias. By strongly guiding the model towards high covariance solutions, it strongly biases the solution towards only the largest principal components. But what if the correlation between the views is not concentrated in the largest principal components? Although one would rarely resort to maximally regularised ridge regression except in extremely low sample sizes or high-dimensional data, it has become almost standard practice to use PLS in neuroimaging and genetics (Cruciani et al., 2022; Krishnan et al., 2011). One of the core contributions of this chapter will be to demonstrate that PLS is usually a poor choice for regularisation even in these very high-dimensional settings and that more nuanced regularisation methods can offer significant improvements in performance and interpretability. PLS is evidently not a nuanced tool for regularisation because it offers no control over the degree of regularisation applied.

2.2.2 Ridge Regularisation

For this reason, Vinod (1976) proposed the Canonical Ridge or Ridge CCA, which combined the PLS and CCA constraints in a single constrained optimisation:

$$
\begin{aligned}
&u^{(1)}_{\text{opt}} = \underset{u^{(1)}}{\operatorname{argmax}}\{u^{(1)\,T}\hat{\Sigma_{12}}u^{(2)}\} \\
&\text{subject to:} \\
&(1-c_1)u^{(1)\,T}\hat{\Sigma_{11}}u^{(1)} + c_1 u^{(1)\,T}u^{(1)} = 1 \\
&(1-c_2)u^{(2)\,T}\hat{\Sigma_{22}}u^{(2)} + c_2 u^{(2)\,T}u^{(2)} = 1
\end{aligned}
\tag{III.4}
$$

Where c_1 and c_2 are the ridge regularisation parameters for the first and second views respectively. By tuning these parameters, we can control the degree of regularisation applied to each view independently. If we set c_1 and c_2 to zero, we recover the standard CCA objective while if we set c_1 and c_2 to one, we recover the PLS objective. This allows us to interpolate between the two extremes, allowing us to control the level of shrinkage and therefore the level of bias towards the largest principal components. Ridge CCA has been shown to be effective for neuroimaging data for both CCA (A. Tenenhaus and M. Tenenhaus, 2011; Tuzhilina, Tozzi, and Hastie, 2023; Hardoon, Szedmak, and Shawe-Taylor, 2004) and Kernel CCA (Hardoon, Mourao-Miranda, et al., 2007).

2.2.3 Principal Component Regularisation for CCA

Principal Component Analysis (PCA) can be used as an implicit regularisation method for CCA. Most obviously, by using only the first k principal components of each view as the input to CCA, we can reduce the dimensionality of the data and therefore reduce the number of parameters in the model. Moreover, by working with the principal components, we remove the correlation between the features, which can improve the conditioning of the problem. While PCA and Independent Component Analysis (ICA) are often used as preprocessing steps for CCA, they can also be used as regularisation methods in their own right. Of particular note in neuroimaging are studies with a data-driven approach to the PCA step, where the number of principal components is chosen based on the data (Z. Liu et al., 2022; Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, and Mourão-Miranda, 2022).

2.2.4 The Impact of Eigenvalue Spectra on Linear Model Solutions

The solutions of linear models, such as linear regression and Canonical Correlation Analysis, are sensitive to the eigenvalue spectrum of the input data's covariance matrix. This sensitivity arises from the matrix inversion step in the solution process, where the inverted covariance matrix directly influences the model's weights.

In the context of linear models, the eigenvalues of the covariance matrix represent the variances of the input features in the directions of the corresponding eigenvectors. Large eigenvalues indicate high variances and suggest the presence of strong patterns or associations, while small eigenvalues indicate low variances and suggest the presence of noise or weak patterns.

When the covariance matrix is inverted, the reciprocals of the eigenvalues are used. This inversion process amplifies the influence of small eigenvalues and attenuates the influence of large eigenvalues on the model's solution. Consequently, the model's weights become highly sensitive to noise and weak patterns associated with small eigenvalues, potentially leading to overfitting and unstable solutions.

Regularization techniques, such as Ridge regularization and Principal Component regularization (PCR), aim to mitigate this sensitivity by altering the eigenvalue spectrum of the covariance matrix. Figure III.1 illustrates these effects by plotting the eigenvalues of covariance matrices as perceived by models with different regularization techniques[5].

Ridge regularization compresses the spectrum, reducing the influence of the largest eigenvalues and increasing the influence of the smallest eigenvalues. PCA truncates the spectrum, preserving the influence of the largest eigenvalues while eliminating the influence of the smallest eigenvalues.

By modifying the eigenvalue spectrum, regularization techniques control the sensitivity of linear model solutions to different patterns in the data. However, while these techniques can improve model performance and stability, they do not inherently enhance the interpretability of the results, as the weights are shrunk

[5] For Ridge, we plot the eigenvalues of $(1 - c_i)\hat{\Sigma_{ii}} + c_i I$, while for PCA, we plot the eigenvalues of $\hat{\Sigma_{ii}}$ truncated to include only the largest k principal components.

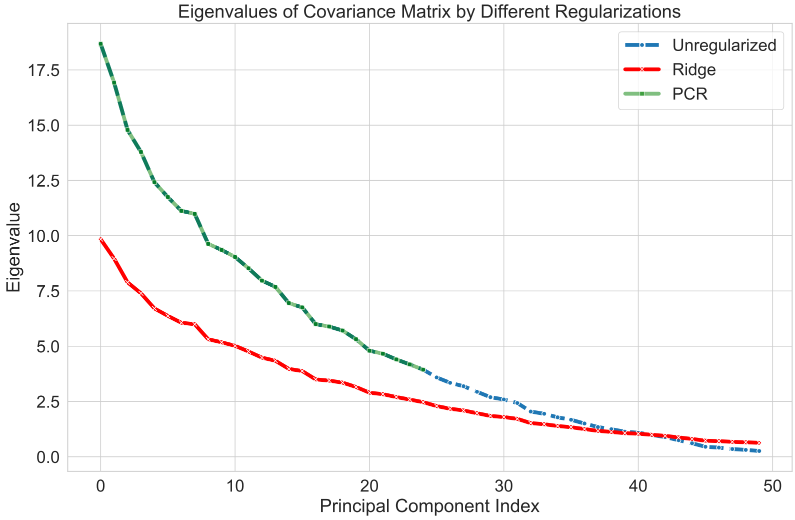


Figure III.1: Comparison of the effect of Ridge and Principal Components regularization on the eigenvalues of the covariance matrix.

towards zero but not eliminated, and the models still utilize all features.

- **Unregularized**: Utilizes the unaltered spectrum, retaining potential subtle patterns but susceptible to noise.
- **Ridge**: Warps the spectrum, attenuating the influence of the largest eigenvalues and amplifying the influence of the smallest eigenvalues, potentially emphasizing stronger associations at the expense of subtle patterns.
- **PCA**: Truncates the spectrum, focusing solely on the largest eigenvalues and discarding the smallest eigenvalues, potentially emphasizing stronger associations at the expense of subtle patterns.

While these regularization techniques can improve CCA performance, they do not inherently enhance the interpretability of the results. Weights are shrunk towards zero but not eliminated, meaning the model still utilizes all features, and the results remain dense.

2.3 Sparse Regularisation

Sparse regularisation is a powerful tool for improving the performance and interpretability of linear models. Sparse regularisation encourages the model to use only a subset of the features, which can both help to avoid overfitting and improve the interpretability of the model. Sparse regularisation works on the premise that only a subset of the features are relevant to the model. Sparsity is typically achieved by adding either an L1 penalty or constraint[6]. The L1 penalty is defined as:

$$\|u\|_1 = \sum_i |u_i| \tag{III.5}$$

Intuitively, this is the sum of the absolute values of the elements of the vector. Now, with a foundational understanding of sparse regularisation, we review a number of approaches to adding sparsity to the CCA problem.

2.3.1 Sparse PLS: Penalised Matrix Decomposition

Penalised Matrix Decomposition (PMD) (Witten, Tibshirani, and Hastie, 2009) provides an approximate solution to the sparse CCA problem by altering the constraints of the classical CCA formulation. Specifically, PMD replaces the constraints $u^{(i)\,T}\hat{\Sigma_{ii}}u^{(i)} = 1$ with the PLS constraints $u^{(i)\,T}u^{(i)} = 1$ and additionally imposes $\|u^{(i)\,T}\|_1 \leq \tau$. The optimisation problem for PMD is then given by:

$$\begin{aligned} &u^{opt} = \underset{u}{\mathrm{argmax}}\{u^{(1)\,T}\hat{\Sigma_{12}}u^{(2)}\} \\ &\text{subject to:} \\ &u^{(1)\,T}u^{(1)} = 1, u^{(2)\,T}u^{(2)} = 1 \\ &\|u^{(1)}\|_1 \leq \tau_1, \|u^{(2)}\|_1 \leq \tau_2 \end{aligned} \tag{III.6}$$

Computational Efficiency and Limitations PMD has become highly influential largely due to its computational efficiency - it can be solved using a simple variant of the power method that alternates between multiplying $u^{(1)}$ by $\hat{\Sigma_{12}}$ and applying soft-thresholding. However, this efficiency comes at a significant theoretical cost. The method is only equivalent to sparse CCA when the covariance matrices are identity matrices, an assumption that rarely holds in practice, particularly for neuroimaging data where features are often highly correlated. Under non-identity covariance, PMD effectively becomes a Sparse PLS method rather than a true sparse CCA.

[6] The L0 norm of the weight vector is the number of non-zero elements in the vector and is arguably a closer match to the goal, but the L0 norm is (a) not a proper norm in the mathematical sense and (b) not convex and so is difficult to optimize.

This limitation is not unique to PMD - several other methods employ similar PLS approximations (Parkhomenko, Tritchler, and Beyene, 2009; Waaijenborg, Witt Hamer, and Zwinderman, 2008; Lindenbaum et al., 2021). While computationally convenient, this approximation has important consequences:

- **Bias Towards Principal Components**: These methods inherit PLS's tendency to emphasize high-variance directions, potentially missing important correlations in lower-variance subspaces.
- **Mishandling of Correlation Structure**: By ignoring within-view correlations, the methods may identify spurious sparse patterns that don't reflect true relationships in the data.

Importantly, these issues stem from the PLS approximation rather than from L1 regularization itself. As we will show empirically in Section 5, methods that maintain proper CCA constraints while incorporating sparsity can avoid these limitations, albeit at increased computational cost.

To address these problems and truly tackle the sparse CCA optimisation, another class of approaches have adopted a penalised least squares approach.

2.3.2 Sparse CCA: Least Squares Approaches

It is well known that the CCA problem can be formulated as a constrained least squares problem with the intuition that for $X^{(1)}u^{(1)} = 1$ and $X^{(2)}u^{(2)} = 1$, correlation is maximised when the squared distance between $X^{(1)}u^{(1)}$ and $X^{(2)}u^{(2)}$ is minimised. (Golub and Zha, 1995) proved the convergence of a simple algorithm which alternates between solving the least squares problem for $u^{(1)}$ and $u^{(2)}$ while keeping the other fixed.

With this intuition, Wilms and Croux, 2015 and Mai and Zhang, 2019 separately proposed iterative penalised least squares methods for sparse CCA.

$$u^{opt} = \underset{u}{\operatorname{argmin}} \left\{ \|X^{(1)}u^{(1)} - X^{(2)}u^{(2)}\|_2^2 + P(u) \right\} \tag{III.7}$$

subject to:

$$u^{(1)\,T}\hat{\Sigma_{11}}u^{(1)} = 1$$

$$u^{(2)\,T}\hat{\Sigma_{22}}u^{(2)} = 1$$

Where $P(u)$ is a penalty function. The penalty term can be any function that penalises the norm of the vector u. (Mai and Zhang, 2019) proved that solving the subproblems where one of $u^{(i)}$ is fixed is easy for one-homogenous P where $P((\mu+1)\theta) = (\mu+1)P(\theta)$ which notably includes the lasso penalty. This means a sparse CCA based on alternating lasso regressions can be solved relatively efficiently using existing solvers. However, the one homogenous penalty in practice limits the flexibility of the method. For example, the elastic net penalty is not one-homogenous and therefore cannot be used with this method. Chi et al. (2013) and Mullins et al., 2021 added ridge penalties to the subproblems to improve the conditioning of the problem in a way that could be considered a form of elastic net regularisation but the subproblems no longer correctly optimize the global objective[7].

2.3.3 Sparse CCA: Proximal Gradient Descent and ADMM

Kanatsoulis et al. (2018) proposed solving equation III.7 for more general classes of P using the alternating direction method of multipliers (ADMM) (Boyd et al., 2011). Fu et al., 2017 propose a regularised CCA based on an alternative classical CCA formulation, sometimes called the MAXVAR formulation, which views the problem as a constrained least squares with an auxiliary representation T(Carroll, 1968; Kettenring, 1971).

$$\underset{U,T}{\operatorname{argmin}} \left\{ \sum_i \|X^{(i)}U^{(i)} - T\|_F^2 \right\} \tag{III.8}$$

$$\text{subject to: } T^\top T = I \tag{III.9}$$

$$\tag{III.10}$$

In this formulation, $U^{(i)}$ represents the weights for the i^{th} view, and T denotes the latent variable matrix. The premise is that when T closely mirrors $X^{(i)}U^{(i)}$ across all i, the scores correlate. Notably, this method is adaptable to multiple views. The authors employed proximal gradient descent for regularisation, specifically suited for penalties like the lasso. While these methods are flexible, they don't have the plug-and-play nature of the penalised least squares methods. Not just a matter of convenience, this means that these methods are not compatible with existing solvers for regularised least squares problems like for example total variation regularisation solvers in nilearn, which are often highly optimised for specific problems and

[7] when rescaling the penalised solutions back to unit variance

modalities.

2.3.4 Structured Regularisation

As highly structured data, linear models using both structural MRI and fMRI data have been shown to benefit from structured regularisation methods but notably these methods have not been applied to CCA. Total variation regularisation, which biases spatially neighboring weights to be similar, has been shown to improve the performance of PCA (De Pierrefeu et al., 2017) and regression (Michel et al., 2011; Dohmatob et al., 2014; Baldassarre, Mourao-Miranda, and Pontil, 2012). Similarly, Laplacian (or GraphNet) regularisation, which induces a similar spatial bias with additional smoothness, has been shown to improve the performance of CCA on functional MRI data (Grosenick et al., 2013; Cuingnet et al., 2012).

Having discussed the benefits of both shrinkage (e.g., PCA-CCA, Ridge CCA, PLS), sparsity (SPLS, Sparse CCA), and structure (Total Variation, Laplacian) in handling high-dimensional, noisy, and structured data, a natural progression is to integrate these advantages. Specifically, the challenge lies in creating a framework that allows for users to match the regularisation method to their data and research question, enhancing the interpretability and performance of Brain-Behaviour association models. This led us to propose the Flexible Regularised Alternating Least Squares (FRALS).

3 Methods: Flexible Regularised Alternating Least Squares (FRALS)

The primary goal of our Flexible Regularised Alternating Least Squares framework is to provide a versatile and user-friendly interface for Canonical Correlation Analysis that excels at detecting shared patterns between views, even when those patterns don't dominate the variance within each view. Unlike PLS-based methods which are biased toward high-variance components, FRALS maintains CCA's focus on correlation rather than covariance, making it particularly effective at identifying subtle but meaningful relationships between views.

FRALS achieves this enhanced sensitivity through two key mechanisms. First, by solving the CCA objective directly through alternating least squares rather than using PLS approximations, FRALS preserves CCA's scale-invariance property. This means it can detect correlated patterns regardless of their absolute scale in either view. Second, the framework allows independent tuning of regularization for each view, enabling it to effectively handle cases where important shared patterns may be masked by stronger but unrelated variations within individual views.

This approach marks a significant departure from traditional methodologies in CCA, which often focused on developing specific solvers tailored for particular types of data or computational constraints. By contrast, FRALS democratises access to advanced CCA techniques by maintaining compatibility with any scikit-learn compatible regularised least squares solver. Such flexibility is particularly advantageous in interdisciplinary fields like neuroimaging, where diverse datasets and varying levels of technical expertise are common.

In the FRALS framework, we consider the formulation for a single latent variable t with regularisation $\lambda_i P_i$ on the weights $u^{(i)}$:

$$\underset{u}{\operatorname{argmin}} \left\{ \sum_i \|X^{(i)}u^{(i)} - t\|_2^2 + \lambda_i P_i(u^{(i)}) \right\} \quad \text{subject to: } t^\top t = 1 \tag{III.11}$$

This problem decomposes into subproblems. The first subproblem for the auxiliary variable t:

$$\underset{t}{\operatorname{argmin}} \left\{ \sum_i \|X^{(i)}u^{(i)} - t\|_2^2 \right\} \quad \text{subject to: } t^\top t = 1 \tag{III.12}$$

is solved in closed form by averaging and normalizing: $t = \frac{\sum_i X^{(i)}u^{(i)}}{\|\sum_i X^{(i)}u^{(i)}\|_2}$. This makes t an estimate of the latent variables of a generative CCA model, capturing shared patterns regardless of their scale in the original views.

The subproblems for the weights $u^{(i)}$:

$$\underset{u^{(i)}}{\operatorname{argmin}} \left\{ \|X^{(i)}u^{(i)} - t\|_2^2 + P_i(u^{(i)}) \right\} \tag{III.13}$$

are regularised least squares problems solvable using any suitable regularised least squares solver[8].

[8] We could also in principle replace $X^{(i)}u^{(i)}$ with $f(X^{(i)})$ for any function f including kernels, neural networks, or random forests

In this chapter, we demonstrate the framework's power using the Elastic Net solver from `scikit-learn` (Pedregosa et al., 2011), where $P_i = \alpha_i \times \text{l1\_ratio} \|u^{(i)}\|_1 + \alpha_i \times (1 - \text{l1\_ratio}) \|u^{(i)}\|_2^2$. This combination of L1 and L2 penalties enables FRALS to identify sparse patterns while maintaining stability, with independent tuning for each view. The framework's solver flexibility extends beyond Elastic Net to include specialized solvers for structured data, such as spatial regularization for neuroimaging applications, though these are not explored in this chapter.

4 Experiment Design

This section outlines the methodologies used in our study to explore the Flexible Regularized Alternating Least Squares (FRALS) and associated techniques in Canonical Correlation Analysis. We focus on fitting a single latent dimension for the analyses.

4.1 Datasets

For this chapter, we chose the HCP and the ADNI datasets to facilitate comparison with two influential brain-behaviour studies (S. M. Smith, Nichols, et al., 2015; Monteiro et al., 2016) as well as the tutorial paper that this chapter is loosely related to (Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al., 2022). We are particularly interested in the performance of an Elastic Net FRALS on these datasets as Ridge CCA has been shown to outperform PLS (Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al., 2022), implying that shrinkage regularisation is beneficial, and Sparse PLS has been shown to outperform PLS (Monteiro et al., 2016), implying that sparsity is beneficial. We therefore expect that Elastic Net FRALS will outperform PLS, Ridge CCA, and Sparse PLS on these datasets.

4.2 The Predictive Framework for CCA

Our evaluation of CCA models used a standard predictive framework, dividing the data into an 80:20 ratio for training and testing. This method ensures fitting the model on the training set without incorporating information from the test set.

4.2.1 Model Comparisons

The experiment aims to demonstrate the effectiveness of tunable shrinkage and sparsity in CCA models, enabled by the FRALS framework. We compare the performance of Elastic Net FRALS with other CCA variants such as PLS, Ridge CCA, and SPLS particularly in the context of high-dimensional datasets like HCP and ADNI. Our choice of comparisons reflects our focus on sparsity and elastic-net style regularization: Ridge CCA (L2 regularization only) and Sparse PLS (fixed L2, flexible L1 regularization) represent key points in this regularization spectrum.

We include PCA as our baseline, where separate Principal Component Analysis is applied independently to each view, and the first principal component from each view is compared. Despite PCA's known limitations in high-dimensional settings—particularly its bias towards directions of high variance rather than high correlation—it serves as an important baseline for three key reasons. First, as discussed earlier, PCA represents the simplest form of dimensionality reduction and implicit regularization. Second, comparing against PCA allows us to verify that multiview methods like CCA are actually discovering meaningful cross-view associations rather than simply identifying the dominant modes of variation within each view independently. Third, and crucially, comparing PLS and SPLS against PCA allows us to empirically test our theoretical prediction that these methods are biased towards the principal components. If PLS and SPLS perform similarly to PCA, this would provide strong evidence for our earlier theoretical argument that these methods may be too strongly biased towards high-variance directions, potentially missing important correlations in other directions.

We note that PCA-CCA, where PCA is used as a preprocessing step before CCA, was not included in our comparisons. While this is another common approach in the literature, we hypothesize it would show similar behavior to Ridge CCA since both methods bias solutions toward the dominant principal components in the data, as demonstrated in our earlier discussion of shrinkage regularization.

Table 4.1: Employed CCA Variants

Model	**Abbreviation**	**Hyperparameters**	**Hyperparameter Range**
Principal Component Analysis	PCA	-	-
Regularised CCA	RCCA	c_1, c_2	0-1 (log scaled)
FRALS - Elastic	Elastic	$\alpha_1, \alpha_2, \text{l1}_1, \text{l1}_2$	(1e-5,1e-1), (0-1)
Partial Least Squares	PLS	-	-
Sparse PLS	SPLS	τ_1, τ_2	0-1 (log scaled)

4.2.2 Model Selection

For models requiring hyperparameter tuning, we implemented a two-stage process. First, we performed hyperparameter optimization using grid search with 5-fold cross-validation on the training data only. This involved splitting the training data into five parts, where for each hyperparameter combination, we trained on four folds and validated on the fifth, rotating through all folds. The optimization goal was to maximize the average out-of-sample correlation across these validation folds.

After identifying the optimal hyperparameters using this cross-validation procedure, we trained the final model using these optimized parameters on the complete training dataset. The model's performance was then evaluated on the entirely held-out test data, which was not used during the hyperparameter optimization process. This approach ensures an unbiased assessment of the model's generalization capability.

4.2.3 The Human Connectome Project (HCP)

The HCP offers publicly available resting-state functional MRI (rs-fMRI) and non-imaging measures like demographics, psychometrics, and other behavioral measures. Specifically, we sourced data from 1003 subjects out of the 1200-subject data release of the HCP. The rs-fMRI data provided brain connectivity matrices. These were derived from pairwise partial correlations between subject components obtained through group independent component analysis (ICA), using 25 components. This resulted in 300 brain variables, corresponding to the lower triangle of the connectivity matrix. In our analysis, 145 non-imaging subject measures were incorporated, similar to prior studies, with the exception of 13 measures that were unavailable in the 1200-subject data release. Furthermore, nine confounding variables, including the acquisition reconstruction software version, a summary statistic of head motion during rs-fMRI acquisition, weight, height, systolic and diastolic blood pressure, hemoglobin A1C level, and cube-root of total brain and intracranial volumes as estimated by FreeSurfer, were regressed out from both data types. More details can be found in S. M. Smith, Nichols, et al. (2015) and Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al. (2022). We summarize the characteristics of the HCP data in table 4.2.

Table 4.2: HCP Data Characteristics

Characteristic	**Value**
Number of samples (*n*)	1003
Number of features in View 1 (*p*)	300
Number of features in View 2 (*q*)	145

4.2.4 The Alzheimer's Disease Neuroimaging Initiative (ADNI)

Accessible at `adni.loni.usc.edu`, the ADNI database was initiated in 2003. Its primary aim is the examination of how well serial MRI, PET (Positron Emission Tomography), biological markers, along with clinical and neuropsychological assessments, track the progression of Mild Cognitive Impairment (MCI) and the early stages of Alzheimer's disease. In our study, we used data from a subset of 592 unique individuals, selected based on having both a T1-weighted MRI scan and complete Mini-Mental State Examination data available at baseline. This subset comprised 309 males (average age 74.68 ± 7.36 SEM) and 283 females (average age 72.18 ± 7.50 SEM), including 147 healthy controls, 335 individuals with Mild Cognitive Impairment (MCI), and 110 diagnosed with dementia.

T1 weighted structural MRI (sMRI) scans were the source of whole-brain voxel-based grey matter volumes. The sMRI data underwent preprocessing with SPM12 (Ashburner et al., 2014), which involved segmentation, normalisation using DARTEL, reslicing to a resolution of $2 \times 2 \times 2\,\text{mm}^3$, and spatial smoothing using a Gaussian kernel with 2 mm full width at half maximum (FWHM). During the segmentation step, SPM12's unified segmentation algorithm assigns gray matter probabilities to each voxel through an iterative process that combines tissue intensity distributions with deformable tissue probability maps. The algorithm models each tissue class (gray matter, white matter, CSF) as a mixture of Gaussians and uses spatial priors from the MNI template, updating both the tissue classifications and the deformation parameters until convergence. A grey matter voxel selection mask was then created by including voxels that had a probability of being grey matter greater than or equal to 10% across all participants. This mask, with a threshold of $\geq$10%, was then applied to all participants' scans, resulting in 168,130 brain variables.

The Mini-Mental State Examination (MMSE) is a widely recognised neurocognitive test comprising 30 questions across five cognitive domains (M. F. Folstein, S. E. Folstein, and McHugh, 1975): orientation (questions 1-10), registration (questions 11-13), attention and calculation (questions 14-18), recall (questions 19-21), and language (questions 22-30). An additional item was included in our study to

Table 4.3: ADNI Data Characteristics

Characteristic	**Value**
Number of samples (*n*)	592
Number of features in View 1 (*p*)	168130
Number of features in View 2 (*q*)	31

account for the number of attempts a subject needed to correctly respond to the registration domain questions, leading to a total of 31 variables. We retained all variables without removing potential confounds to facilitate direct comparison with previous analyses, particularly Monteiro et al. (2016), rather than based on specific hypotheses about variable relationships. We summarize the characteristics of the ADNI data in table 4.3.

5 Experiment Results

5.1 HCP Results

Next, we consider the results of applying the various CCA variants to the HCP data.

5.1.1 Out of Sample Correlation

Both Ridge CCA and Elastic Net outperformed PLS and SPLS in terms of holdout correlation captured (Figure III.2). This suggests that tunable L2 regularisation is important, even for very high-dimensional data, and that resorting to PLS is suboptimal. On the other hand, while the additional sparsity improved SPLS over PLS (consistent with previous work (Monteiro et al., 2016)), it did not improve the performance of the Elastic Net model over Ridge CCA. This suggests that the associations in the HCP data are non-sparse, and the sparse regularization provided by the Elastic Net did not offer any advantage over the Ridge CCA's L2 regularization.

Nonetheless, the Elastic Net model demonstrated a more sparse representation than the Ridge CCA model. Using scikit-learn's default tolerance of 1e-4 for identifying non-zero coefficients, we found the Elastic Net model retained 241 and 96 weights above this threshold for the brain and behavior views, respectively (Table 5.1). In contrast, the Ridge CCA model used all 300 and 145 weights in the respective views, with none falling below this threshold. The SPLS model achieved an even sparser solution with only 118 and 56 weights above the threshold for the

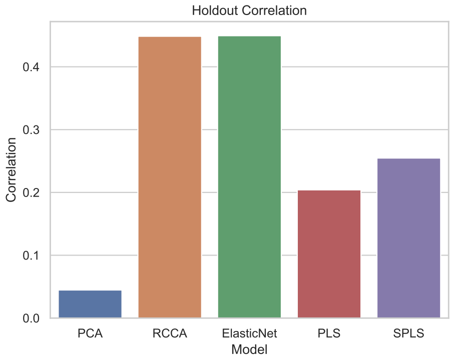


Figure III.2: HCP: Comparative out-of-sample canonical correlations among PCA, RCCA, ElasticNet, PLS, and SPLS models. The bars represent the correlation coefficients, indicating that Ridge CCA and Elastic Net models have superior performance over PLS and SPLS in capturing holdout correlation.

brain and behavior views. The choice between a sparse or dense representation depends on the specific aims of the analysis. If the goal is to uncover all potential associations, a dense representation involving all variables might be preferable. However, if the aim is to identify variables that may be driving the associations, a sparse representation like the one provided by the Elastic Net or SPLS models could be helpful by reducing the number of variables to consider. That said, with hundreds of non-zero weights still retained in the Elastic Net model, the practical interpretability advantage over Ridge CCA may be limited. Furthermore, given the comparable performance of the Elastic Net and Ridge CCA models, the sparsity achieved might be more a reflection of the regularization process than a meaningful identification of truly important variables.

Table 5.1: HCP: Sparsity of models reflected by the count of weights above scikit-learn's default tolerance threshold (1e-4). Elastic Net and SPLS demonstrate increased sparsity in the model weights for both brain and behaviour views, compared to PCA, RCCA, and PLS.

Model	Brain Weights (out of 300)	Behaviour Weights (out of 145)
PCA	300	145
RCCA	300	145
Elastic Net	241	96
PLS	300	145
SPLS	118	56

5.1.2 Behaviour Weights

Figure III.3 presents the top eight positive and negative non-imaging weights for each model to visualize the behavioural data variations observed in the previous section. The PCA model emphasizes a mode of variation with positive correlations to psychiatric and life function tests, contrasting with negative correlations to certain emotion and personality tests. In comparison, the RCCA and Elastic Net models highlight a variation mode negatively correlated with the Line Orientation test and to a lesser extent, smoking, while showing positive correlations with other cognitive assessments. The PLS model's variation mode echoes the positive-negative pattern identified by S. M. Smith, Nichols, et al., 2015, showing positive correlations with agreeableness, vocabulary tests, and life satisfaction, juxtaposed with strong negative correlations with smoking and antisocial behaviors. SPLS selects a similar mode but prioritizes vocabulary tests and smoking over rule-breaking and antisocial personality traits.

5.1.3 Brain Connectivity Weights

In this section, we use two different methods to visualize the brain connectivity weights. The first method is to use chord diagrams to visualize the top 8 positive and negative brain weights for each model. This approach is inspired by the chord diagrams used in S. M. Smith, Nichols, et al., 2015. The second method is to use surface maps to visualize the brain connectivity weights. This approach has been used by both Ferreira et al., 2022 and S. M. Smith, Nichols, et al., 2015.

Chord Diagrams We grouped the nodes of the connectivity matrix of our data into 7 parcels according to the Yeo 7 network parcellation Yeo et al., 2011. This was achieved by assigning each node to the network with the highest voxelwise overlap. These are then arranged around the circumference of the chord diagram using the Nichord package (P. C. Bogdan et al., 2023). The plots then show the 8 strongest positive and negative weights for each model as 'chords'. The chord diagrams in Figure III.4 show the top 8 positive and negative brain weights for each model. The text color for each network matches the color of the corresponding region on the outside of the chord diagram, providing a helpful visual guide.

- The **RCCA** model displays a diverse set of connections across all networks, with especially prominent weights in the somatomotor and default mode networks.

- The **ElasticNet** model presents similar connections between the somatomotor and default mode networks.
- The **PLS** model exhibits strong connections between the frontoparietal and visual networks.
- The **SPLS** model exhibits similar connections between the frontoparietal and visual networks.

The patterns we observe align with our theoretical understanding of these methods' behaviors. As predicted in Section 2.2, PLS and SPLS identify similar connectivity patterns, focusing on networks (frontoparietal and visual) that likely represent high-variance components in the data. This is consistent with our earlier discussion of PLS's bias towards principal components, suggesting these patterns might reflect the dominant modes of variation rather than necessarily the most meaningful brain-behavior relationships.

In contrast, RCCA and ElasticNet identify a broader set of connections involving the default mode and somatomotor networks. While one might speculate about relationships between these networks and the behavioral measures (for instance, the somatomotor network's potential role in the Line Orientation test), we should be cautious about such interpretations. The fact that these methods identify different patterns from PLS/SPLS could simply reflect their different biases rather than necessarily indicating more meaningful associations.

These visualization results underscore a key challenge in brain connectivity analysis: different methods can reveal quite different patterns of connectivity, and the "ground truth" of brain-behavior relationships remains unknown. While we can demonstrate superior out-of-sample prediction for some methods, this doesn't necessarily validate their biological interpretations. Indeed, connecting back to the theory we developed in the previous background chapter, it is important to recognize that these methods are fundamentally representation learning algorithms - their primary value lies in their ability to learn useful representations of the data for downstream tasks, rather than in their ability to reveal "true" underlying brain-behavior relationships. This perspective helps explain why RCCA and ElasticNet can achieve better predictive performance despite identifying different connectivity patterns from PLS/SPLS: they may be learning more useful representations of the data, even if these representations don't align with our intuitive interpretations of brain networks. Future work combining these analyses with other forms of validation, such as task-based fMRI or lesion studies, would be valuable for assessing which of these connectivity patterns truly reflect meaningful brain-behavior relationships, but their primary validation comes from their performance on downstream prediction tasks.

5.1.4 Model Similarity

In this section, we compare the models in terms of their similarity. We can measure the pairwise similarity between two models by comparing their weights and their representations[9]. We can compare the weights by computing the correlation between the weights of the two models, and we can compare the representations by computing the correlation between the representations of the two models.

In Figure III.5, we plot the correlation between the brain and behaviour representations for each model. We can see clearly that both PCA, PLS, and SPLS are all highly correlated in terms of their brain representations, revealing the bias of PLS towards the largest principal components. On the other hand, in the behaviour space, the models are less correlated, with the exception of PLS and SPLS which are highly correlated with one another. There is however still substantial correlation between the PCA and PLS models. The very low correlation between the Ridge CCA and Elastic Net models with the PCA model is evidence that there are stronger correlations outside of the first principal components.

In Figure III.6, we similarly plot the correlation between the brain and behaviour weights for each model. The story is similar, albeit with marginally lower correlations between the PLS and PCA-based models. Finally, in the weights space, the Ridge CCA and ElasticNet models are even less correlated with the PCA model.

5.2 ADNI Results

5.2.1 Out of Sample Correlation

In this experiment, the Elastic Net model outperformed all other models in terms of out-of-sample correlation (Figure III.7). The RCCA model also outperformed the PLS and SPLS models while SPLS outperformed PLS. Suprisingly, PCA performed almost as well as PLS. This suggests that there is value in both tunable shrinkage and sparsity in this dataset. It also reveals that the correlated signal between the brain structure and behavioural data is relatively much stronger than in the HCP data.

[9] Recall that in CCA models, what we call representations are sometimes referred to as scores or latent variables.

Table 5.2: ADNI: Number of non-zero weights for each model.

Model	Brain Weights (out of 168130)	Behaviour Weights (out of 31)
PCA	168130	31
RCCA	168130	31
Elastic Net	59617	17
PLS	168130	31
SPLS	74995	10

5.2.2 Sparsity of Weights

Table 5.2 once again shows the number of non-zero weights for each model. We can see that tuned SPLS and Elastic Net once again identify sparse weights. In this case, the difference in performance is more convincing and suggests that this sparsity is less spuriously induced than for the HCP data. This is supported by the fact that Elastic Net and SPLS models find a similar level of sparsity in the brain weights. On the other hand SPLS finds a much sparser set of behavioural weights.

5.2.3 Behaviour Weights

As for the HCP data, Figure III.8 plots the top 8 positive and negative non-imaging weights for each model. A striking feature of these results is that certain behavioral variables, particularly the orientation tests, receive high weights across all models, including PCA. Since PCA identifies directions of maximum variance independently in each view, the fact that it identifies similar behavioral variables to the multiview methods (CCA variants) suggests these variables are both high-variance within the behavioral data and strongly related to brain structure. This alignment between single-view (PCA) and multiview methods is notable because it differs from our HCP results, where PCA identified quite different patterns from the CCA variants.

SPLS and Elastic Net both emphasize the orientation and recall tests in the weight space. The RCCA and Elastic Net models show some interesting differences in the weight space, with RCCA assigning high weights to attention and calculation tests in addition to the orientation and recall tests that appear consistently across all models.

5.2.4 Brain Structure Weights

We plot the weights as a mosaic plot with 3 slices in each direction in Figure A.2. Previous work using SPLS with the ADNI dataset identified a striking pattern of weights with the model selecting the hippocampal regions (Monteiro et al., 2016).

The Elastic Net produces a distinct and interpretable pattern of weights, showing sparse but well-defined selection of regions including a structured pattern near the brain boundaries. Similarly, RCCA identifies specific regions of interest. Interestingly, while PCA, PLS and SPLS assign weights in the same direction, RCCA and Elastic Net differentiate regions with opposing signs, potentially capturing more nuanced relationships in the brain structure.

5.2.5 Model Similarity

In this section, we once again compare the models in terms of their similarity. In Figure III.10, we can see that all of the models are highly correlated in terms of their behaviour representations. The brain representations are less correlated, but once again PCA, PLS, and SPLS are highly correlated with one another and less correlated with the Ridge CCA and Elastic Net models.

Suprisingly, in Figure III.11, we can see that the weights in both views are less correlated. This is particularly true for the brain weights where PCA exhibits a very low correlation with Ridge CCA and Elastic Net.

6 Discussion and Limitations

The Flexible Regularised Alternating Least Squares (FRALS) framework for CCA, introduced in this chapter, exhibits promising performance in terms of out-of-sample correlation. Our findings indicate that while Elastic Net CCA generally outperforms other CCA variants, much of the benefit is derived from using properly tuned Ridge regularization.

To assess the stability and uniqueness of the identified associations, we examined the singular value spectrum of the PLS solution (Figure III.12). The substantial eigengap observed between the first and subsequent components in both datasets suggests that the primary brain-behavior mode is well-separated and likely represents a stable, identifiable pattern rather than an arbitrary rotation of a degenerate subspace. This provides evidence that the differences between methods are primarily related to their regularization approaches rather than different rotations of components with similar singular values.

The relative importance of Ridge regularization and sparsity, however, appears to be dataset-dependent. In the HCP dataset, our results suggest that the associations between brain and behavioral measures are not sparse, as sparsity did not improve the out-of-sample correlation. This questions the interpretability of the sparse model and casts doubt on whether the additional computational cost of Elastic Net CCA is justified for this particular dataset.

In contrast, for the ADNI dataset, the Elastic Net CCA model benefited from the sparsity regularization, indicating that the associations between brain and cognitive measures in this context are likely sparse. From a neuroscientific perspective, these results highlight that the effectiveness of Ridge regularization and sparsity can vary depending on the specific phenomenon under study.

Our experiments demonstrate that Ridge CCA typically achieves higher out-of-sample correlations compared to PLS across both datasets. While PLS optimizes for covariance, we focus on correlation-based metrics due to CCA's advantageous scale invariance properties. This scale invariance makes CCA particularly suitable for identifying meaningful associations between brain and behavioral measures, as it captures relationships that are independent of the arbitrary scaling of individual measurements.

While our analysis focuses primarily on the first component due to methodological constraints in comparing secondary components across different regularization approaches, the clear eigengap in the PLS solution provides confidence in the stability of our primary findings. Future work could explore methods for consistent deflation across different regularization approaches to enable robust comparison of secondary components.

In summary, while both Ridge regularization and sparsity can be helpful in CCA models, their relative trade-off and the associated computational cost should be evaluated in the context of the specific research question and the expected sparsity of the underlying associations. The stability analysis through singular value decomposition provides additional confidence that the identified brain-behavior relationships represent genuine, well-separated modes rather than arbitrary rotations of degenerate solutions.

6.1 FRALS Limitations

The Flexible Regularised Alternating Least Squares (FRALS) framework, while effective in certain aspects, is notably limited by its computational inefficiency. This inefficiency arises from two main factors: the dynamic nature of regression targets and the intensive computation required for each iteration.

6.1.1 Changing Regression Targets

In FRALS, regression targets are not static but dynamically evolve during the algorithm's execution. These targets are essentially projections of the other view, and as they change, they alter the optimization landscape. Consequently, the algorithm must frequently recompute the least squares solution for each view. This process results in significant computational overhead and often leads to redundant calculations, thereby contributing to the inefficiency of the FRALS framework.

6.1.2 Computational Time

The primary computational challenge in FRALS is the repeated calculation of the least squares solution for each view in every iteration. Figure III.13 shows the time taken (in seconds) to fit each model across complete training datasets over ten runs. While FRALS with Elastic Net regularization operates approximately 10 times slower than Ridge CCA, the absolute computation times remain manageable for the datasets examined here – typically on the order of minutes rather than hours.

The computational cost primarily scales with the dimensionality of the data, as FRALS using scikit-learn's elastic net regressors requires iterative solutions for each feature. While SPLS, PLS, and RCCA maintain relatively fast processing times due to their optimized C implementations and PCA preprocessing, the additional computational overhead of FRALS may be justified given the demonstrated benefits of proper Elastic Net regularization for certain datasets. However, this scalability limitation could become significant when applying the method to very high-dimensional data, such as voxel-level neuroimaging analyses in large-scale studies.

6.2 Beyond Single Components: Deflation and Subspace Extensions

A significant limitation of current sparse CCA methods, including SPLS, is their reliance on deflation procedures to find multiple components. Deflation approaches, while computationally convenient, can be problematic for several reasons:

- **Error Propagation**: When working with sparse or regularized solutions, the first component may not be a true eigenvector. Deflation based on such components can propagate and amplify errors through subsequent components (Mackey, 2008).

- **Loss of Structure**: Particularly for sparse methods, deflation can destroy the sparsity structure of the data, making interpretation of subsequent components challenging (Sameni, Jutten, and Shamsollahi, 2009).

- **Artificial Orthogonality**: Different deflation methods enforce different forms of orthogonality between components, which may not reflect the natural structure of the relationships in the data.

While the current implementation of FRALS focuses on single components, its formulation through alternating least squares suggests a natural extension to simultaneous multi-component optimization. Instead of sequential deflation, we could modify FRALS to optimize for a subspace directly by:

- Extending the auxiliary variable t to a matrix $T = [t_1, ..., t_k]$ containing multiple components
- Applying Gram-Schmidt orthogonalization to maintain orthogonality constraints on T
- Leveraging scikit-learn's multi-output regression capabilities for the weight updates

The proposed extension of FRALS to simultaneous multi-component optimization offers several compelling advantages. By eliminating the need for explicit deflation, this approach would address many of the fundamental issues that plague current sequential methods. The framework maintains complete flexibility in regularization choices, allowing researchers to select appropriate constraints for their specific applications. Importantly, this method preserves the natural underlying structure of the data, avoiding the artificial transformations that often result from deflation procedures. Perhaps most significantly, this approach could reveal important interactions between components that might otherwise be obscured or lost entirely when components are extracted sequentially. These advantages make the proposed extension particularly promising for advancing the field beyond the limitations of current deflation-based methods.

The mathematical formulation would become:

$$\underset{U,T}{\operatorname{argmin}} \left\{ \sum_i \|X^{(i)}U^{(i)} - T\|_F^2 + \lambda_i P_i(U^{(i)}) \right\} \tag{III.14}$$
$$\text{subject to: } T^\top T = I_k$$

where $U^{(i)}$ are now matrices of weights and $\|\cdot\|_F$ denotes the Frobenius norm.

This extension aligns better with the theoretical understanding of CCA as a subspace method, discussed in Chapter IV, where components are unique only up to rotation within their span. Future work could explore this direction, potentially leading to more robust and interpretable multi-component analyses.

The challenge of finding multiple components highlights a broader tension in the field between computational tractability and theoretical rigor. While deflation-based approaches like SPLS offer practical solutions, they may compromise the underlying mathematical structure of the problem. The proposed extension of FRALS suggests a path forward that maintains mathematical consistency while remaining computationally feasible through modern optimization tools.

6.3 Conclusion

In this chapter, we introduced the Flexible Regularised Alternating Least Squares (FRALS) framework for CCA. We used the FRALS framework to implement Elastic Net CCA. We then compared the performance of Elastic Net CCA with other CCA variants on two datasets: the HCP and ADNI. We found that Elastic Net CCA outperformed other CCA variants on both datasets but that the performance of Elastic Net CCA was similar to Ridge CCA on the HCP dataset. However, we found that Elastic Net CCA was much slower than other CCA variants.

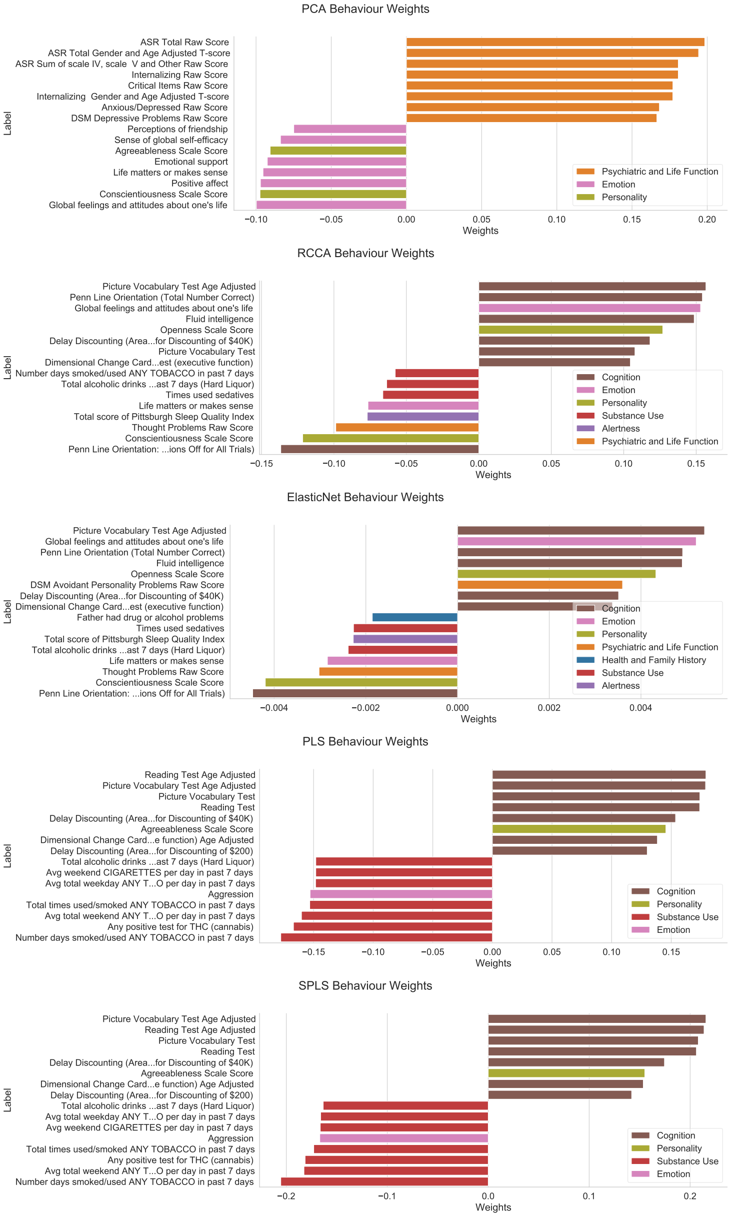


Figure III.3: HCP: Behavioural weights highlighting the top-8 positive and negative non-imaging weights. Each subfigure represents a distinct model's weight distribution across various behavioural domains such as cognition, emotion, personality, substance use, alertness, and psychiatric and life function. The variations in the weight profiles across models reflect differing patterns of association with the behavioural traits considered in the study.

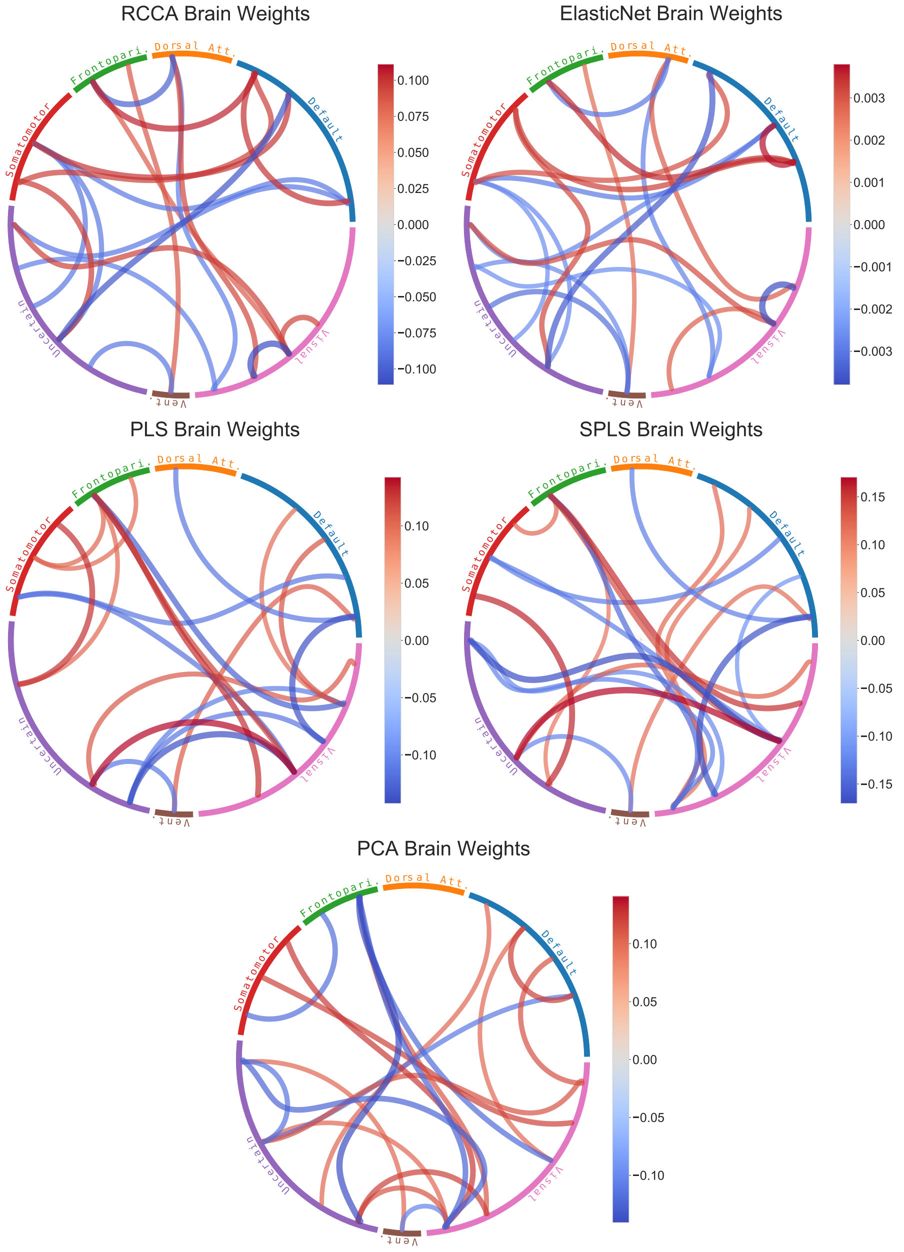


Figure III.4: HCP: Brain connectivity weights visualized through chord diagrams for multiple models. Each diagram portrays the 8 strongest positive (red to blue gradient) and negative (blue to red gradient) weights, grouped by the Yeo 7 network parcellation.

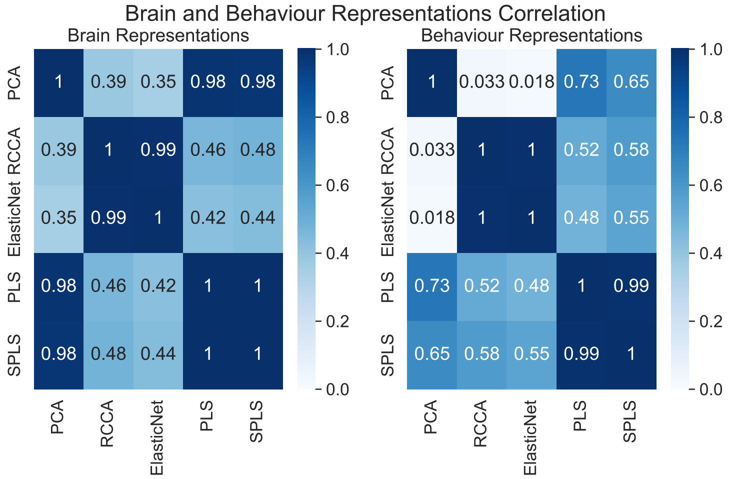


Figure III.5: HCP: Pairwise correlation matrix of brain representations across different models. The high correlation coefficients between PCA, PLS, and SPLS indicate a significant overlap in the brain representations they produce, suggesting a bias of PLS toward principal components. Contrarily, the Ridge CCA and Elastic Net models show notably lower correlations with PCA, indicating that these models capture brain representations beyond the first principal components.

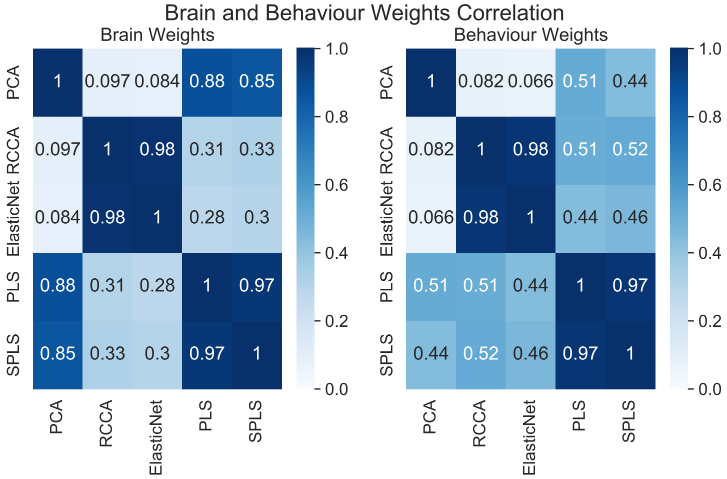


Figure III.6: HCP: Pairwise correlation matrix of the brain and behaviour weights used by each model. Similar to the brain representations, PCA, PLS, and SPLS show a high correlation in their weights, indicating similarity in the factors they consider significant. The lower correlations observed for Ridge CCA and Elastic Net with PCA suggest that these models give importance to different aspects of the data, potentially capturing more nuanced relationships.

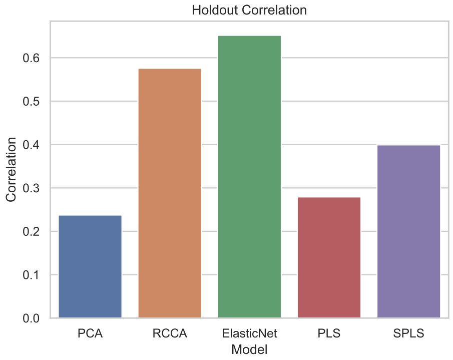


Figure III.7: ADNI: Comparative out-of-sample canonical correlations among PCA, RCCA, ElasticNet, PLS, and SPLS models. The bars represent the correlation coefficients, indicating that the Elastic Net models has superior performance over Ridge CCA, PLS, and SPLS in capturing holdout correlation.

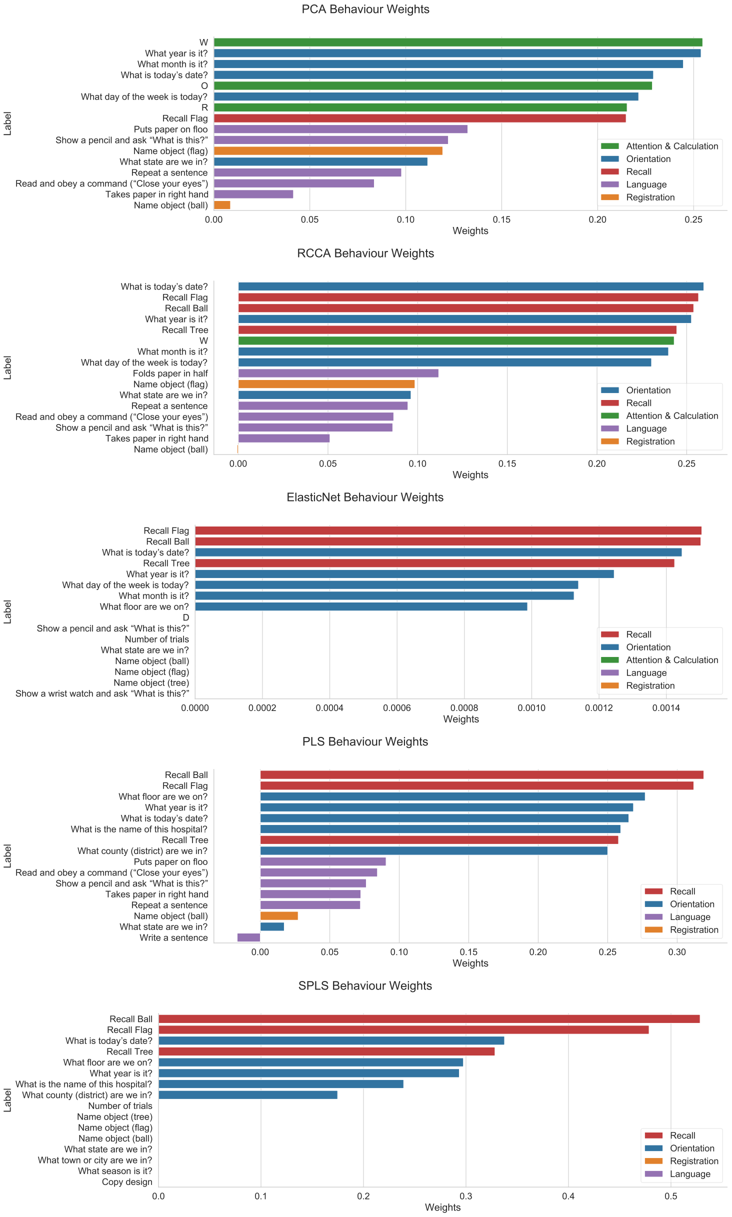


Figure III.8: ADNI: Bar plots of the behaviour weights for each model.

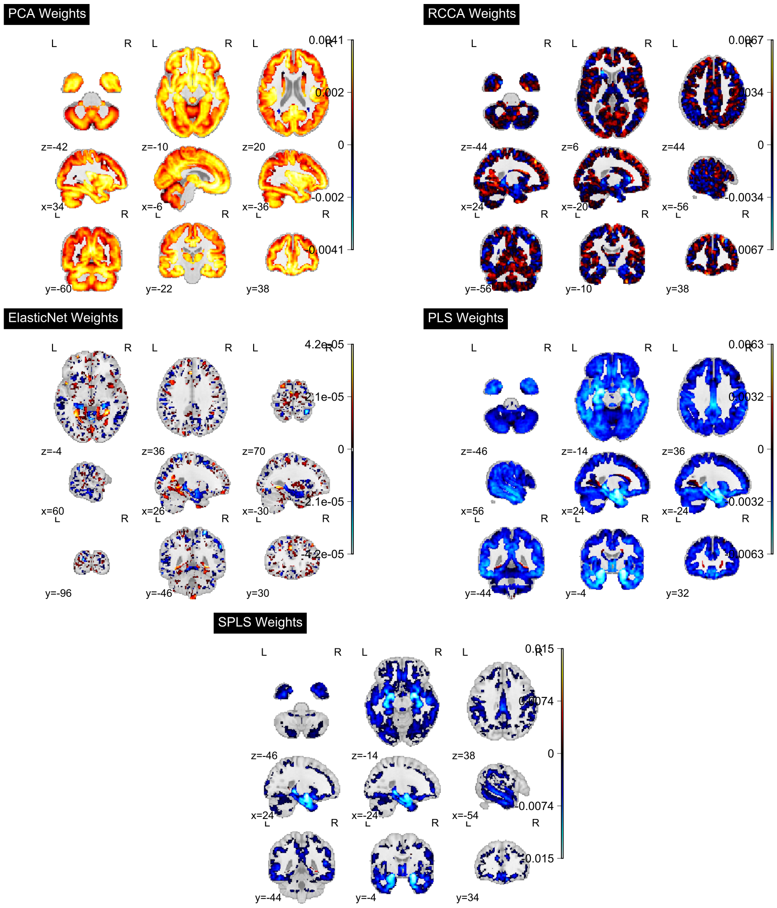


Figure III.9: ADNI: Statistical maps of brain structure weights for each model.

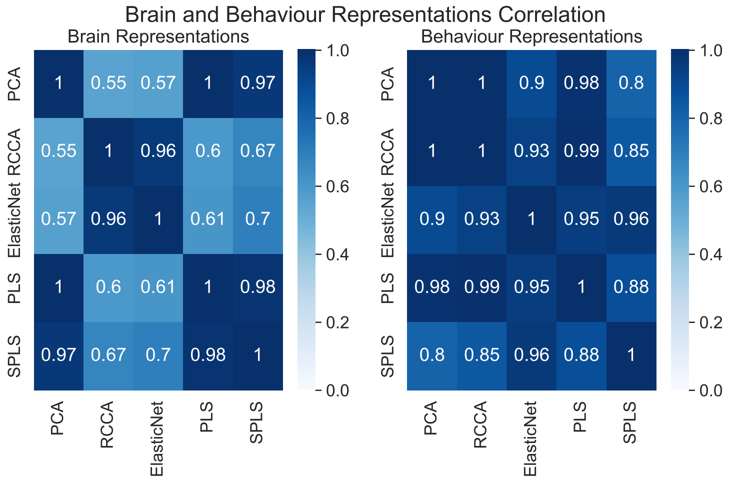


Figure III.10: ADNI: Correlation between the brain and behaviour representations for each model.

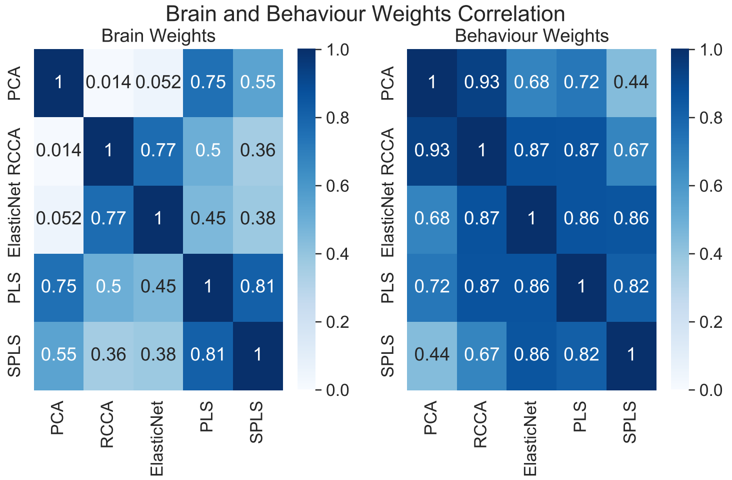


Figure III.11: ADNI: Correlation between the brain and behaviour weights for each model.

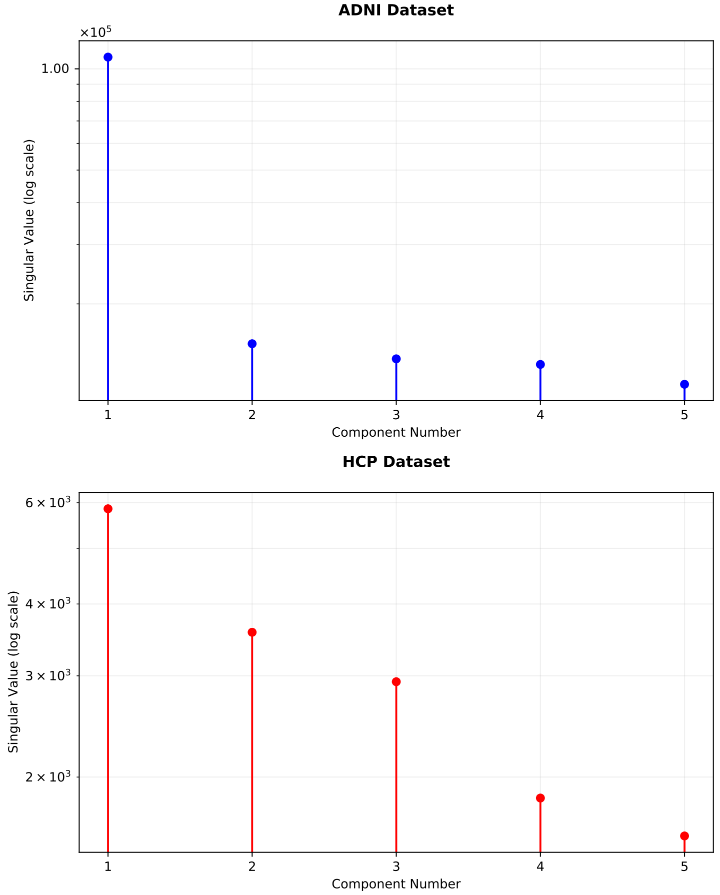


Figure III.12: Singular value spectrum from PLS analysis showing a clear eigengap between the first and subsequent components for both ADNI and HCP datasets. While we cannot perform consistent deflation across all methods to compare secondary components, this analysis suggests the primary brain-behavior mode is well-separated and stable.

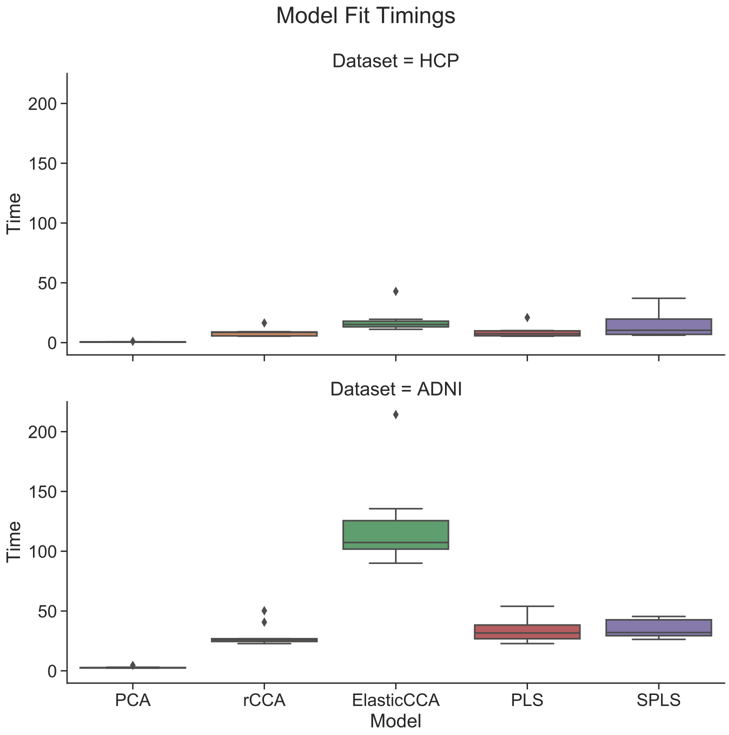


Figure III.13: Time taken (in seconds) to fit each model over ten runs. The interquartile range is plotted as a box with whiskers drawn to the farthest datapoint within 1.5 times the interquartile range.

Chapter IV

Insights From Generating Simulated Data for CCA

Contents

Preface

This chapter, deriving insights from various projects, lays out both my arguments for the use of loadings in the interpretation of CCA models and a number of computational tricks that we used to generate simulated data with significantly higher dimensions than have been previously considered in the literature. The simulated data generation methods were used to generate simulated data in Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al. (2022). The arguments for the use of loadings influenced our choice of loadings for the interpretation of the results in Adams et al. (2024).

1 Introduction

Despite its popularity, there is an ongoing debate in the CCA literature regarding the interpretation of model weights versus loadings (Gu and Wu, 2018). This chapter aims to contribute to this debate by providing mathematical insights from generative models of CCA and empirical results from simulated data with higher dimensionality than previously considered in the literature.

We begin by categorizing methods for generating CCA simulated data into explicit and implicit latent variable models. This categorization allows us to compare and contrast the generative models in CCA literature with the generative model for linear regression. We highlight that in linear regression, regularization can be interpreted as a prior on the weights, whereas in CCA, it is perhaps more natural to interpret regularization as a prior on the loadings. By leveraging computational tricks, we demonstrate how to generate simulated data with significantly higher dimensions than previously considered in the literature (Helmer et al., 2020; Matkovič et al., 2023).

Furthermore, we rigorously prove that loadings are invariant to columnwise transformations in data matrices, unlike weights. This property makes CCA unique compared to Principal Component Analysis (PCA) or Partial Least Squares (PLS) and is particularly relevant in fields like brain-behavior studies, where data preprocessing often involves columnwise manipulation.

Our experimental design focuses on two main aspects. First, we evaluate the ability of CCA models to accurately recover the true model weights and loadings. Second, we examine the out-of-sample performance, which is often observed to be poor in practical datasets despite statistical significance, particularly for PLS-based models. This observation led us to question whether the issue lies in poor model fit or a lack of signal in the data with weak or biologically spurious correlations.

One of our most striking findings, consistent with the previous chapter, is the efficacy of Ridge Regularized CCA models compared to PLS models in identifying high correlations under anisotropic noise conditions; where the noise covariance matrices (Ψ) are not scalar multiples of the identity matrix, leading to non-spherical noise distributions. This complements earlier work (Helmer et al., 2020) that found that the number of samples needed to find high correlations increases with dimensionality; our results suggest that the important variable is the dimensionality of the smaller view.

Through this chapter, we aim to provide a comprehensive understanding of the relationship between weights and loadings in CCA models, the impact of regularization on model interpretation, and the performance of CCA models in high-dimensional settings. By unifying generative perspectives, proving mathematical properties, and conducting extensive simulations, we contribute to the ongoing debate in the CCA literature and provide valuable insights for researchers and practitioners applying CCA in various domains.

2 Background: Weights and Loadings in Canonical Correlation Analysis

CCA can be interpreted in two ways: either as a method that finds linear combinations of variables in two datasets that exhibit the highest correlation, or as a technique that estimates latent variables that are maximally correlated.

The concept of latent variables is particularly important in biomedical applications, as it can help uncover underlying factors influencing observable data. For example, in brain-behavior studies, latent variables may represent hidden neurological or cognitive processes that drive the relationship between brain structure or function and behavioral outcomes. Similarly, in imaging-genetics, latent variables can capture the genetic factors that influence brain morphology or activity patterns. By introducing latent variables, CCA enables researchers to gain a deeper understanding of complex phenomena like gene expression, pathologies, and normative variations in health-related data (Lawry Aguila, Chapman, and Altmann, 2023).

2.1 Generative and Discriminative Approaches in CCA

CCA's practical application revolves around two main approaches: the discriminative approach and the generative approach. The generative approach, known as the 'forward model', emphasizes the data generation process and employs loadings to describe the relationship between latent variables and observed data. It models the joint distribution of the observed data conditioned on the latent variables, expressed as $P(X^{(1)}, X^{(2)}|Z)$. In this approach, the latent variables are assumed to be the underlying cause[1] of the observed data, and the goal is to learn the parameters of the generative model that best explains the data.

In contrast, the discriminative approach, represented as the 'backward model' in Figure IV.1, uses weights to estimate highly correlated latent variables from observed

[1] Used here very loosely to give intuition

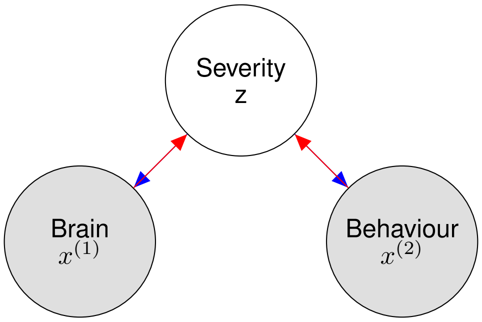


Figure IV.1: ***Forward and Backward Multiview Models:*** The generative/forward and discriminative/backward approaches in CCA.

data. It focuses on modeling the conditional distribution of the latent variables given the observed data, denoted as $P(Z|X^{(1)}, X^{(2)})$. The discriminative approach aims to find the optimal linear combinations of the observed variables that maximize the correlation between the latent variables, without explicitly modeling the data generation process.

2.2 Analogy Between CCA and PCA

The distinction between the generative and discriminative approaches in CCA is analogous to the different interpretations of Principal Component Analysis (PCA) (Park, Ceulemans, and Van Deun, 2023). PCA can also be viewed from a generative perspective, where the observed data are assumed to be generated from latent variables (Tipping and Bishop, 1999), or from a discriminative perspective, where the principal components are linear combinations of the observed variables that maximize the variance (Hotelling, 1933).

In both CCA and PCA, the generative approach focuses on modeling the joint distribution of the observed data and the latent variables, while the discriminative approach emphasizes finding the optimal linear combinations of the observed variables to estimate the latent variables or principal components.

2.3 The Debate Regarding Weights and Loadings in CCA

In CCA research, there is an ongoing debate regarding the interpretation of models in terms of weights or loadings (Gu and Wu, 2018). Weights are often preferred for prediction tasks, as they directly relate the observed variables to the latent variables. On the other hand, loadings are favored for interpretation, as they provide insights into the structure and relationships within the data (Z. Liu et al., 2022). This discussion is particularly relevant to our work in chapter III and various studies involving sparse CCA and sparse Partial Least Squares (PLS), where understanding the meaning and implications of sparse loadings and weights is crucial.

Given the importance of this topic, especially in the context of our work in chapter III and other studies employing variants of sparse CCA and sparse PLS, it is essential to delve deeper into the interpretation of sparse loadings and weights. In the following sections, we will explore the mathematical properties and practical implications of weights and loadings in CCA, with a focus on sparse and regularized models. By addressing this key aspect of CCA, we aim to contribute to the ongoing debate and provide insights that can guide the application and interpretation of CCA in various domains, including neuroimaging, genetics, and health-related research.

3 Methods: Unifying Generative Perspectives in CCA: Explicit and Implicit Latent Variable Models of Multiview Data

This section categorizes the generative models in CCA literature into explicit and implicit latent variable types, each offering distinct insights into the data generation process and the relationship between weights and loadings.

3.1 Additional Notational Conventions

We will use some additional notational convention to describe probabilistic models. We will use lowercase letters to represent samples from a distribution, and uppercase letters to represent random variables. For example, x represents a sample from the distribution $P(X)$, and X represents the random variable X. We will use $\sim$ to denote the sampling process, and $|$ to denote conditioning. For example, $x|z \sim \mathcal{N}(\mu, \Psi)$ represents a sample x from a Gaussian distribution with mean Wz and covariance Ψ conditioned on the latent variable z. We also introduce the notation $w_j^{(i)}$ to refer to the loading of the j-th feature in the i-th view on a latent variable, as well as $W^{(i)}$ to refer to the matrix of loadings for the i-th view on all latent variables.

3.2 Explicit Latent Variable Models: Probabilistic CCA and Group Factor Analysis

Explicit latent variable models assume that the observed data in each view is generated from a shared latent space, with view-specific linear transformations and added noise. These models provide a probabilistic framework for understanding the relationship between multiple views of data and the underlying latent factors that give rise to them.

3.2.1 Probabilistic CCA

Probabilistic CCA (PCCA) is an explicit latent variable model that extends classical CCA to a probabilistic setting. In PCCA, the generative process for two views can be described as follows:

$$z \sim \mathcal{N}(0, I) \tag{IV.1}$$

$$x^{(i)} \sim \mathcal{N}(W^{(i)}z + \mu^{(i)}, \Psi^{(i)}) \tag{IV.2}$$

where z is a shared latent variable drawn from a standard normal distribution, $x^{(i)}$ represents a sample from the i-th view, $W^{(i)}$ are the view-specific loadings that map the latent space to the observed space, $\mu^{(i)}$ is the mean of the i-th view, and $\Psi^{(i)}$ is the view-specific noise covariance matrix.

The key idea behind PCCA is that the shared latent variable z captures the common structure across views, while the view-specific loadings $W^{(i)}$ allow for flexibility in how this structure is expressed in each view. The noise covariance matrices $\Psi^{(i)}$ account for view-specific variation not explained by the shared structure. Bach and Jordan (2005) showed that the maximum likelihood estimates of the loadings $W^{(i)}$ in PCCA are related to the classical CCA weights $U^{(i)}$ by the within-view covariance matrices Σ_{ii}:

$$W^{(i)} = \Sigma_{ii}U^{(i)}R \tag{IV.3}$$

$$U^{(i)}R = \Sigma_{ii}^{-1}W^{(i)} \tag{IV.4}$$

where R is an arbitrary rotation matrix. While this provides a mathematical link between the probabilistic and non-probabilistic formulations of CCA, the correlation structure within each view (Σ_{ii}) means that the weights and loadings can be quite different in practice. The weights $U^{(i)}$ and loadings $W^{(i)}$ capture different aspects of the relationship between the views, with the loadings directly modeling how the latent variables manifest in each view's observations.

3.2.2 Group Factor Analysis

Group Factor Analysis (GFA) (Klami et al., 2014) is another explicit latent variable model that extends PCCA by assuming isotropic (i.e., spherical) noise in each view:

$$z \sim \mathcal{N}(0, I)\, x^{(i)} \qquad \sim \mathcal{N}(W^{(i)} z + \mu^{(i)}, \sigma_i^2 I) \tag{IV.5}$$

where σ_i^2 is the noise variance in the i-th view. This assumption simplifies the model and can lead to computational benefits, as well as supporting extensions like sparsity on the loadings. The joint distribution of multiple views under the GFA model is a multivariate Gaussian distribution with a structured covariance matrix:

$$\left[X^{(1)} \vdots X^{(m)}\right] \sim \mathcal{N}\left(\left[\mu^{(1)} \vdots \mu^{(m)}\right], \begin{bmatrix} W^{(1)} W^{(1)\,T} + \sigma_1^2 I & \cdots & W^{(1)} W^{(m)\,T} \\ \vdots & \ddots & \vdots \\ W^{(m)} W^{(1)\,T} & \cdots & W^{(m)} W^{(m)\,T} + \sigma_m^2 I \end{bmatrix}\right) \tag{IV.6}$$

This highlights the fact that the covariance structure in each view is determined by the view-specific loadings and noise variances. When the loadings $W^{(i)}$ have large singular values compared to the noise variances σ_i^2, the resulting covariance matrices $\Sigma_{ii} = W^{(i)} W^{(i)\,T} + \sigma_i^2 I$ are often referred to as "spiked covariance matrices" (Johnstone, 2001).

3.2.3 Connecting GFA and Probabilistic PCA

Probabilistic PCA (PPCA) (Tipping and Bishop, 1999) is a special case of GFA applied to a single view:

$$z \sim \mathcal{N}(0, I) \tag{IV.7}$$

$$x \sim \mathcal{N}(Wz + \mu, \sigma^2 I) \tag{IV.8}$$

Like GFA, PPCA assumes isotropic noise and seeks to identify a lower-dimensional latent space that captures the structure in the observed data. The key difference is that PPCA does not model multiple views or the relationships between them. However, the connection between GFA and PPCA provides an important insight: when the noise in each view is low and isotropic, the shared latent structure dominates the view-specific noise. In this scenario, applying PPCA to a single view should recover latent variables that are very similar to those obtained by applying GFA to multiple views. This suggests that, in low-noise settings, it may be possible to uncover meaningful latent structure using just a single view of the data. Consequently, PPCA can serve as a useful baseline for evaluating the performance of multi-view models like GFA. Likewise in the discriminative setting, if a multi-view model does not significantly outperform PCA applied to a single view, it may indicate that the additional complexity of the multi-view approach is not justified for that particular dataset. For this reason, it is always my recommendation to compare the performance of multi-view models to that of PCA applied to each individual view[2]. This can help assess whether the multi-view approach is truly leveraging complementary information across views, or if the observed performance gains are due to other factors, such as noise reduction or increased model capacity. By understanding the relationships between these explicit latent variable models, researchers can make more informed decisions about when and how to apply them to real-world datasets, and can better interpret the results obtained from multi-view analyses.

3.3 Implicit Latent Variable Models: The Joint Covariance Matrix Perspective

The joint covariance matrix perspective, prevalent in sparse CCA literature (Suo et al., 2017; M. Chen et al., 2013), emphasizes covariance matrices over direct modeling of latent variables. This approach allows us to directly control the sparsity of the weights and the strength of the canonical correlations by constructing the covariance matrices accordingly. By focusing on the covariance structure, we can generate data with desired properties without explicitly modeling the latent variables. This is achieved by constructing the joint covariance matrix of the distribution $P(X^{(1)}, X^{(2)})$:

$$\begin{bmatrix} X^{(1)} \\ X^{(2)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right) \tag{IV.9}$$

Where Σ_{11} and Σ_{22} are the within-view covariance matrices and Σ_{12} and Σ_{21} are the between-view covariance matrices.

For clarity and simplicity in our discussion, we refer to a single canonical correlation coefficient, ρ, without loss of generality. This allows us to focus on the structure of the covariance matrices without the complexity of multiple canonical correlations.

[2] In the growing Deep Multiview Learning literature this is analagous to comparing to separate autoencoders applied to each view

In constructing the between-view covariance matrices Σ_{12} and Σ_{21}, we control the true signal by setting active variables and correlations. Specifically, the between-view covariance matrix is constructed as follows:

$$\Sigma_{12} = \rho \Sigma_{11} u_1^{(1)} u_1^{(2)\,T} \Sigma_{22} \tag{IV.10}$$

Here, ρ is the canonical correlation, and $u_1^{(i)}$ is the first column of the matrix of weights $U^{(i)}$ for the i-th view.

This perspective simplifies the structure of covariance matrices, focusing on the relationship between views as controlled by the canonical correlation coefficient, ρ, and the weights $u^{(i)}$.

3.4 Noise Structures and Their Impact on Covariance Modeling

The noise covariance matrices $\Psi^{(i)}$ in the explicit latent variable models play a crucial role in determining the nature of the noise in each view. When the noise covariance matrix is a scalar multiple of the identity matrix, i.e., $\Psi^{(i)} = \sigma_i^2 I$, the noise is considered isotropic or spherical, with equal variance in all dimensions and no correlations. On the other hand, when the noise covariance matrix is not a scalar multiple of the identity matrix, the noise is considered anisotropic or non-spherical, with potentially different variances across dimensions and correlations between noise components. In real-world datasets, such as those involving brain and behavioral data, the assumption of isotropic noise may not always hold. The correlation between brain regions or behavioral measures refers to the structured relationships in the data, captured by the loadings $W^{(i)}$ in the explicit latent variable models. The noise, on the other hand, represents the unstructured variation or measurement error not explained by the latent factors. While brain regions and behavioral measures may be correlated due to shared underlying processes, this does not necessarily imply that the noise itself is correlated. When analyzing brain and behavioral data using explicit latent variable models or related methods, it is important to consider the potential for both correlated and uncorrelated noise structures. Ignoring the possibility of correlated noise and assuming an identity covariance matrix may oversimplify the noise characteristics, potentially leading to suboptimal results or misinterpretations. Conversely, assuming correlated noise when it is not present can lead to overparameterization and reduced interpretability. In addition to the noise covariance, the observed covariance structures are also important to consider. Discriminative methods, such as classical CCA, often make assumptions about the observed covariance matrices Σ_{ii}, taking them as given or estimating them from the data. In contrast, generative methods, such as Probabilistic CCA and GFA, model the observed covariance as a function of the loadings and the noise covariance, as shown in Table 3.1. Researchers should carefully examine the properties of the data and use prior knowledge about the measurement process to guide their assumptions about both the noise and observed covariance structures. Comparing models with different assumptions (e.g., isotropic vs. anisotropic noise, identity vs. non-identity observed covariance) and using model selection techniques can help determine the most appropriate choice when the covariance structures are uncertain. By considering the potential for different noise and observed covariance structures, researchers can make more informed decisions about the modeling assumptions and improve the accuracy and interpretability of their results.

3.5 Summary of Data Generation Methods

To summarize the key differences between the data generation methods discussed above, we present two tables. Table 3.1 compares the covariance structures of each method, highlighting how the within-view and between-view covariances are modeled. Table 3.2 illustrates the relationship between weights and loadings in both population and sample cases, emphasizing the implications for sparsity and identifiability.

Table 3.1: Covariance Structures in Data Generation Methods

	Method	**Within-view Covariance Σ_{ii}**	**Between-view Covariance Σ_{12}**
Explicit	Probabilistic CCA	$W^{(i)}W^{(i)\,T} + \Psi^{(i)}$	$W^{(1)}W^{(2)\,T}$
	GFA	$W^{(i)}W^{(i)\,T} + {\sigma^{(i)}}^2 I$	$W^{(1)}W^{(2)\,T}$
Implicit	Joint Covariance	Σ_{ii}	$\rho\Sigma_{11}u_1^{(1)}{u_1^{(2)}}^T\Sigma_{22}$
	Joint Covariance (Identity)	I	$\rho u_1^{(1)}{u_1^{(2)}}^T$

As shown in Table 3.1, the explicit latent variable models (Probabilistic CCA and GFA) incorporate the noise covariance matrices $\Psi^{(i)}$ or $\sigma_i^2 I$ into the within-view covariance expressions. This allows for more flexible modeling of the noise structure, as the noise covariance can be either isotropic (in the case of GFA) or anisotropic (in the case of Probabilistic CCA). In contrast, the implicit latent variable models (Joint Covariance) assume a simpler noise structure, often taking the within-view covariance matrices Σ_{ii} as given or assuming an identity covariance for tractability.

Table 3.2 summarizes the relationship between the weights and loadings in each data generation method, distinguishing between population and sample cases. This distinction is crucial, especially in scenarios where the population covariance matrix Σ is identity, but the sample covariance matrix $\hat{\Sigma}$ is only an approximation. An important observation is that for the implicit latent variable models, we can generate data with sparse weights but not, in general, sparse loadings. For the explicit latent variable models, we can generate data with sparse loadings but not, in general, sparse weights.

Table 3.2: Relationship Between Weights and Loadings in Population and Sample Cases

	Method	**Case**	**Weights**	**Loadings**
Explicit	Probabilistic CCA	Population	$(W^{(i)}W^{(i)\,T} + \Psi^{(i)})^{-1}W^{(i)}$	$W^{(i)}$
		Sample	$\hat{\Sigma_{ii}}^{-1}W^{(i)}$	$W^{(i)}$
	GFA	Population	$(W^{(i)}W^{(i)\,T} + {\sigma^{(i)}}^2 I)^{-1}W^{(i)}$	$W^{(i)}$
		Sample	$\hat{\Sigma_{ii}}^{-1}W^{(i)}$	$W^{(i)}$
Implicit	Joint Covariance (Non-Identity)	Population	$U^{(i)}$	$\Sigma_{ii}U^{(i)}$
		Sample	$U^{(i)}$	$\hat{\Sigma_{ii}}\hat{U^{(i)}}$
	Joint Covariance (Identity)	Population	$U^{(i)}$	$U^{(i)}$
		Sample	$U^{(i)}$	$\hat{\Sigma_{ii}}\hat{U^{(i)}}$

3.6 Regularization and Generative Models

Regularization is crucial in CCA to prevent overfitting and promote interpretability. However, the way regularization is interpreted in CCA differs from linear regression due to the latent variable nature of CCA models. In linear regression, regularization can be directly interpreted as a prior on the weights. In contrast, for CCA, regularization can be interpreted as a prior on either the loadings or the weights, depending on the generative perspective. This distinction has important implications for model interpretation and the identifiability of weights in CCA.

3.6.1 Regularization and the Generative Model for Linear Regression

Linear regression assumes data generation from a linear model with added noise:

$$y = xU + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2 I) \tag{IV.11}$$

Here, y are samples of the target variable, x samples from the data matrix, U the regression coefficients, and ϵ represents independent and identically distributed (i.i.d.) Gaussian noise.

Lasso Regression The Lasso imposes a Laplace prior on the regression coefficients, leading to a double-exponential prior on weights:

$$U \sim \mathcal{L}(0, \lambda) \tag{IV.12}$$

Ridge Regression Ridge regression, in contrast, employs a Gaussian prior on the regression coefficients, equivalent to a Gaussian prior on weights:

$$U \sim \mathcal{N}(0, \lambda) \tag{IV.13}$$

3.6.2 Regularization and Generative Models for CCA

CCA models differ in their approach to regularization compared to linear regression because they are latent variable models.

Explicit Latent Variable Model Regularization in the context of the explicit latent variable naturally relates to priors on the loadings $W^{(i)}$. For example, sparsity in the loadings can be achieved by imposing a Laplace prior on the loadings:

$$W^{(i)} \sim \mathcal{L}(0, \lambda) \tag{IV.14}$$

This expresses the prior belief that latent factors only explain the data through a small number of features. For example, in the context of latent factors in brain-behavior studies, this prior belief is equivalent to the assumption that a latent mode of variance (perhaps a subtype) is only expressed through a small number of brain regions.

Implicit Latent Variable Model In the implicit latent variable model of CCA, the joint likelihood is modeled as a block covariance matrix Σ(Suo et al., 2017), constructed from the weights $U^{(i)}$.

$$\Sigma = \begin{bmatrix} \Sigma_1 & \Sigma_1 U^{(1)} \rho U^{(2)\,T} \Sigma_2 \\ \Sigma_2 U^{(2)} \rho U^{(1)\,T} \Sigma_1 & \Sigma_2 \end{bmatrix} \tag{IV.15}$$

Where the off-diagonal blocks $\Sigma_1 U^{(1)} \rho U^{(2)\,T} \Sigma_2$ and its transpose represent the between-view covariance matrices. These matrices are functions of the weights $U^{(i)}$ and within-view covariance matrices Σ_i, modulated by ρ, the canonical correlation coefficients.

Here the regularization naturally relates to priors on the weights $U^{(i)}$. For example, sparsity in the weights can be achieved by imposing a Laplace prior on the weights:

$$U^{(i)} \sim \mathcal{L}(0, \lambda) \tag{IV.16}$$

This expresses the more nuanced prior belief that the latent factors are expressed through a subset of features and then distorted by arbitrary rotations as well as the within-view covariance matrices. Manipulating equation IV.3, the conditional distribution of the implicit latent variable model we have:

$$x^{(i)}|z \sim \mathcal{N}(\Sigma_i U^{(i)} R z = W^{(i)} z, \Sigma_i - W^{(i)} W^{(i)\,T} = \Psi^{(i)}) \tag{IV.17}$$

$$z \sim \mathcal{N}(0, I) \tag{IV.18}$$

The arbitrary rotation matrix R means that for multidimensional $U^{(i)}$, even if $\Sigma_i = I$, and even if the true loadings are sparse, the weights may still not be sparse!

$$x^{(i)}|z \sim \mathcal{N}(U^{(i)}Rz = W^{(i)}_{sparse}z, \Sigma_i - W^{(i)}_{sparse}W^{(i)\,T}_{sparse} = \Psi^{(i)}) \tag{IV.19}$$

$$z \sim \mathcal{N}(0, I) \tag{IV.20}$$

Alternatively, even if we know the true weights (i.e. $R = I$), the CCA model may not be able to recover them. This is to say they are not, in general, identifiable (Park, Ceulemans, and Van Deun, 2023). In other words there are multiple values of $W^{(i)}$ that can produce the same covariance structure.

We can illustrate this with a trivial example:

$$\Sigma_1 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{IV.21}$$

(IV.22)

In this example, we show that the same covariance matrix Σ_1 can be obtained using different weight matrices. The first weight matrix has entries [1, 0, 1], while the second weight matrix has entries [0.5, 0.5, 1]. This example clearly demonstrates the non-identifiability issue in the implicit latent variable model, where multiple weight matrices can produce the same covariance structure. This means that even if we know the true covariance structure, we may not be able to uniquely recover the true weights.

One practical implication of this observation is that it raises serious questions about using stability selection, a common practice in the sparse CCA literature (Mihalik, Ferreira, Moutoussis, et al., 2020; Deng et al., 2021), to select the optimal regularization parameter.For instance, suppose we run stability selection multiple times on the same dataset to select the optimal regularization parameter. Due to the non-identifiability of weights, each run may result in different rotations of the weights, even though the underlying representations and correlations remain the same. This can lead to inconsistent selection of the regularization parameter across runs, potentially resulting in suboptimal hyperparameter choices or incorrect conclusions about the sparsity structure of the data.

In summary, understanding the generative perspectives in CCA is crucial for interpreting regularization, sparsity, and identifiability in these models. The explicit latent variable model allows for intuitive priors on the loadings, while the implicit latent variable model enables priors on the weights, albeit with less straightforward interpretations. The non-identifiability issue in the implicit model highlights the challenges in recovering unique weights and raises questions about the reliability of stability selection. By considering these generative perspectives, researchers can make more informed choices when applying regularization and interpreting the results of CCA models.

4 Invariance of Loadings in CCA: An Intuitive Mathematical Argument

In this section, we present an intuitive mathematical argument for favoring loadings over weights in the interpretation of CCA models. We will demonstrate that loadings are invariant to certain common transformations of the data matrix, including scaling, duplication, and summation of columns. This property is not shared by weights. This invariance has significant practical implications, especially when working with heterogeneous or transformed data.

4.1 Solving CCA in Principal Component Space

Consider the singular value decomposition (SVD) of the data matrices:

$$X^{(i)} = U^{(i)}S^{(i)}V^{(i)\,T} \tag{IV.23}$$

Here, $U^{(i)}$ contains the left singular vectors (principal components) of $X^{(i)}$, $S^{(i)}$ is a diagonal matrix of singular values, and $V^{(i)}$ contains the right singular vectors. The columns of $U^{(i)}$ span the column space of $X^{(i)}$, which is the space of all possible linear combinations of the columns of $X^{(i)}$. Intuitively, the column space captures all the directions in which the data varies.

The CCA objective is to find weights $u^{(1)}, u^{(2)}$ that maximize the correlation between the canonical variables $X^{(1)}u^{(1)}$ and $X^{(2)}u^{(2)}$:

$$\max_{u^{(1)},u^{(2)}} \mathrm{Corr}(X^{(1)}u^{(1)}, X^{(2)}u^{(2)}) = \max_{u^{(1)},u^{(2)}} \mathrm{Corr}(U^{(1)}S^{(1)}V^{(1)\,T}u^{(1)}, U^{(2)}S^{(2)}V^{(2)\,T}u^{(2)}) \tag{IV.24}$$

By reparameterizing the weights as $v^{(i)} = S^{(i)}V^{(i)\,T}u^{(i)}$, we obtain:

$$\max_{v^{(1)},v^{(2)}} \mathrm{Corr}(U^{(1)}v^{(1)}, U^{(2)}v^{(2)}) \tag{IV.25}$$

This shows that CCA can be solved entirely in the principal component space spanned by the matrices $U^{(i)}$. The loadings $w_j^{(i)}$, defined as the correlations between the original features $X_j^{(i)}$ and the canonical variables $U^{(i)}v^{(i)}$, capture the relationships in this space.

4.2 Invariance of Loadings to Data Transformations

We now show that the loadings are invariant to certain transformations of the data matrix $X^{(i)}$, while the weights are not. The key insight is that these transformations change the right singular vectors $V^{(i)}$ and singular values $S^{(i)}$, but not the left singular vectors $U^{(i)}$. Since the loadings depend only on $U^{(i)}$, they remain invariant. Moreover, these transformations preserve the column space of $X^{(i)}$, which is why the principal components $U^{(i)}$ are unaffected.

4.2.1 Scaling Transformation

Consider a diagonal scaling matrix B that scales the columns of $X^{(i)}$:

$$\tilde{X}^{(i)} = X^{(i)}B = (U^{(i)}S^{(i)}V^{(i)\,T})B = U^{(i)}(S^{(i)}B)(V^{(i)\,T}) \tag{IV.26}$$

The weights $\tilde{u}^{(i)}$ in the transformed space are related to the original weights by $\tilde{u}^{(i)} = B^{-1}u^{(i)}$, and thus change with the scaling. However, the principal components $U^{(i)}$ remain unchanged, so the loadings $\tilde{w}_j^{(i)} = \mathrm{Corr}(\tilde{X}_j^{(i)}, U^{(i)}v^{(i)}) = w_j^{(i)}$ are invariant.

4.2.2 Duplication Transformation

Consider duplicating columns of $X^{(i)}$ using a transformation matrix B that contains an identity matrix I_n and a duplication matrix D:

$$B = \begin{bmatrix} \\ I_n \; D \end{bmatrix}, \quad \tilde{X}^{(i)} = X^{(i)}B = U^{(i)}(S^{(i)}B)(V^{(i)\,T}) \tag{IV.27}$$

The weights $\tilde{u}^{(i)}$ become underdetermined in the transformed space due to the added linear dependencies. However, since the column space of $\tilde{X}^{(i)}$ is the same as that of $X^{(i)}$, the principal components $U^{(i)}$ and thus the loadings remain unchanged.

4.2.3 Linear Combination Transformation

Consider adding or removing linear combinations of columns using a transformation matrix B that contains an identity matrix I_n and a coefficient matrix C:

$$B = \begin{bmatrix} & \\ I_n & C \end{bmatrix}, \quad \tilde{X}^{(i)} = X^{(i)}B = U^{(i)}(S^{(i)}B)(V^{(i)\,T}) \tag{IV.28}$$

As before, the weights $\tilde{u}^{(i)}$ change in the transformed space, but since the column space is preserved, the principal components $U^{(i)}$ and loadings remain invariant.

4.3 Practical Implications

The invariance of loadings to data transformations has significant practical implications, especially in fields like biomedical research, psychometrics, and social sciences where questionnaire and survey data are common:

- **Interpretability**: Loadings provide a consistent interpretation of the relationships between the original features and the canonical variables, even if the data is rescaled or transformed. This is particularly valuable in interdisciplinary research, where different data normalization practices may be employed.

- **Feature Selection**: Decisions about including, excluding, or combining features can be made based on the loadings without worrying about their impact on the CCA solution. This is especially relevant when dealing with summary measures that effectively sum other variables, a common scenario in questionnaire and biomedical data. The invariance of loadings to such alterations in the data structure makes them a more robust choice for interpreting relationships between variables in these contexts.

- **Robustness**: CCA models can be trained on transformed data (e.g., normalized or standardized) while still allowing for meaningful interpretation in the original feature space. While the identifiability of weights can be partially solved by the standardization of data, and while this is a common practice, it is not always necessary or desirable and always introduces assumptions.

In conclusion, this section provides a strong mathematical foundation for the preference of loadings over weights in the interpretation of CCA models. The invariance of loadings to columnwise transformations, including scaling and linear combinations, ensures a more robust and consistent interpretation of variable relationships. This property is especially valuable in fields dealing with heterogeneous or transformed data, where data preprocessing choices may vary. By focusing on loadings, researchers can obtain more reliable insights into the underlying structure of their data, facilitating cross-disciplinary collaborations and the advancement of knowledge.

5 Methods: Efficient Sampling of Simulated CCA Data

Efficient sampling is crucial for CCA because it allows researchers to work with larger datasets and explore more complex or more nuanced relationships between variables, ultimately expanding the scope of research and analysis. Traditional methods can be computationally intensive and storage-demanding, especially for large datasets. This has in practice limited the dimensionality of simulated data, restricting the scope of research and analysis. For example Matkovic et al. (2023) simulate data with 8,000 observations and 100 features while Helmer et al. (2020) used at most 10,000 observations and 64 features. We were interested in the behavior of CCA in high-dimensional settings like voxel-wise MRI and brain connectivities, which can have hundreds of thousands of features (Jack Jr et al., 2008) and up to tens of thousands of observations (Sudlow et al., 2015). By leveraging the assumptions that biomedical data often exhibit low-rank and/or sparse covariance structures, we develop efficient sampling methods that overcome the computational and storage limitations associated with high-dimensional data.

5.1 Challenges with High-Dimensional Data

Direct sampling from a multivariate normal distribution is impractically slow for high-dimensional data, which has been a core research challenge for Monte Carlo methods (Mackay, 1998). The implicit latent variable model, in particular, requires storage of the full covariance matrix, which is prohibitive for high-dimensional data. For example, a covariance matrix with 100,000 dimensions would require 80GB of memory, far exceeding the capacity of most personal computers.

5.2 Efficient Sampling for Explicit Latent Variable Models

The explicit latent variable model offers more efficient approaches for sampling high-dimensional data by employing sparse and low-rank covariance matrices.

5.2.1 Sampling from Multivariate Normal Distributions

An efficient approach to sampling from a multivariate normal distribution is to use the Singular Value Decomposition (SVD) or Cholesky decomposition of the covariance matrix. This involves decomposing the covariance matrix and using the resulting components to transform samples from a standard multivariate normal distribution:

$$Z \sim \mathcal{N}(0, I) \tag{IV.29}$$

$$X = \Sigma^{1/2} Z \tag{IV.30}$$

Where $\Sigma^{1/2}$ is a square root of the covariance matrix, obtained through SVD or Cholesky decomposition. This is the same as the generative model for the explicit latent variable model, where $\Sigma^{1/2}$ is the matrix of loadings. Low-rank noise can be added by sampling from an independent multivariate normal distribution and adding it to the transformed samples. This approach requires sampling from a univariate normal distribution and performing a matrix multiplication of complexity $\mathcal{O}(np^2)$.

5.2.2 Using Sparse and Low-Rank Covariance Matrices

Sparse covariance matrices, with many zero entries, reduce both computational complexity and storage requirements. For example, a sparse covariance matrix with

100,000 dimensions and 10% density would only require 8GB of memory to store.

Low-rank covariance matrices further reduce complexity by storing only the factorized rank-k components, reducing storage requirements to $\mathcal{O}(kp)$. For example, a low-rank covariance matrix with 100,000 dimensions, 10% density, and rank 1000 would only require 80MB of memory to store. This approach also requires drawing $\mathcal{O}(kp)$ samples from a univariate normal distribution and performing a matrix multiplication with complexity $\mathcal{O}(nkp)$, rather than $\mathcal{O}(np^2)$ for the full-rank case.

5.3 Calculating True Canonical Correlations and Weights

The population canonical correlations can be controlled by varying the signal-to-noise ratio (SNR), i.e., the ratio of the signal variance to the noise variance.

For the explicit latent variable model, the loadings are obtained directly as the low-rank square root of the covariance matrix. The weights can be calculated from the loadings and the covariance matrix using the relationship:

$$\hat{W}^{(i)} = \Sigma_{ii}\hat{U}^{(i)}R \tag{IV.31}$$

Where R is an arbitrary rotation matrix and $\hat{U}^{(i)}$ is the matrix of CCA weights for the ith view. For invertible covariance matrices, the 'true' CCA weights associated with the top-k subspace can be accessed by multiplying the loadings by the inverse of the covariance matrix:

$$\hat{U}^{(i)}R = \Sigma_{ii}^{-1}\hat{W}^{(i)} \tag{IV.32}$$

Although inverting the $\mathcal{O}(p^2)$ covariance matrix is computationally expensive, the Sherman-Morrison-Woodbury formula can be used to calculate the inverse in $\mathcal{O}(kp^2)$ time for a rank-k covariance matrix. This allows for the calculation of weights in $\mathcal{O}(kp^2)$ time, which is faster than the $\mathcal{O}(p^3)$ time required to calculate the weights directly from the covariance matrix.

In the next section, we will present experiments demonstrating the relationship between weights and loadings in simulated data using these efficient sampling techniques.

6 Experiment Design

Our goal in this section is to empirically demonstrate the relationship between weights and loadings in CCA models as well as to better understand the behavior of CCA models in the high-dimensional settings that section 5 enables, and which are of interest in the neuroimaging community.

The first set of experiments illustrates the relationship between weights and loadings in simulated data using explicit latent variable models with identity and non-identity covariance matrices. The second set of experiments illustrates the ammount of information that can be recovered from simulated data using CCA and PLS models with varying signal-to-noise ratios and sample sizes.

6.1 Exploring the Relationship Between Weights and Loadings in CCA Using Simulated Data

Our first experiment is designed to illustrate the challenges of recovering the true weights and loadings respectively in CCA models for explicit and implicit latent variable models with identity and non-identity covariance matrices.

We compare the true weights derived from the data generation model with the estimated weights of CCA, Ridge CCA, Elastic Net CCA (implemented with FRALS-EN), PLS, and PCA models. We expect that when the covariance matrix is identity, the weights and loadings will be identical. When the covariance matrix is non-identity, we expect that the weights and loadings will be different. Moreover, we expect that the estimated loadings will be more stable than the estimated weights for CCA models because the weights are not always identifiable. Under the explicit latent variable model, we expect that the weights will only be (close to) sparse when the covariance matrix is close to identity. This means we do not expect the Elastic Net CCA model to improve on the Ridge CCA model since the Lasso regularizes the weights but not the loadings. Finally, we expect that when using the explicit latent variable model, for high signal-to-noise ratios, the PLS and even PCA models will recover the true weights and loadings because the majority of the variance is explained by the latent variables.

6.1.1 Detailed Parameters of Simulated Data for Weights and Loadings Analysis in CCA

We generate data with 100 samples and 10 features in each view. We then generate data under two implicit latent variable models and two explicit latent variable models.

Table 6.1: Simulated Data Parameters for Weight and Loadings Recovery Experiments

Parameter	**Value**
Number of samples (*n*)	100 train, 500 test
Number of features in View 1 (*p*)	10
Number of features in View 2 (*q*)	10
True Latent dimensions	1
Fraction of active features View 1	0.5
Fraction of active features View 2	0.5

The ridge penalty is coarsely tuned between 0.1 and 0.9 in order to illustrate the effect of regularization as we already show the corner cases of no regularization (CCA) and full regularization (PLS). For the Elastic Net CCA model, we tune the l1 ratio between 0.1 and 0.9. This ensures that the Elastic Net CCA has some sparsity as compared to the Ridge model, effectively avoiding the corner case of no sparsity where the Elastic Net CCA is equivalent to the Ridge model. We summarize the parameters of these experiments in table 6.1.

6.2 Assessing Information Recovery in CCA and PLS Models Under Varying Signal-to-Noise Ratios

Our next experiment was motivated by the observation that PLS models (including sparse PLS) often exhibit low but non-zero out of sample correlations in real high-dimensional data. We want to understand how much of this is due to the fact that PLS models optimize covariance rather than correlation, and how much is due to the fact that the signal-to-noise ratio is too low. For brain-behavior relationships specifically, we can estimate likely SNR ranges based on domain knowledge. Measurement noise comes from scanner artifacts, physiological noise, and behavioral measurement error, while signal strength can be bounded using prior studies; for

Table 6.2: Simulated Data Parameters for Brain-Behaviour Simulations

Parameter	**Value**
Number of features in View 1 (*p*)	100-10000
Number of features in View 2 (*q*)	100-10000
True Latent dimensions	1
Fraction of active features View 1	1.0
Fraction of active features View 2	1.0
Signal-to-noise ratio	0.001-1

instance, heritability studies suggest upper bounds on brain-behavior relationships, as shared variance cannot exceed heritability estimates. These considerations suggest operating in the lower SNR regime of our simulations. Since we are interested in studying these effects in high-dimensional data, we aimed to simulate data with similar numbers of features to real brain-behavior datasets. This means that we are only able to use our memory-efficient sampling methods for the explicit latent variable model.

6.2.1 Detailed Parameters of Simulated Data for Signal-to-Noise Simulations

We simulated data with 1000 samples and between 100 and 10,000 features in one view and 100 features in the other. These are of the same order of magnitude as typical brain-behaviour datasets. We summarise these data properties in table 6.2.

6.3 Methodology for Constructing Correlated Covariance Matrices in CCA Simulations

In both experiments, we construct correlated covariance matrices by generating a random matrix A with entries drawn from a uniform distribution between -1 and 1. We then construct the covariance matrix as $\Sigma = AA^\top$. This ensures that

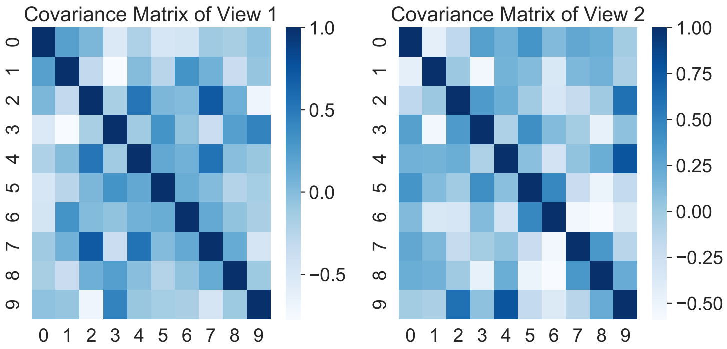


Figure IV.2: Example instances of correlated covariance matrices.

the covariance matrix is positive semi-definite and also tends to produce strong correlations.

We plot an example of the covariance matrices for correlated covariance matrices in both views in figure IV.2.

Recalling table 3.1, note that in the implicit latent variable models, these covariance matrices are precisely the population within-view covariance matrices. In the explicit latent variable models, these covariance matrices are just the covariance matrices of the noise to which we add the signal covariance matrices. Nonetheless, for strong enough noise, this process ensures that there are large correlations between features.

7 Experiment Results

7.1 Exploring the Relationship Between Weights and Loadings in CCA Using Simulated Data

We first present the results of the experiments demonstrating the relationship between weights and loadings in simulated data from explicit and implicit latent variable models with identity and non-identity covariance matrices.

For both cases, we plot the true weights and loadings along with the estimated weights and loadings for each model. We estimate model loadings by multiplying the model weights by the sample within-view covariance matrix following equation IV.3. This means that the estimated model loadings may not be sparse even when the estimated model weights are sparse and the *population* covariance matrix is identity.

We can also quantify the similarity between the true and estimated weights and loadings using the cosine similarity; a measure of the similarity between two vectors that is invariant to the scale of the vectors. The cosine similarity between two vectors is defined as the cosine of the angle between them (Luo et al., 2018). Since we are indifferent to the direction of the vectors, we take the absolute value of the cosine similarity. The absolute cosine similarity between two vectors is 1 if they are identical (up to a sign) and 0 if they are orthogonal.

7.1.1 Implicit Latent Variables (Sparse Weights)

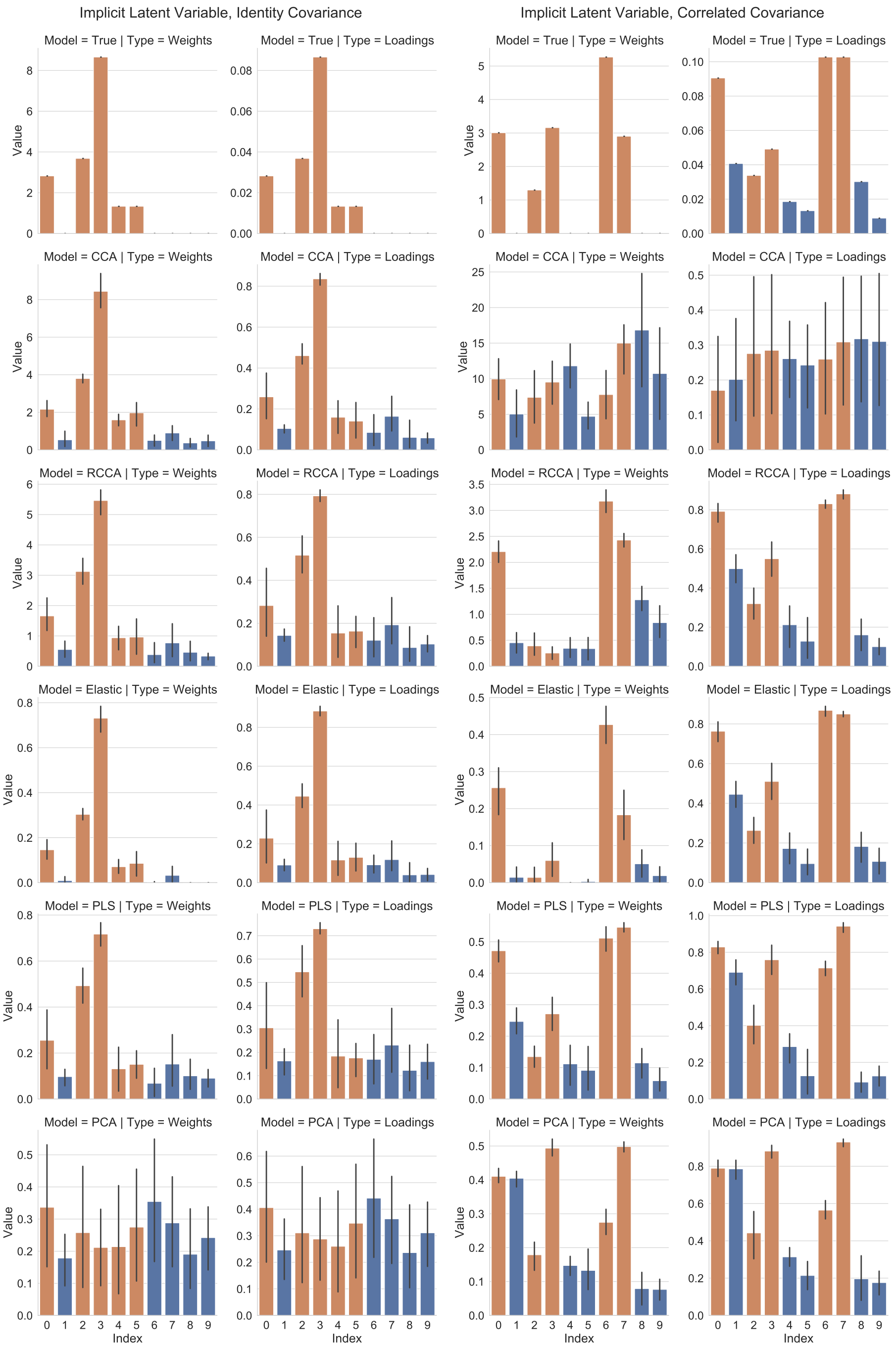
Figure IV.3 shows the true and estimated weights and loadings for data generated from the implicit latent variable models with sparse weights. The Elastic net model exhibits no false negatives (i.e. where the true weight is non-zero but the estimated weight is zero) in both cases. This shows that the Elastic Net CCA model is able to recover the true weights and that the Lasso penalty is indeed inducing sparsity in the weights. The CCA model appears recover spectrum of the true weights much better for the identity covariance matrices than for the correlated covariance matrices. This is likely because the multicollinearity introduced makes the learnt weights substantially less stable with respect to a change in the data.

We plot the cosine similarity between the true and estimated weights and loadings for data generated from the implicit latent variable models with sparse weights in figure IV.4.

Interestingly, we see that for the identity covariance matrices, weight differences are smaller than loading differences. On the other hand for the correlated covariance matrices, the loading differences are smaller than the weight differences. This is evidence of the fact that the weights are not identifiable in the implicit latent variable model as suggested by our theory. Only when the covariance matrices are identity, and when there is only one latent variable, are the weights identifiable.

7.1.2 Explicit Latent Variables

Figure IV.5 shows the true and estimated weights and loadings for data generated from the explicit latent variable models with sparse loadings. The left column shows the results for the identity covariance matrices, while the right column shows the results for the correlated covariance matrices. Once again, the Elastic Net CCA model exhibits no false negatives (i.e. where the true weight is non-zero but the estimated weight is zero) when the noise covariance matrix is identity such that both the weights and loadings are sparse.


Implicit Latent Variable, Identity Covariance

Implicit Latent Variable, Correlated Covariance

Model = True | Type = Weights

Model = True | Type = Loadings

Model = CCA | Type = Weights

Model = CCA | Type = Loadings

Model = RCCA | Type = Weights

Model = RCCA | Type = Loadings

Model = Elastic | Type = Weights

Model = Elastic | Type = Loadings

Model = PLS | Type = Weights

Model = PLS | Type = Loadings

Model = PCA | Type = Weights

Model = PCA | Type = Loadings

Figure IV.3: Bar plots of the true and estimated weights and loadings for the X data generated from the implicit latent variable models with sparse weights. The left column shows the results for the identity covariance matrices, while the right column shows the results for the correlated covariance matrices.

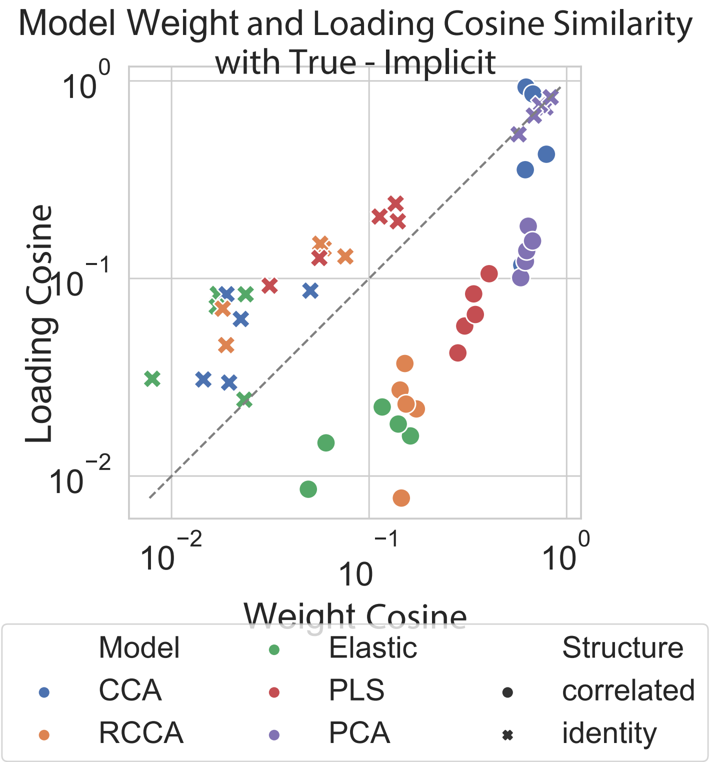


Figure IV.4: Cosine similarity between the true and estimated weights and loadings for data generated from the implicit latent variable models with sparse weights. We plot each run as a point on a scatter plot with a log scale. The grey line indicates where the similarity between weights and loadings are equal.

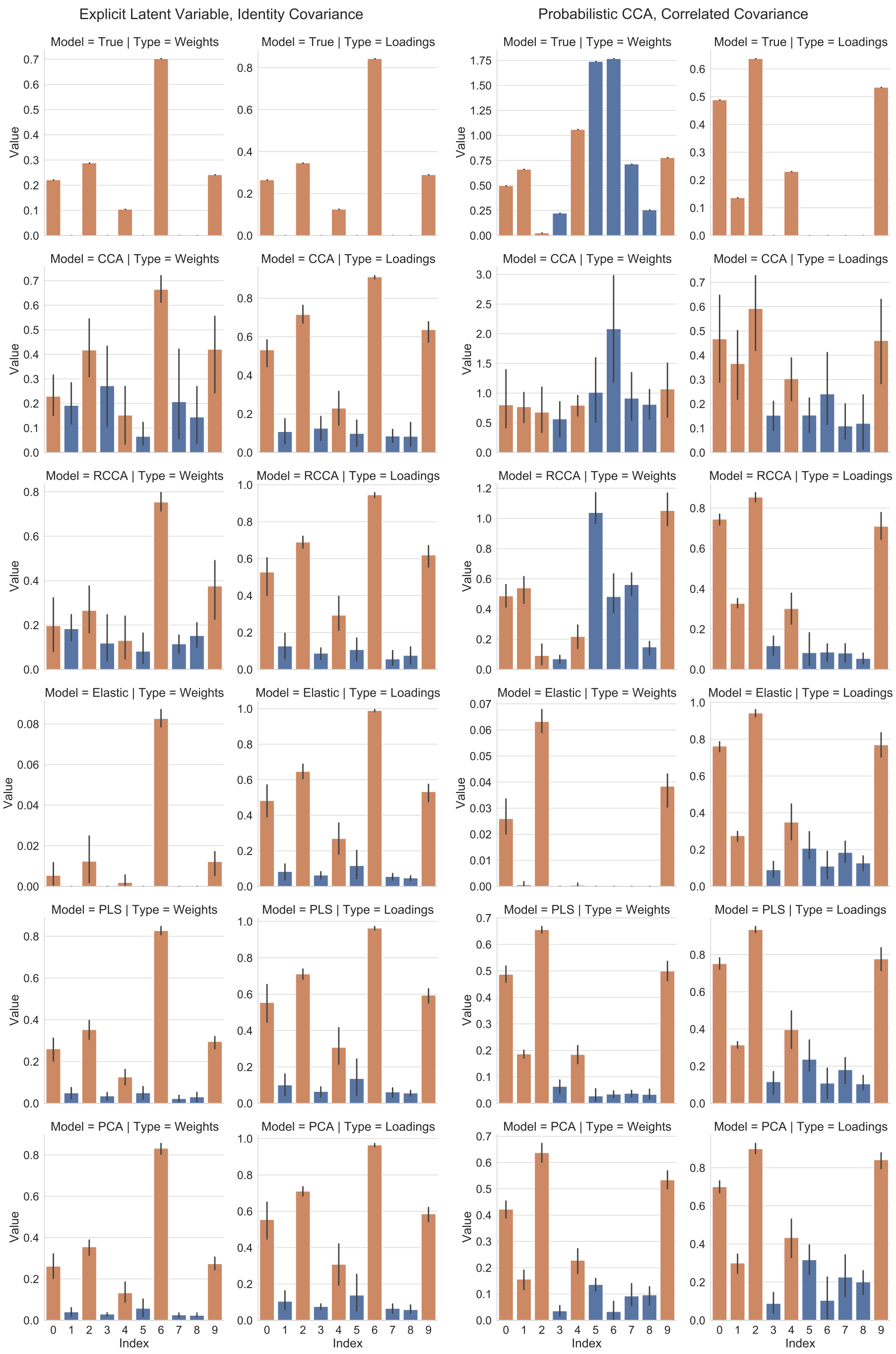


Figure IV.5: Bar plots of the true and estimated weights and loadings for the X data generated from the explicit latent variable models with sparse loadings. The left column shows the results for the identity covariance matrices, while the right column shows the results for the correlated covariance matrices.

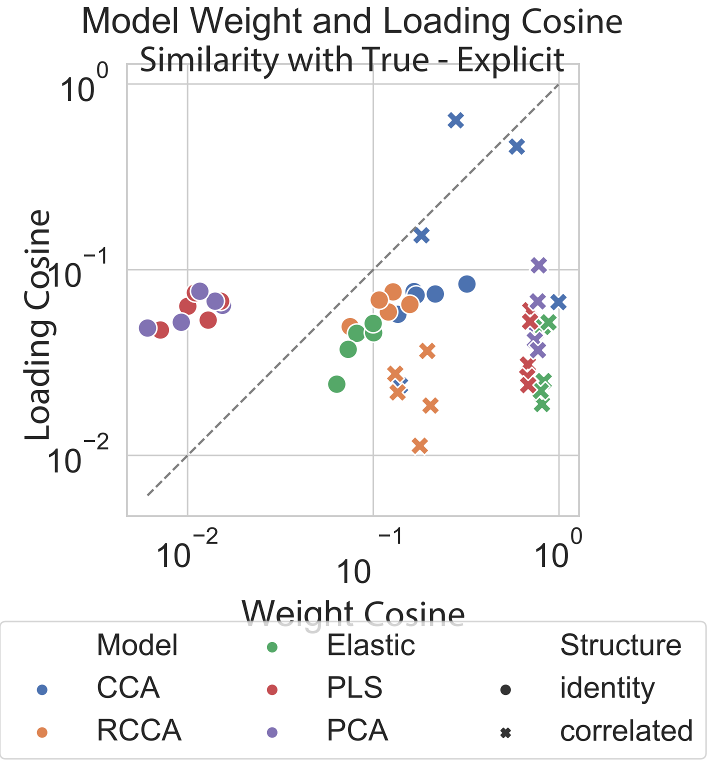


Figure IV.6: Cosine similarity between the true and estimated weights and loadings for data generated from the explicit latent variable models with sparse loadings. We plot each run as a point on a scatter plot with a log scale. The grey line indicates where the similarity between weights and loadings are equal.

Once again, we can quantify the similarity between the true and estimated weights and loadings using the cosine similarity (Figure IV.6).

Notably, when the noise covariance matrix is correlated, the difference in recovery of the weights is much larger than the difference in recovery of the loadings. Suprisingly, when the noise covariance matrix is identity, the PLS and PCA models appear to better recover the weights than the loadings in this case.

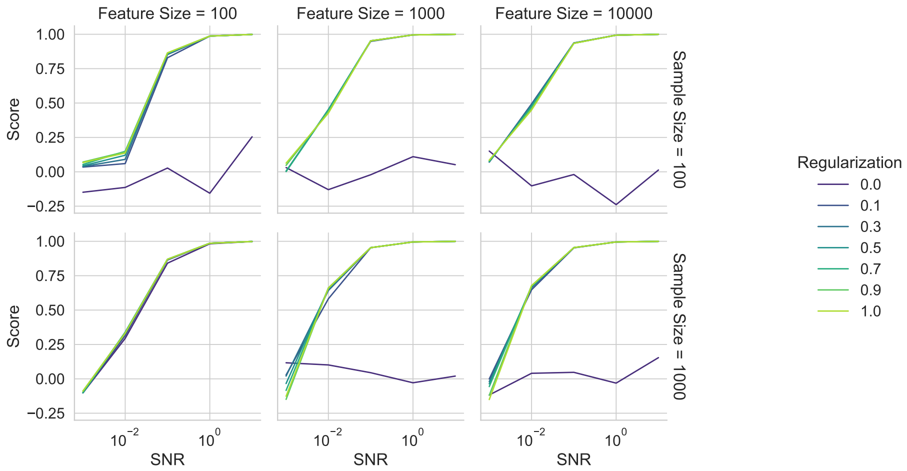


Figure IV.7: Varying signal to noise ratio with identity covariance matrices. We plot the performance of different levels of Regularized CCA from 0 (CCA) to 1 (PLS) for different sample sizes.

7.2 Assessing Information Recovery in CCA and PLS Models Under Varying Signal-to-Noise Ratios

In Figures IV.7 and IV.8 we plot the test correlation (score) varying the signal-to-noise ratio and the number of features under the identity and correlated noise covariance matrices respectively.

In figure IV.7, we can see that the PLS model outperforms all of the Ridge CCA models for all values of the signal-to-noise ratio and dimensionality, though only by a small margin. The unregularized CCA model is much worse than even the Ridge CCA model with the smallest regularization. In this experiment the performance of PLS is directly related to the signal-to-noise ratio.

In figure IV.8, we see a totally different picture. The PLS model is now outperformed by the Ridge CCA model with the smallest regularization. While CCA is still the worst performing model, PLS is now much worse across signal-to-noise ratios and dimensions than any of the Ridge CCA models. This suggests that the PLS model is not able to recover anything like the true signal when the covariance matrices are correlated.

In this experiment it is also clear that the signal-to-noise ratio must be higher to obtain the same performance with higher dimensional data. It is interesting that performance of the Ridge CCA improves across the board with lower regularization.

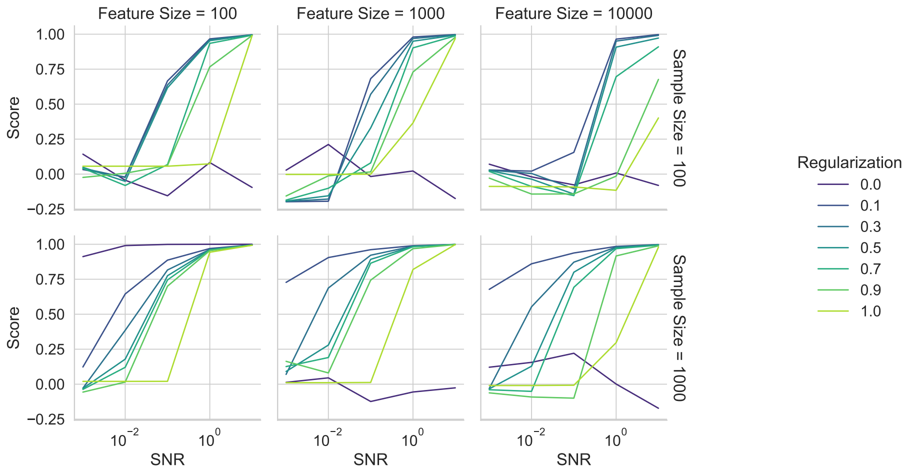


Figure IV.8: Varying signal to noise ratio with correlated covariance matrices. We plot the performance of different levels of Regularized CCA from 0 (CCA) to 1 (PLS) for different sample sizes.

8 Discussion and Limitations

8.1 Revisiting the results from chapter III

In appendix A, we revisit the results from chapter III in the context of the theoretical results from this chapter. While they are not the focus of this chapter, they provide a useful comparison and point of reference for the results in this chapter.

8.2 Future Work

Given our theoretical observations in this chapter, a natural question to ask is whether we can construct a regularization functional that imposes sparsity on the loadings (instead of the weights). The answer is yes, but it is not straightforward and in the small sample setting, it is not clear that it is a good idea. The principle would be much the same as the Lasso, but we would need to use the sample covariance matrix to define the norm:

$$P(W) = \|W\|_1 \tag{IV.33}$$

$$P(L) = \|\hat{\Sigma}U\|_1 \tag{IV.34}$$

Which imposes an L1 penalty on the loadings via an L1 penalty on the weights multiplied by the sample covariance matrix. We could in principle apply the soft-thresholding operator to the estimated loadings. However we would need to be careful to ensure that the sample covariance matrix is invertible in order to get back to the weights. This is of course not guaranteed in the small sample setting.

8.3 Conclusion

In this chapter, we explored the relationship between weights and loadings in CCA models from both theoretical and empirical perspectives. We unified methods for generating simulated multiview data using implicit and explicit latent variable models, providing a framework for understanding the properties of CCA and PLS models.

Through a rigorous mathematical argument, we demonstrated that loadings are invariant to columnwise transformations of the data matrix, while weights are not. This invariance property makes loadings a more reliable choice for interpreting CCA models, as the weights can be arbitrarily set by scaling the data matrix or adding linear combinations of columns.

Our experiments using simulated data provided empirical evidence supporting the theoretical findings. We showed that the recovery of true weights and loadings depends on the underlying covariance structure and the choice of regularization. The results highlighted the importance of considering the signal-to-noise ratio and dimensionality when applying CCA and PLS models to real-world datasets. In neuroimaging-behavior applications, for example, we expect relatively weak signals against substantial measurement noise, suggesting an SNR regime where Ridge CCA may be more appropriate than PLS, particularly if brain regions are known to have correlated noise structure.

Overall, this chapter contributes to a better understanding of the behavior and interpretation of CCA models, providing valuable insights for researchers and practitioners working with multiview data. The findings emphasize the importance of considering the invariance properties of loadings and the impact of covariance structure and regularization on model performance. Future research could explore the extension of these insights to more complex data scenarios and the development of efficient algorithms for imposing sparsity on loadings in CCA models.

Chapter V

Scaling CCA: Stochastic Methods for High-Dimensional Subspace Learning

> It seems easier to train a bi-directional LSTM with attention than to compute the SVD of a large matrix
>
> Chris Ré

(I. Gemp, McWilliams, et al., 2021)

Contents

Preface

The content of this chapter is based on a series of papers (Chapman, Aguila, and Wells, 2022; Chapman, Wells, and Aguila, 2024) as well as a NeurIPS workshop paper (Chapman and Wells, 2023). I am grateful to my co-authors Lennie Wells and Ana Lawry Aguila for their contributions to this work. I conceived the original idea of developing new stochastic algorithms based on the GEP formulation CCA using the GEP formulation, and developed a number of methods that appeared to perform well empirically including writing all of the code and preliminary proofs. Lennie formalised the Eckart-Young-Mirsky theorem-based objective and developed the theoretical results, while Ana provided the processed data from the UK Biobank and ran the UK Biobank experiments. In the interests of communicating my contribution, in this chapter I include the results from the papers but refer the reader to the original papers for Lennie's detailed derivations and proofs.

1 Introduction

Generalized Eigenvalue Problems (GEPs) are fundamental to a wide range of machine learning algorithms, including Canonical Correlation Analysis (CCA), Partial

Least Squares (PLS), Principal Component Analysis (PCA), Independent Component Analysis (ICA), and Linear Discriminant Analysis (LDA). Solving high-dimensional GEPs is a critical challenge in many applications, as classical algorithms often struggle with the computational complexity and numerical instability that arise when dealing with large-scale datasets. This has motivated the development of stochastic algorithms that aim to approximate the solutions of GEPs in a more efficient and scalable manner. Stochastic algorithms for simple Eigenvalue Problems (EPs), where the matrix B in the GEP is the identity matrix, have been extensively studied and have shown promising results (Arora, Cotter, et al., 2012; Arora, Mianjy, and Marinov, 2016). These algorithms not only offer computational advantages but also introduce a form of implicit regularization through the noise in the stochastic updates, which can be particularly beneficial in high-dimensional settings. The regularizing effect of stochastic algorithms has been shown to improve generalization performance and robustness to noise, making them an attractive alternative to batch learning algorithms. However, when it comes to solving more complex GEPs, such as those arising in CCA and PLS, the existing stochastic algorithms have some limitations that hinder their practical applicability. For instance, the γ-EigenGame algorithm (I. M. Gemp et al., 2020; I. Gemp, McWilliams, et al., 2021) requires the tuning of a hyperparameter γ that controls the trade-off between computational efficiency and the accuracy of the stochastic updates. Finding the optimal value of this hyperparameter can be challenging and may vary depending on the problem and the data, making the algorithm less convenient to use in practice. Another limitation is that some stochastic algorithms, like the Stochastic Generalized Hebbian Algorithm (SGHA) (Z. Chen et al., 2019), rely on heuristic primal-dual update rules rather than a principled optimization framework. This makes it difficult to integrate these algorithms with more sophisticated optimizers like Adam (Kingma and Ba, 2014), which have been shown to improve convergence speed and stability in many machine learning applications. To address these limitations and provide a more principled and robust approach to solving GEPs in the stochastic setting, we propose a novel formulation of the CCA problem based on the Eckhart–Young–Mirsky inequality (**stewart_matrix_1990**; **mirsky1960symmetric** ; Eckart and Young, 1936). Our formulation leads to a new objective function that characterizes the top-K subspace of GEPs, including CCA as a special case. Importantly, this objective function is unconstrained and can be optimized using standard stochastic gradient descent or batch gradient descent methods, making it compatible with modern optimization techniques. The key advantages of our proposed approach are:

- It provides a unified framework for solving a wide range of GEPs using a single unconstrained objective function, making it applicable to CCA, PLS, PCA, ICA, LDA, and other problems that can be formulated as GEPs.
- The objective function can be efficiently optimized using stochastic gradient descent or batch gradient descent, allowing for seamless integration with state-of-the-art optimizers like Adam.
- The stochastic nature of the optimization introduces implicit regularization, which can improve the generalization performance and robustness of the learned subspaces.
- The approach is more principled and theoretically grounded compared to heuristic primal-dual update rules used in some existing stochastic algorithms.

The rest of this chapter is organized as follows: Section 2 provides background on GEPs and existing methods for solving them. Section 3 introduces our novel Eckhart-Young inspired objective and the GEP-EY algorithm. Section 4 presents experimental results on various datasets. Finally, Section 5 discusses limitations, future work, and concludes the chapter.

2 Background: Efficient Solutions to GEPs

2.1 Solving High-Dimensional Generalized Eigenvalue Problems

As introduced in Section 3.1, Generalized Eigenvalue Problems play a crucial role in many machine learning algorithms. While the formal definition and properties of GEPs were established earlier, it's important to highlight the challenges that arise when solving these problems in high-dimensional spaces.

Solving GEPs becomes particularly challenging when dealing with high-dimensional data, where the matrices A and B can be very large. The computational complexity of classical algorithms for solving GEPs, such as the power method, scales cubically with the size of the matrices, making them infeasible for large-scale problems. Moreover, when the matrices are ill-conditioned or nearly singular, these algorithms can suffer from numerical instability, leading to inaccurate or unreliable solutions.

2.2 Unified GEP formulation for CCA, Ridge CCA, PLS, and PCA

As discussed in Section 4, CCA, ridge-regularized CCA, PLS, and even PCA can be formulated as GEPs with specific structures. This unified formulation involves block matrices A and B_α defined as:

$$A^{(ij)} = \mathrm{Cov}(X^{(i)}, X^{(j)}) \text{ for } i \neq j, \tag{V.1}$$

$$B_\alpha^{(ii)} = \alpha_i I_{D^{(i)}} + (1-\alpha_i)\mathrm{Var}(X^{(i)}), \tag{V.2}$$

where $\alpha \in [0,1]^I$ is a vector of ridge penalty parameters, $X^{(i)}$ represents the i-th view of the data, and $D^{(i)}$ is the dimensionality of the i-th view.

By adjusting the values of α, we can recover various subspace learning methods:

- Pure CCA: Setting $\alpha_i = 0 : \forall i$ recovers the classic CCA problem.
- Ridge Extensions: Smoothly transitioning to ridge-regularized CCA or PLS by selectively adjusting values within α.
- PCA Subsumption: PCA emerges as a single-view form of ridge-regularized PLS.

This unified formulation allows us to focus on solving the core CCA problem, with the understanding that the insights and solutions developed will naturally generalize to the entire family of methods and GEPs more generally. The challenge, then, is to develop efficient methods for solving these high-dimensional GEPs, which is the focus of our proposed approach.

2.3 Classical Methods for Solving CCA

2.3.1 Reminder of notations

Before discussing classical methods for solving CCA, let's recall some key notation and concepts from our unified GEP formulation. As established in Section 3, we use d to denote the dimensionality of a view. For CCA and related problems like PLS, we typically work with two views, so we have d_1 and d_2. This means that the matrices A and B in the GEP have dimensions $(d_1 + d_2) \times (d_1 + d_2)$. The goal is to find the top K generalized eigenvectors of (A, B), where K is the number of components we want to extract.

2.3.2 Eigendecomposition-based Solution

One common technique to solve the GEP for CCA is to transform it into a standard eigenvalue problem:

$$B^{-\frac{1}{2}} A B^{-\frac{1}{2}} y = \lambda y \tag{V.3}$$

followed by eigendecomposition. This approach is based on the observation that if (u, λ) is a generalized eigenpair of (A, B), then $(B^{-\frac{1}{2}}u, \lambda)$ is an eigenpair of $B^{-\frac{1}{2}}AB^{-\frac{1}{2}}$. An alternative approach is to solve the eigenvalue problem:

$$B^{-1}Av = \lambda v \tag{V.4}$$

However, this approach requires computing the inverse of B, which can be numerically unstable and computationally expensive, especially when B is ill-conditioned or nearly singular. While the $B^{-\frac{1}{2}}AB^{-\frac{1}{2}}$ formulation still requires computing the inverse of the square root of B, this operation can be somewhat more stable when combined with techniques like the Cholesky decomposition, though numerical issues may still arise. Nonetheless, both approaches have a computational complexity of $\mathcal{O}((d_1 + d_2)^3)$ (Golub and Van Loan, 2013), which can be prohibitive for high-dimensional data, and may suffer from numerical instability, especially when B is ill-conditioned or nearly singular.

2.3.3 Classical Iterative Algorithms

Classical iterative algorithms, such as the power method, Sanger's rule (which gives the Generalized Hebbian Algorithm) (Sanger, 1989), and Oja's rule (Oja, 1982), can be used to solve the GEP for CCA. These algorithms are based on the idea of iteratively updating the eigenvector estimates using matrix-vector multiplications. The power method is a simple iterative algorithm for finding the dominant eigenvector of a matrix A. The update rule for the power method is:

$$u \leftarrow \frac{Au}{|Au|} \tag{V.5}$$

where u is the current estimate of the dominant eigenvector. The power method converges to the dominant eigenvector of A under mild conditions. The Generalized Hebbian Algorithm (GHA), is an iterative learning rule for finding the principal components of a data set. It can be written as:

$$U \leftarrow U + \eta \left(AU - U\text{tril}(U^TAU)\right) \tag{V.6}$$

where tril$(\cdot)$ denotes the lower triangular part of a matrix. Oja's rule, a simplified version of Sanger's rule, can be written as:

$$U \leftarrow U + \eta \left(AU - U(U^TAU)\right) \tag{V.7}$$

To maintain orthogonality of the eigenvectors during the iterative updates, Oja's rule can be combined with a QR decomposition step:

$$U \leftarrow U + \eta \left(AU - U(U^T AU) \right) \tag{V.8}$$

$$U \leftarrow \text{qr}(U) \tag{V.9}$$

where qr$(\cdot)$ denotes the QR decomposition, which factorizes a matrix into an orthogonal matrix Q and an upper triangular matrix R. By using only the orthogonal matrix Q, we ensure that the eigenvectors remain orthogonal throughout the iterative process. Alternatively, other orthogonalization techniques such as the Gram-Schmidt process can be employed. These iterative algorithms can be extended to solve the generalized eigenvalue problem $AU = BU\Lambda$, where B is a positive definite matrix. The update rule for this problem is:

$$U \leftarrow U + \eta \left(AU - BU\text{tril}(U^T AU) \right) \tag{V.10}$$

which subsumes the original GHA as a special case when $B = I$. While these classical iterative algorithms can be computationally efficient, especially for sparse or structured matrices, they may suffer from slow convergence and sensitivity to initialization.

2.3.4 PCA-CCA

To reduce the computational complexity of solving the GEP for CCA, the PCA-CCA method first applies PCA to each view of the data separately and then solves the GEP in the reduced spaceMihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al., 2022. This approach has two main steps:

- Apply PCA to each view: Compute the top K_1 and K_2 principal components for each view, with complexity $\mathcal{O}(d_1^3 + d_2^3)$.
- Solve the GEP in the reduced space: Solve the GEP in the reduced space of size $(K_1 + K_2) \times (K_1 + K_2)$, with complexity $\mathcal{O}((K_1 + K_2)^3)$.

The overall complexity of PCA-CCA is thus $\mathcal{O}(d_1^3 + d_2^3 + (K_1 + K_2)^3)$, which can be significantly lower than the direct solution when $K_1 \ll d_1$ and $K_2 \ll d_2$. However, it's important to note that even the PCA step can be computationally expensive for high-dimensional data. In the case where the number of samples n is smaller than the dimensionalities d_1 and d_2, the maximum number of principal components is $K_1 = K_2 = n$. The complexity of PCA in this case is $\mathcal{O}(n^3 + n^3)$, and the overall complexity of PCA-CCA becomes $\mathcal{O}(2n^3 + (2n)^3) = \mathcal{O}(10n^3)$. While this is lower than the direct solution, it can still be prohibitive for large sample sizes.

2.3.5 Kernel CCA

Kernel CCA (KCCA) is another approach that offers computational advantages for high-dimensional data. KCCA maps the original data to a high-dimensional feature space using a kernel function and then performs CCA in that space. The main advantage of KCCA is that its complexity scales with the number of samples n rather than the dimensionalities d_1 and d_2. The optimization problem for KCCA can be written as:

$$\begin{aligned}
&\alpha_{\text{opt}} = \underset{\alpha}{\operatorname{argmax}} \alpha^{(1)} K^{(1)\,T} K^{(2)} \alpha^{(2)} \\
&\text{subject to:} \\
&\alpha^{(1)} K^{(1)\,T} K^{(1)} \alpha^{(1)} = 1 \\
&\alpha^{(2)} K^{(2)\,T} K^{(2)} \alpha^{(2)} = 1
\end{aligned} \tag{V.11}$$

where $\alpha^{(i)}$ are dual variables, $K^{(i)}$ are kernel matrices defined as $K^{(i)} = \phi(X^{(i)})\phi(X^{(i)})^T$, and $\phi(\cdot)$ is a nonlinear mapping function. The complexity of KCCA is $\mathcal{O}(n^3)$, which can be much lower than the direct solution when $d_i > n$. However, KCCA has some significant drawbacks:

- The need to store and manipulate the kernel matrices, which have size $n \times n$. This can be memory-intensive for large sample sizes.
- The requirement to access all training data at test time, which raises concerns about efficiency and scalability.
- The difficulty in interpreting the results in the original feature space, as the learned projections are in the high-dimensional kernel space.

In summary, classical methods for solving CCA offer various trade-offs between computational complexity, numerical stability, and interpretability. However, these methods may still be prohibitively expensive for high-dimensional data, motivating the development of stochastic algorithms that can efficiently approximate the solution of the CCA problem.

2.4 Stochastic Algorithms for CCA

The principle of empirical risk minimization (ERM) forms the basis for many learning algorithms. ERM involves evaluating and optimizing the performance of an algorithm over a known, fixed dataset. This approach is grounded in the law of large numbers: while we cannot know the true risk associated with the underlying data distribution, we can estimate and optimize the algorithm's performance on a known set of training data, referred to as the empirical risk (Vapnik, 1999).

Traditionally, ERM algorithms process the entire dataset in a batch, updating the model parameters based on the full dataset. Stochastic algorithms extend the ERM framework to handle large-scale datasets by processing the data in small batches or even one sample at a time.

Stochastic methods offer several advantages. They allow us to handle large-scale datasets that may not fit in memory, provide implicit regularization through the noise in the stochastic updates, and often lead to better generalization by approximating the population objective rather than overfitting to the empirical risk.

While stochastic optimization has proven highly effective for solving unconstrained objectives in machine learning, its application to Generalized Eigenvalue Problems and Canonical Correlation Analysis presents unique challenges. The primary difficulty lies in handling the data-dependent constraint $U^\top BU = I$, which is essential for ensuring the orthogonality of the learned subspaces. This constraint is particularly challenging to address in the stochastic setting, as B is not directly accessible and must be estimated from random samples, introducing additional uncertainty into the optimization process.

2.4.1 Stochastic Power Method for PLS

Arora, Mianjy, and Marinov (2016) demonstrate that PLS can be approximated by applying a stochastic power method. The stochastic power method is a simple iterative algorithm that updates the estimates of the left and right singular vectors $U^{(1)}$ and $U^{(2)}$ using the following rules:

$$U_t^{(1)} = \mathcal{P}_{\text{orth}} \left(U_{t-1}^{(1)} + \eta_t X_t^{(1)} (X_t^{(2)})^\top U_{t-1}^{(2)} \right),$$
$$U_t^{(2)} = \mathcal{P}_{\text{orth}} \left(U_{t-1}^{(2)} + \eta_t X_t^{(2)} (X_t^{(1)})^\top U_{t-1}^{(1)} \right),$$

where $\mathcal{P}_{\text{orth}}(\cdot)$ represents an orthogonal projection operator that projects a vector or matrix onto the space orthogonal to the current subspace, $X_t^{(1)}$ and $X_t^{(2)}$ are the new data points at time t, and η_t is the learning rate. Intuitively, the update rule for $U_t^{(1)}$ can be understood as follows:

The term $X_t^{(1)}(X_t^{(2)})^\top U_{t-1}^{(2)}$ computes the correlation between the new data point $X_t^{(1)}$ and the current estimate of the right singular vector $U_{t-1}^{(2)}$, weighted by the corresponding $X_t^{(2)}$. This correlation term is then added to the current estimate of the left singular vector $U_{t-1}^{(1)}$, effectively updating it in the direction that maximizes the covariance between the views. The orthogonal projection operator $\mathcal{P}_{\text{orth}}(\cdot)$ ensures that the updated estimate remains orthogonal to the previous estimates, preventing the algorithm from converging to a suboptimal solution.

The update rule for $U_t^{(2)}$ follows a similar logic, with the roles of $X_t^{(1)}$ and $X_t^{(2)}$ reversed.

The stochastic power method has a low computational complexity of $\mathcal{O}(k(d_1 + d_2))$ per iteration, where k is the number of components being estimated. However, it has some limitations:

Convergence is not guaranteed, as the algorithm may oscillate or diverge in some cases. The orthogonal projection step does not extend naturally to the CCA problem, where the constraints involve the matrix B (i.e., $U^\top BU = I$) rather than the identity matrix (i.e., $U^\top U = I$).

Despite these limitations, the stochastic power method provides valuable insights into the design of stochastic algorithms for CCA and serves as a foundation for more advanced methods.

2.4.2 Stochastic Generalized Hebbian Algorithm (SGHA)

The Stochastic Generalized Hebbian Algorithm (SGHA), proposed by Z. Chen et al. (2019), is an extension of the Generalized Hebbian Algorithm (GHA) to the stochastic setting. SGHA aims to find the top-k generalized eigenvectors of a matrix pair (A, B), where A is symmetric and B is symmetric positive definite. SGHA formulates the constrained optimization problem for the top-k subspace as:

$$\min_U -\operatorname{Tr}\left(U^T AU\right) \quad \text{subject to} \quad U^T BU = I \tag{V.12}$$

Using Lagrange multipliers, this constrained problem can be transformed into an unconstrained one:

$$\min_U -\operatorname{Tr}\left(U^T AU\right) + \lambda\left(U^T BU - I\right) \tag{V.13}$$

Differentiating with respect to U and λ and setting the derivatives to zero yields the stationary points:

$$2AU - 2BU\lambda = 0 \quad \text{and} \quad U^TBU - I = 0 \tag{V.14}$$

$$\implies \lambda = U^TAU \tag{V.15}$$

Based on these stationary points, SGHA proposes a primal-dual update rule:

$$U \leftarrow U - \eta\,(AU - BU\lambda) \tag{V.16}$$

$$\lambda \leftarrow \left(U^TAU\right) \tag{V.17}$$

where η is a learning rate. These updates can be combined into a single update rule:

$$U \leftarrow U - \eta\left(AU - BU\left(U^TAU\right)\right) \tag{V.18}$$

Intuitively, the SGHA update rule can be understood as follows:

The term AU computes the correlation between the current estimate of the generalized eigenvectors U and the data matrix A. The term $BU\left(U^TAU\right)$ serves as a correction term that ensures the orthogonality of the eigenvectors with respect to the matrix B. The matrix $\left(U^TAU\right)$ can be interpreted as an estimate of the generalized eigenvalues. The difference between these two terms is then used to update the current estimate of the eigenvectors U in the direction that maximizes the objective function.

SGHA has a computational complexity of $\mathcal{O}(k^2(d_1 + d_2))$ per iteration, which is higher than that of the stochastic power method but still potentially much faster than batch methods. While SGHA is simple to implement, it relies on a heuristic primal-dual update rule rather than a principled optimization framework. This makes it difficult to integrate with more sophisticated optimizers like Adam (Kingma and Ba, 2014), which have been shown to improve convergence speed and stability in many machine learning applications.

2.4.3 γ-EigenGame for CCA

The γ-EigenGame, proposed by I. M. Gemp et al. (2020) and I. Gemp, McWilliams, et al. (2021), is a stochastic algorithm for CCA inspired by the EigenGame algorithm for PCA. The key idea behind the γ-EigenGame is to view the generalized eigenvectors as competing players in a game, where each player tries to maximize its own utility function. In this game-theoretic formulation, the utility function of each player (eigenvector) u_i consists of a reward term and a penalty term:

$$\max_{u_i} \overbrace{\frac{u_i^T A u_i}{u_i^T B u_i}}^{\text{reward}} - \overbrace{\sum_{j<i} \frac{(u_j^T A u_j)(u_i^T B u_j)^2}{(u_j^T B u_j)^2 (u_i^T B u_i)}}^{\text{penalty}} \tag{V.19}$$

The reward term $\frac{u_i^T A u_i}{u_i^T B u_i}$ encourages the eigenvector u_i to align with the direction of maximum correlation between the views, while the penalty term $\sum_{j<i} \frac{(u_j^T A u_j)(u_i^T B u_j)^2}{(u_j^T B u_j)^2 (u_i^T B u_i)}$ discourages u_i from aligning with the directions already captured by the previous eigenvectors u_j for $j < i$. The γ-EigenGame proposes an update rule for each eigenvector u_i based on this utility function. In the stochastic setting, the algorithm introduces an additional hyperparameter γ that controls the trade-off between computational efficiency and the accuracy of the stochastic updates. The update rule is modified to use a rolling average of the matrix B, which helps to reduce the computational complexity and memory requirements of the algorithm. While the γ-EigenGame has shown promising results in practice, it still requires the tuning of both the hyperparameter γ and a learning rate, which can be challenging and may depend on the specific problem and data at hand. Moreover, the use of a rolling average of the matrix B introduces additional approximation error and may not fully capture the uncertainty in the stochastic orthogonality constraint $U^\top B U = I$.

2.4.4 Limitations of Existing Stochastic Algorithms

In this section, we have highlighted some of the key challenges and limitations of existing stochastic algorithms for CCA: the need to tune hyperparameters like γ and learning rates, the reliance on heuristic primal-dual update rules, and the difficulty of integrating with more sophisticated optimizers like Adam.

In the next section, we will introduce a novel stochastic algorithm for CCA based on the Eckart-Young-Mirsky theorem, which aims to address some of the limitations of existing methods and provide a more principled and robust approach to solving high-dimensional CCA problems in the presence of stochastic constraints.

3 Methods: GEP-EY, An Efficient Algorithm for Generalized Eigenvalue Problems

This section introduces GEP-EY, a novel algorithm for solving Generalized Eigenvalue Problems, with a particular focus on Canonical Correlation Analysis and Partial Least Squares. The algorithm stems from a new perspective on GEPs: optimizing an Eckart-Young inspired objective that characterizes the GEP solution.

3.1 A New Perspective: Eckhart-Young Inspired Objective for GEPs

3.1.1 Formulation and Intuition

At the core of our approach is the reformulation of GEPs as an unconstrained optimization problem. This reformulation is inspired by the Eckhart-Young-Mirsky theorem, which is fundamental in matrix approximation theory.

The Eckhart-Young-Mirsky theorem states that the best rank-K approximation of a matrix M in the Frobenius norm is given by its truncated singular value decomposition (SVD). Specifically, if $M = U\Sigma V^\top$ is the SVD of M, then the optimal rank-K approximation is $M_K = U_K \Sigma_K V_K^\top$, where U_K, Σ_K, and V_K are the matrices containing the top K singular vectors and singular values.

We apply a similar approach to the generalized eigenvalue problem (GEP) defined by matrices A and B. By considering the eigen-decomposition of $B^{-\frac{1}{2}}AB^{-\frac{1}{2}}$, we can derive a new objective function for characterizing the top-K subspace of the GEP.

Proposition 3.1 (Eckhart-Young inspired objective for GEPs)**.** *The top-K subspace of the GEP (A, B) can be characterized by minimizing:*

$$\mathcal{L}_{\textit{EY-GEP}}(U) := \operatorname{trace}\left(-2U^\top AU + \left(U^\top BU\right)\left(U^\top BU\right)\right) \tag{V.20}$$

over $U \in \mathbb{R}^{D\times K}$, with a minimum value of $-\sum_{k=1}^{K}\lambda_k^2$, where (λ_k) are the generalized eigenvalues.

This objective intuitively balances maximizing between-view correlations and minimizing within-view variances.

Sketch of the Proof: To provide some intuition, we sketch the key steps in the proof, with full details available in our supplementary material.

1. **Matrix Approximation and SVD:** Begin with the Eckhart-Young-Mirsky theorem, which gives the best rank-K approximation in the Frobenius norm for any matrix using its SVD. The theorem states that for a matrix $M \in \mathbb{R}^{d\times d}$, the best rank-$K$ approximation M_K is obtained by truncating the SVD of M.

2. **Transformation of GEP:** For the GEP defined by matrices A and B, consider the transformation $M = B^{-\frac{1}{2}}AB^{-\frac{1}{2}}$. The eigenvalues and eigenvectors of M are related to the generalized eigenvalues and eigenvectors of (A, B). This transformation simplifies the problem, as M is symmetric.

3. **Objective Reformulation:** By focusing on the top-K eigenvalues and eigenvectors, we can reformulate the problem as an unconstrained optimization problem. Specifically, consider the objective function:

   $$\|M - \tilde{W}\tilde{W}^T\|_F^2.$$

   Since M is symmetric, the optimal $\tilde{W}$ is of the form $W = W_K\Lambda_K^{\frac{1}{2}}O_K$, where W_K contains the top-K eigenvectors, Λ_K contains the corresponding eigenvalues on the diagonal, and O_K is an orthogonal matrix.

4. **Transformation under reparameterisation:** Consider how this expression transforms under the reparameterisation $\tilde{W} = B^{\frac{1}{2}}\tilde{U}$. We get:

   $$\|M - \tilde{Z}\tilde{Z}^T\|_F^2 = \|B^{-\frac{1}{2}}AB^{-\frac{1}{2}} - B^{\frac{1}{2}}\tilde{U}\tilde{U}^T B^{\frac{1}{2}}\|_F^2.$$

5. **Simplifying the Objective:** Simplifying the expression, we use the properties of the Frobenius norm and the cyclic property of the trace:

$$\begin{aligned}\|B^{-\frac{1}{2}}AB^{-\frac{1}{2}} - B^{\frac{1}{2}}\tilde{U}\tilde{U}^T B^{\frac{1}{2}}\|_F^2 &= \|B^{-\frac{1}{2}}AB^{-\frac{1}{2}}\|_F^2 - 2\,\mathrm{trace}(B^{-\frac{1}{2}}AB^{-\frac{1}{2}}B^{\frac{1}{2}}\tilde{U}\tilde{U}^T B^{\frac{1}{2}}) \\ &\quad + \mathrm{trace}((B^{\frac{1}{2}}\tilde{U}\tilde{U}^T B^{\frac{1}{2}})^2) \\ &= \|B^{-\frac{1}{2}}AB^{-\frac{1}{2}}\|_F^2 - 2\,\mathrm{trace}(\tilde{U}^T A\tilde{U}) + \mathrm{trace}((\tilde{U}^T B\tilde{U})^2).\end{aligned}$$

6. **Final Reformulation:** This leads to the final form of the objective function:

   $$\mathcal{L}_{\text{EY-GEP}}(U) := \mathrm{trace}\left(-2U^\top AU + \left(U^\top BU\right)\left(U^\top BU\right)\right),$$

For the full proof and additional technical details, we refer the reader to the supplementary material in Chapman, Wells, and Aguila (2024).

This new perspective provides a more intuitive and computationally efficient way to approach GEPs, leveraging the well-established principles of the Eckhart-Young-Mirsky theorem.

3.2 GEP-EY: A Stochastic Algorithm for Generalized Eigenvalue Problems

We now introduce GEP-EY, a stochastic algorithm designed to solve Generalized Eigenvalue Problems efficiently. While applicable to a wide range of GEPs, we focus particularly on its application to Canonical Correlation Analysis, Partial Least Squares, and related methods.

3.2.1 Batch GEP-EY Algorithm

We first present the batch version of the GEP-EY algorithm in Algorithm 1. This algorithm optimizes the Eckart-Young inspired objective for GEPs using full-batch gradient descent. While it provides an unbiased estimate of the gradient, it may be computationally expensive for large datasets.

Algorithm 1: GEP-EY: Batch algorithm for General GEPs

```
Input: Data $\mathbf{X}$, learning rate $(\eta_t)_t$, iterations $T$
Initialize: $U \in \mathbb{R}^{D \times K}$ randomly
for $t = 1$ to $T$ do
  Construct $A, B$ from $\mathbf{X}$
  Compute loss $\mathcal{L}(U) = \mathrm{trace}(-2U^\top AU + (U^\top BU)(U^\top BU))$
  Update $U \leftarrow U - \eta_t \nabla_U \mathcal{L}(U)$
end for
```

Listing 1 provides PyTorch-style pseudocode for implementing the batch GEP-EY algorithm. The `compute_batch_loss` function directly implements the loss computation from Algorithm 1, while the `batch_gep_ey` function implements the main optimization loop.

3.2.2 Challenges in Stochastic Estimation

A naive approach to stochastic optimization of the GEP-EY objective would be to simply replace A and B with mini-batch estimates $\hat{A}$ and $\hat{B}$. However, this leads to biased gradient estimates due to the quadratic term $(U^\top BU)(U^\top BU)$.

```
def compute_batch_loss(U, A, B):
    UAU = torch.matmul(torch.matmul(U.T, A), U)
    UBU = torch.matmul(torch.matmul(U.T, B), U)
    loss = -2 * torch.trace(UAU) + torch.trace(torch.matmul(UBU, UBU))
    return loss

def batch_gep_ey(X, lr, T):
    U = torch.randn(X.shape[1], K)
    A, B = compute_A_B(X)  # Compute A and B matrices from data
    optimizer = torch.optim.SGD([U], lr=lr)

    for t in range(T):
        optimizer.zero_grad()
        loss = compute_batch_loss(U, A, B)
        loss.backward()
        optimizer.step()

    return U
```

Listing 1: PyTorch-style pseudocode for the batch GEP-EY algorithm

The key issue lies in the fact that the expectation of the product of random variables is not equal to the product of their expectations unless the variables are independent. In our case:

$$\mathbb{E}[(U^\top \hat{B} U)(U^\top \hat{B} U)] \neq \mathbb{E}[U^\top \hat{B} U]\mathbb{E}[U^\top \hat{B} U] \tag{V.21}$$

This inequality arises because both instances of $\hat{B}$ in the quadratic term are estimated from the same mini-batch, introducing correlation between the factors.

To obtain unbiased gradients, we require independent samples for each instance of $U^\top BU$ in the quadratic term. Specifically, we need:

$$\mathbb{E}[(U^\top \hat{B}_1 U)(U^\top \hat{B}_2 U)] = \mathbb{E}[U^\top \hat{B}_1 U]\mathbb{E}[U^\top \hat{B}_2 U] = (U^\top BU)^2 \tag{V.22}$$

where $\hat{B}_1$ and $\hat{B}_2$ are independent estimates of B from different mini-batches.

3.2.3 Dimensionality Reduction for CCA and PLS

For CCA-type problems, owing to the structure of the covariance matrices, we can significantly reduce the computational complexity by exploiting the structure of A and B. Instead of working with the full $D \times D$ matrices, we introduce smaller matrices $C(\theta)$ and $V_\alpha(\theta)$:

$$C(\theta) = \sum_{i \neq j} \mathrm{Cov}(Z^{(i)}, Z^{(j)}) = U^\top A U, \tag{V.23}$$

$$V_\alpha(\theta) = \sum_{i} \alpha_i U^{(i)\top} U^{(i)} + (1 - \alpha_i)\mathrm{Var}(Z^{(i)}) = U^\top B U, \tag{V.24}$$

where $Z^{(i)} = f^{(i)}(X^{(i)}; \theta^{(i)})$ are learned representationss of dimension $K \ll D_i$, and α_i are ridge penalty parameters that allow us to interpolate between different methods in the CCA family. This leads to our reduced Eckart-Young inspired objective:

$$\mathcal{L}_{\mathsf{EY}}(\theta) = -2 \operatorname{trace} C(\theta) + |V_\alpha(\theta)|_F^2. \tag{V.25}$$

To illustrate how $C(\theta)$ and $V_\alpha(\theta)$ relate to $U^\top A U$ and $U^\top B U$ respectively for various methods in the CCA family, we present Table 3.1. The key advantage here is that

Table 3.1: Covariance matrices A, B and their low-rank equivalents $C(\theta)$, $V_\alpha(\theta)$ for different methods in the CCA family. The α_i values determine the ridge penalty for each method.

Method	A	B	$C(\theta)$	$V_\alpha(\theta)$	α_i
CCA	$\begin{pmatrix} 0 & \Sigma_{12} \\ \Sigma_{21} & 0 \end{pmatrix}$	$\begin{pmatrix} \Sigma_{11} & 0 \\ 0 & \Sigma_{22} \end{pmatrix}$	$\sum_{i \neq j} \mathrm{Cov}(Z^{(i)}, Z^{(j)})$	$\sum_i \mathrm{Var}(Z^{(i)})$	0
MCCA	$\begin{pmatrix} 0 & \Sigma_{12} & \cdots \\ \Sigma_{21} & 0 & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$	$\begin{pmatrix} \Sigma_{11} & 0 & \cdots \\ 0 & \Sigma_{22} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$	$\sum_{i \neq j} \mathrm{Cov}(Z^{(i)}, Z^{(j)})$	$\sum_i \mathrm{Var}(Z^{(i)})$	0
PLS	$\begin{pmatrix} 0 & \Sigma_{12} \\ \Sigma_{21} & 0 \end{pmatrix}$	I	$\sum_{i \neq j} \mathrm{Cov}(Z^{(i)}, Z^{(j)})$	$\sum_i U^{(i)\top} U^{(i)}$	1
Ridge CCA	$\begin{pmatrix} 0 & \Sigma_{12} \\ \Sigma_{21} & 0 \end{pmatrix}$	$\begin{pmatrix} \Sigma_{11} + \alpha_1 I & 0 \\ 0 & \Sigma_{22} + \alpha_2 I \end{pmatrix}$	$\sum_{i \neq j} \mathrm{Cov}(Z^{(i)}, Z^{(j)})$	$\sum_i \alpha_i U^{(i)\top} U^{(i)} + (1 - \alpha_i)\mathrm{Var}(Z^{(i)})$	$0 < \alpha_i < 1$
PCA	Σ	I	$\mathrm{Cov}(Z, Z)$	$U^\top U$	1

we only need to operate on and store $K \times K$ matrices, which is much more efficient when $K \ll D_i$ for all i. This dimensionality reduction is crucial for the efficiency of our algorithm, as demonstrated in Algorithm 2 and the corresponding PyTorch-style pseudocode in Listing 2. By working with $C(\theta)$ and $V_\alpha(\theta)$ instead of the full A and B matrices, we significantly reduce the computational complexity while still capturing the essential structure of the problem. The ridge penalty parameter α_i allows us to smoothly interpolate between different methods in the CCA family:

For CCA and MCCA, $\alpha_i = 0$ for all i, focusing entirely on the variance of the learned representations. For PLS and PCA, $\alpha_i = 1$ for all i, focusing on the magnitude of the projection vectors. For Ridge CCA, $0 < \alpha_i < 1$, balancing between the variance of the representations and the magnitude of the projection vectors.

This unified formulation allows our GEP-EY algorithm to efficiently handle a wide range of multi-view learning problems within a single framework.

3.2.4 Unbiased Stochastic Estimation for CCA

By combining our efficient low-dimensional approach with independent minibatch sampling, we can develop a highly efficient and unbiased method for optimizing stochastic CCA and related problems.

Recall that we introduced the low-rank matrices $C(\theta)$ and $V_\alpha(\theta)$ to reduce computational complexity:

$$C(\theta) = \sum_{i \neq j} \mathrm{Cov}(Z^{(i)}, Z^{(j)}) = U^\top A U, \tag{V.26}$$

$$V_\alpha(\theta) = \sum_{i} \alpha_i U^{(i)\top} U^{(i)} + (1 - \alpha_i) \mathrm{Var}(Z^{(i)}) = U^\top B U, \tag{V.27}$$

In practice, we work with sample estimates rather than true population covariances. We construct unbiased estimates of $C(\theta)$ and $V_\alpha(\theta)$ from mini-batches:

$$\hat{C}(\theta)[\mathbf{Z}] = \sum_{i \neq j} \widehat{\mathrm{Cov}}(\mathbf{Z}^{(i)}, \mathbf{Z}^{(j)}), \quad \hat{V}_\alpha(\theta)[\mathbf{Z}] = \sum_{i} \alpha_i U^{(i)\top} U^{(i)} + (1 - \alpha_i) \widehat{\mathrm{Var}}(\mathbf{Z}^{(i)}). \tag{V.28}$$

To address the bias issue in the quadratic term, we use two independent minibatches $\mathbf{Z}$ and $\mathbf{Z}'$. This allows us to define the following unbiased loss estimate:

$$\hat{\mathcal{L}}_{\mathsf{EY}}[\mathbf{Z}, \mathbf{Z}'] := -2 \operatorname{trace} \hat{C}[\mathbf{Z}] + \langle \hat{V}_\alpha[\mathbf{Z}], \hat{V}_\alpha[\mathbf{Z}'] \rangle_F, \tag{V.29}$$

where $\langle \cdot, \cdot \rangle_F$ denotes the Frobenius inner product.

This formulation combines the benefits of our low-rank approach (computational efficiency) with independent minibatch sampling (unbiased estimation). It allows us to optimize the population problem using only sample data, making it particularly well-suited for large-scale CCA and related problems. By using this approach, we can perform stochastic optimization of CCA that is both computationally efficient and statistically unbiased, bridging the gap between theoretical analysis and practical application in high-dimensional settings.

3.2.5 The GEP-EY Algorithm for CCA and PLS

Incorporating these dimensionality reduction and unbiased estimation techniques, we present the specialized GEP-EY algorithm for CCA, PLS, and related methods in Algorithm 2.

```
Algorithm 2: GEP-EY: Stochastic algorithm for CCA and PLS
    Input: Data stream $(\mathbf{X}(b))_{b=1}^{\infty}$, learning rate $(\eta_t)_t$, iterations $T$, function class $f(\cdot; \theta)$
    Initialize: $\hat{\theta}$ randomly
    for $t = 1$ to $T$ do
      Sample mini-batches $\mathbf{X}(b), \mathbf{X}(b')$ independently
      Compute $\mathbf{Z}(b) = f(\mathbf{X}(b); \theta), \mathbf{Z}(b') = f(\mathbf{X}(b'); \theta)$
      Estimate loss $\hat{\mathcal{L}}_{\mathsf{EY}}(\theta)$ using Eq. equation V.29
      Update $\theta$ using gradient descent
    end for
```

This algorithm efficiently solves GEPs for CCA, PLS, and related methods by leveraging the low-rank structure of the problem and unbiased stochastic estimation. The weightss U are implicitly learned through the optimization of θ, and the resulting representationss Z can be used for downstream tasks or analysis.

The algorithm can be implemented efficiently using automatic differentiation frameworks like PyTorch. Listing 2 provides PyTorch-style pseudocode for the unbiased stochastic GEP-EY algorithm. The `compute_unbiased_stochastic_loss` function implements the loss computation from Equation equation V.29, while the `unbiased_stochastic_gep_ey` function implements the main loop of Algorithm 2.

This implementation ensures unbiased estimation even with small batch sizes, making it suitable for large-scale problems. By using two independent mini-batches for each iteration (as seen in the `unbiased_stochastic_gep_ey` function), we avoid the bias issues discussed earlier while maintaining computational efficiency.

This formulation ensures unbiased estimation even with small batch sizes, making it suitable for large-scale problems. The algorithm can be implemented efficiently using automatic differentiation frameworks like PyTorch:

This algorithm efficiently solves GEPs for CCA, PLS, and related methods by leveraging the low-rank structure of the problem and unbiased stochastic estimation. The weightss U are implicitly learned through the optimization of θ, and the resulting representationss Z can be used for downstream tasks or analysis.

In summary, GEP-EY provides a novel, efficient, and theoretically grounded approach to solving GEPs, particularly for CCA and PLS problems, by leveraging a new Eckhart-Young inspired objective and stochastic optimization techniques.

4 Experiments and Results

4.1 Comparison with Standard Batch Solvers

In this experiment, we compare our CCA-EY method in the full batch setting to the traditional approach of solving the CCA Generalized Eigenvalue Problem (GEP) using the `scipy.linalg.eigh` function (Virtanen et al., 2020). This comparison aims to demonstrate the scalability advantages of our method, particularly for high-dimensional data.

4.1.1 Data Generation and Experimental Design

Our experimental framework employs carefully constructed synthetic datasets, designed to challenge both methods across a spectrum of dimensionalities while maintaining consistent statistical properties. The key characteristics of our data generation process are as follows:

We generate multivariate Gaussian datasets with a prescribed covariance structure, ensuring the existence of a true CCA solution. The dimensionality of these datasets ranges from low to high, specifically from 10 to 5,000 features, allowing us to assess performance across a broad spectrum of problem sizes. To isolate the effect of dimensionality, we maintain a constant sample size of 10,000 instances across all datasets.

To ensure robust results and account for statistical variations, we repeat the data generation and evaluation process five times for each feature size. This repetition allows us to compute average performance metrics and their associated variances, providing a more reliable basis for comparison.

In our analysis, we focus on the top-5 components of the CCA subspace. This choice reflects a common practical scenario where the primary interest lies in the most significant correlations between the two views of the data.

4.1.2 Evaluation Methodology

Our evaluation methodology is designed to provide a fair and informative comparison between the traditional GEP solver and our CCA-EY method. For each generated dataset:

- We first apply the `scipy.linalg.eigh` function to solve the GEP, recording the computation time required to obtain the CCA solution.
- We then train our CCA-EY method on the same dataset. To ensure a fair comparison with `scipy.linalg.eigh`, we allow CCA-EY to iterate until convergence, defined as the point where the Frobenius norm of the weight change between iterations falls below a threshold of 10^{-5}. This criterion ensures that CCA-EY achieves a solution quality comparable to the batch method.
- We record the computation time for both methods, providing a direct comparison of their efficiency across different data dimensionalities.

4.1.3 Results and Observations

Figure V.1 shows the results of this experiment. For feature sizes up to 1,000, our CCA-EY method is slower than `scipy.linalg.eigh`. However, beyond this point, CCA-EY significantly outperforms the baseline in terms of computation time. As the number of features increases, the time taken to solve the GEP using `scipy.linalg.eigh` scales quadratically, while our CCA-EY method exhibits a linear scaling. This is because CCA-EY scales linearly with both the number of samples and the number of features, whereas `scipy.linalg.eigh` has a quadratic dependence on the feature size.[1].

[1] When the number of features is greater than the number of samples as in ridge CCA and PLS, we can use the PCA or Kernel trikcs to scale CCA quadratically with the minimum of the number of features and samples, though this still requires a somewhat expensive PCA

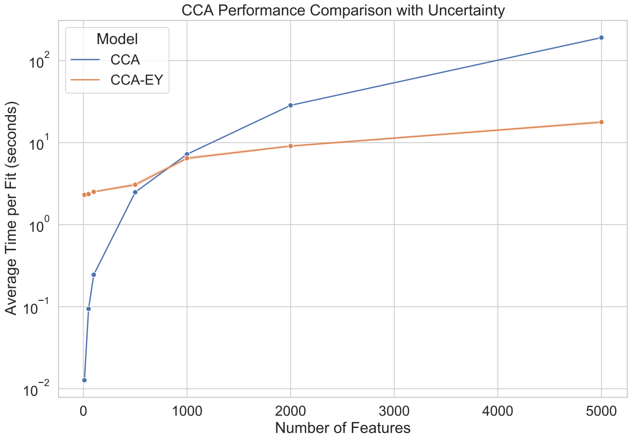


Figure V.1: Comparison of the time taken to solve CCA using `eigh` and our CCA-EY method.

4.2 Stochastic CCA on Real-World Datasets

This experiment evaluates our CCA-EY method against two established baselines, γ-EigenGame (I. Gemp, C. Chen, and McWilliams, 2022) and SGHA (Z. Chen et al., 2019), on real-world datasets. We aim to assess the algorithms' ability to handle high-dimensional data efficiently and accurately without prior dimensionality reduction.

4.2.1 Datasets

We evaluate our CCA-EY method on two diverse datasets: MediaMill and Split-CIFAR.

The MediaMill dataset (Snoek et al., 2005) offers a multimodal challenge in the domain of video analysis. It comprises 25,800 test instances, each represented by two views: 120-dimensional visual features extracted from video frames and 101-dimensional textual features derived from accompanying commentary. This dataset tests the model's ability to learn correlated representations across different modalities, bridging the gap between visual and textual information in video content.

The Split-CIFAR dataset (Z. Meng, Chakraborty, and Singh, 2021) presents a novel approach to evaluating CCA methods using image data. Derived from the CIFAR-10 dataset, it splits each image into two halves, creating two distinct views. The dataset consists of 50,000 training instances and 10,000 test instances. Each view (half-image) is represented by features of dimension 32x16x3. This splitting mechanism challenges the model to learn correlated representations from partial image information, effectively reconstructing the original image content. The high dimensionality and inherent correlations between image halves make Split-CIFAR an excellent testbed for assessing the efficiency and accuracy of CCA methods in processing visual data.

4.2.2 Experimental Setup

Our experimental framework is designed to rigorously evaluate the performance of our proposed method, CCA-EY, against two established baselines: γ-EigenGame and SGHA. We have carefully crafted this setup to align closely with the methodologies employed by Z. Meng, Chakraborty, and Singh (2021) and I. Gemp, C. Chen, and McWilliams (2022), ensuring comparability with existing literature while introducing key innovations.

At the core of our experiment is a direct comparison between CCA-EY and the aforementioned baselines, allowing us to assess the relative strengths and potential advantages of our approach in the context of stochastic CCA. Notably, our methodology diverges from that of I. Gemp, C. Chen, and McWilliams (2022) in one crucial aspect: we eschew the simplification of performing Principal Component Analysis (PCA) on the data before applying the CCA methods. This decision to work with raw, high-dimensional data presents a more challenging and realistic scenario, which may account for any observed decrease in the performance of γ-EigenGame compared to their reported results.

To evaluate the efficiency and rapid learning capabilities of each method, we constrain the training process to a single epoch. This design choice challenges the algorithms to extract meaningful canonical correlations from limited exposure to the data. Within this single-epoch framework, we explore the impact of stochasticity by implementing a range of mini-batch sizes, specifically from 5 to 100. This spectrum allows us to observe how each method performs under varying degrees of sample randomness, from highly stochastic scenarios with small batches to more stable conditions with larger batches.

Table 4.1 details the hyperparameter ranges explored for each method, replicating those in prior work for consistency and comparability.

4.2.3 Evaluation Metric: Proportion of Correlation Captured (PCC)

We employ the Proportion of Correlation Captured (PCC) metric for evaluation:

Table 4.1: Hyperparameter ranges for CCA methods

Parameter	Values
Minibatch size	5, 20, 50, 100
Components	5
Epochs	1
Seeds	1, 2, 3, 4, 5
Learning rate	0.01, 0.001, 0.0001
γ^2	0.01, 0.1, 1, 10

$$\text{PCC} = \frac{\sum_{k=1}^{K} \rho_k}{\sum_{k=1}^{K} \rho_k^*}$$

where ρ_k are the empirical correlations between the estimated representations $Z^{(i)} = X^{(i)}\hat{U}^{(i)}$ on the test set, and ρ_k^* are the canonical correlations computed from full batch covariance matrices. K denotes the number of canonical components considered.

The PCC metric offers several advantages for evaluating CCA methods. By comparing the correlations of the estimated representations to the true canonical correlations, PCC provides a normalized measure of how well a method captures the underlying correlations in the data. This normalization allows for fair comparisons across different datasets and experimental settings.

Interpreting PCC is intuitive: a value closer to 1 indicates that the method captures a higher proportion of the true correlations, with 1 representing perfect correlation capture. This directness enables easy comparison between different algorithms and across various experimental conditions.

Moreover, PCC efficiently tracks algorithm performance over time while minimizing computational overhead. This makes it particularly suitable for evaluating online or iterative CCA methods, where performance may evolve during the learning process.

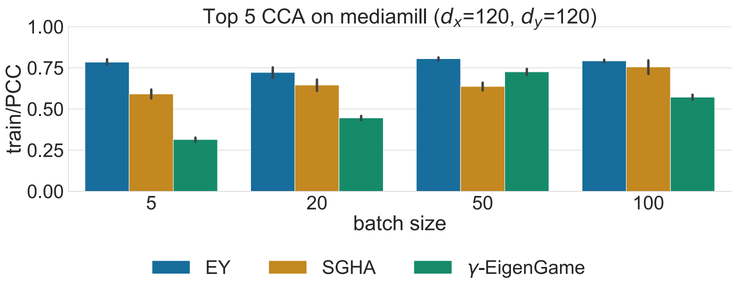


Figure V.2: Stochastic CCA on MediaMill using PCC: Performance across varying mini-batch sizes. Shaded regions represent $\pm$ one standard deviation around the mean of 5 runs.

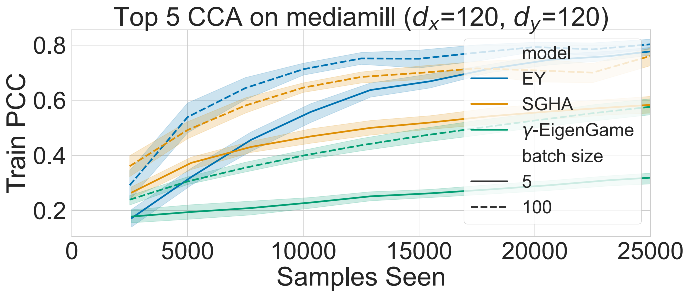


Figure V.3: Stochastic CCA on MediaMill: Training progress over a single epoch for mini-batch sizes 5 and 100.

The widespread use of PCC in the literature (Z. Meng, Chakraborty, and Singh, 2021; I. Gemp, C. Chen, and McWilliams, 2022; Ma, Lu, and Foster, 2015; Ge et al., 2016) also facilitates comparisons with existing work, providing a standardized benchmark for assessing the performance of our CCA-EY method against established baselines.

4.2.4 Results and Observations

Figure V.2 compares the learning curves of the algorithms on the MediaMill dataset for various mini-batch sizes. CCA-EY consistently outperforms the baselines across all batch sizes. Figure V.3 provides a more detailed view of the learning curves for batch sizes 5 and 100, highlighting the superior performance of CCA-EY over time. For the Split-CIFAR dataset, Figure V.4 shows the performance comparison across batch sizes, while Figure V.5 presents the learning curves. The results reveal that γ-

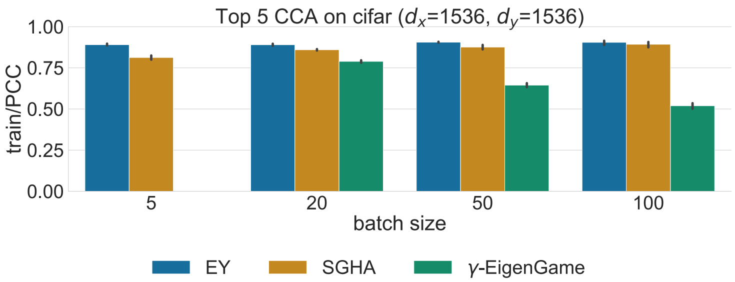


Figure V.4: Stochastic CCA on Split-CIFAR using PCC: Performance across varying mini-batch sizes. Shaded regions represent ± one standard deviation around the mean of 5 runs.

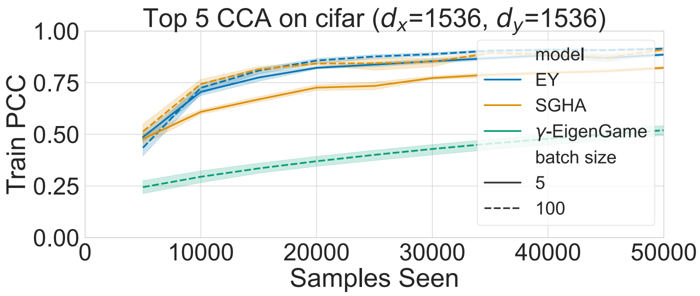


Figure V.5: Stochastic CCA on Split-CIFAR: Training progress over a single epoch for mini-batch sizes 5 and 100.

EigenGame underperforms compared to CCA-EY and SGHA, particularly for smaller batch sizes. These experiments demonstrate that our proposed CCA-EY method can achieve faster convergence with less hyperparameter tuning compared to the established baselines, making it a promising approach for practical applications involving high-dimensional data.

4.3 Stochastic PLS on UK Biobank Data

In this experiment, we showcase the scalability and efficiency of our Stochastic PLS method, PLS-EY, on an extremely high-dimensional imaging genetics dataset from the UK Biobank (Sudlow et al., 2015). The primary aim of this experiment is to demonstrate computational feasibility rather than to establish generalizable biological findings.

4.3.1 Dataset

The UK Biobank is a large-scale biomedical database containing genetic and phenotypic data from over 500,000 participants. For this experiment, we use a subset of the data consisting of brain imaging features (82 regional volumes) and genetic variants (582,565 SNPs) for 33,333 subjects.

The brain imaging data was preprocessed using FreeSurfer (Fischl, 2012) to extract gray-matter volumes for 66 cortical regions (based on the Desikan-Killiany atlas) and 16 subcortical regions. To isolate disease-relevant signals in brain structure, we regressed out several key confounding variables from the brain features:

- Age and age squared: to account for both linear and nonlinear effects of aging on brain volume, as brain structure naturally changes across the lifespan (Fjell et al., 2009)
- Intracranial volume: to control for differences in overall head size, which can mask or confound regional volume variations (Voevodskaya et al., 2014)
- Sex: to account for systematic differences in brain structure between males and females (Ruigrok et al., 2014)
- The first 20 genetic principal components: to control for population stratification effects, which could create spurious correlations between genetic variants and brain structure due to shared ancestry rather than true biological relationships (Price et al., 2006)

After regressing out these confounders, each brain region of interest (ROI) was normalized by removing the mean and dividing by the standard deviation to ensure comparability across regions.

The genetic data was processed using PLINK (Purcell et al., 2007), retaining genetic variants with a minor allele frequency of at least 1% and a maximum missingness rate of 2%. Mean imputation was used to fill in missing values, and each variant was centered.

To generate measures of genetic disease risk, we calculated polygenic risk scores using PRSice (Euesden, Lewis, and O'Reilly, 2014). Scores were computed with a p-value threshold of 0.05 using GWAS summary statistics for the following diseases: Alzheimer's (Lambert et al., 2013), Schizophrenia (Trubetskoy et al., 2022), Bipolar disorder (Mullins et al., 2021), ADHD (Demontis et al., 2023), ALS (Rheenen et al., 2021), Parkinson's disease (Nalls et al., 2019), and Epilepsy (International League Against Epilepsy Consortium on Complex Epilepsies, 2018).

4.3.2 Experimental Setup

We apply our PLS-EY method to the UK Biobank dataset, using a mini-batch size of 500 and training for 100 epochs with a learning rate of 0.0001. A key computational challenge in this experiment is maintaining orthogonality between the weight vectors u_k in the PLS model, which is crucial for the method's effectiveness. This approach allows us to handle the high-dimensional nature of the data while preserving the interpretability of the learned representations. To the best of our knowledge, this experiment represents the largest-scale PLS analysis of biomedical data to date, demonstrating the potential of our method to facilitate discoveries in extremely large datasets.

4.3.3 Results and Observations

Figure V.6 shows the Pearson correlations among the PLS latent variables Z_k derived from the UK Biobank data. We observe strong correlations between corresponding pairs of representations $Z_k^{(1)}$ and $Z_k^{(2)}$, and weak cross-correlations between $Z_k^{(1)}$ and $Z_i^{(2)}$ for $i \neq k$. This indicates that our PLS-EY model learns a coherent and orthogonal subspace. While these results are shown on the training data and not cross-validated, two technical points are noteworthy: first, previous theoretical results in this thesis suggest that PLS tends to be biased towards the largest principal components, which are typically stable in datasets with power-law eigenvalue distributions common in biomedical applications. Second, while our method optimizes for covariance, we observe strong correlations in the learned representations - suggesting we may be detecting genuine structure rather than merely overfitting to spurious covariance patterns.

Furthermore, we investigate the associations between the PLS brain representations Z and the polygenic risk scores for various disorders, as shown in Figure V.7. The results reveal correlations between the learned representations and genetic risk measures for several disorders. While these associations would need to be validated in independent samples for biological inference, they demonstrate our method's ability to capture potential structure in high-dimensional data.

We note that while the imaging weights learned by the model could provide insights into the neuroanatomical patterns associated with genetic variation, a detailed analysis of these components is beyond the scope of this computational feasibility study. Future work could examine the biological interpretability of these patterns through rigorous cross-validation and replication in independent cohorts.

These results primarily demonstrate the scalability of our PLS-EY method to

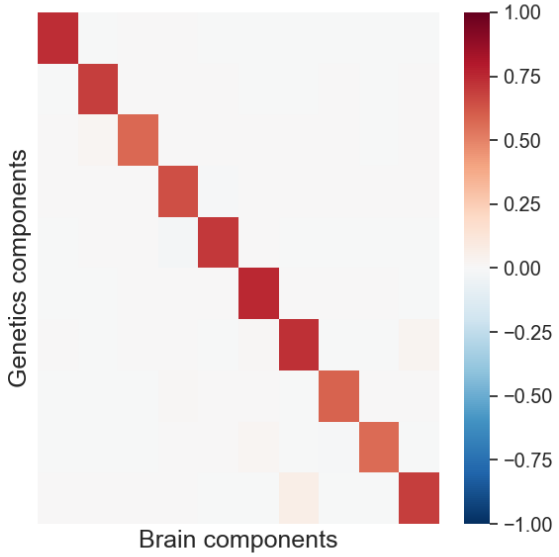


Figure V.6: Pearson correlations among PLS latent variables Z_k derived from UK Biobank data.

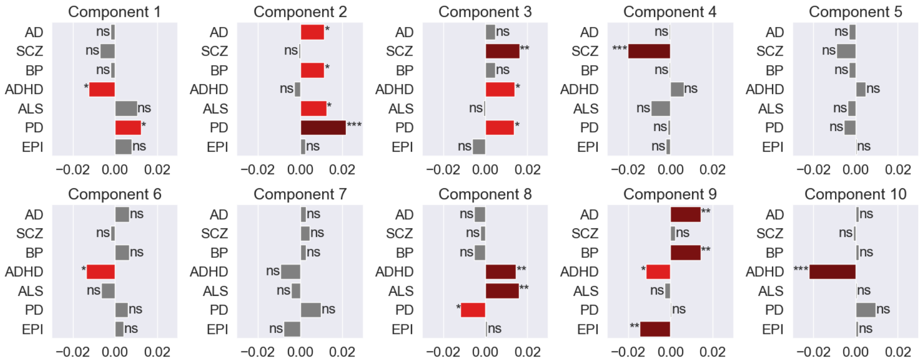


Figure V.7: Correlation between PLS brain representations Z and genetic risk scores for various disorders. AD=Alzheimer's disease, SCZ=Schizophrenia, BP=Bipolar disorder, ADHD=Attention deficit hyperactivity disorder, ALS=Amyotrophic lateral sclerosis, PD=Parkinson's disease, EPI=Epilepsy. ns : $0.05 < p \leq 1, * : 0.01 < p \leq 0.05, ** : 0.001 < p \leq 0.01, *** : 0.0001 < p \leq 0.001$.

extremely high-dimensional data and its ability to extract coherent patterns. The successful application of our method to the UK Biobank dataset highlights its potential as a computational tool for large-scale biomedical studies, though careful validation would be needed for any biological conclusions.

5 Discussion and Limitations

5.1 Limitations

This chapter presents a comprehensive exploration and development of novel algorithms for Canonical Correlation Analysis and Partial Least Squares, focusing on scalability and efficiency in high-dimensional and large-scale datasets. Our approach introduces the Eckhart-Young (EY) inspired objectives for Generalized Eigenvalue Problems and their application in stochastic or data-streaming settings, paving the way for more efficient and scalable solutions to classical subspace learning problems.

Our proposed CCA-EY and PLS-EY methods demonstrate significant advancements over traditional approaches in handling the computational complexity and scalability issues inherent in high-dimensional data. By reformulating the CCA and PLS objectives, we provide a path to efficiently analyze large datasets, which was previously infeasible due to computational limitations. The empirical evaluation on diverse datasets, including MediaMill, Split-CIFAR-10, and the UK Biobank, not only validates the effectiveness of our methods but also highlights their superiority in convergence speed and robustness to hyperparameter tuning.

The results from the MediaMill and Split-CIFAR-10 datasets underscore the potential of CCA-EY in achieving faster convergence with minimal hyperparameter tuning, a crucial factor for practical applications. This advantage is particularly pronounced when comparing our method to established baselines like γ-EigenGame and SGHA. Additionally, the application of our methods to the UK Biobank dataset represents a breakthrough in the analysis of imaging genetics data, showcasing the capability of PLS-EY to manage extraordinarily high-dimensional data while extracting meaningful and interpretable representations.

Furthermore, our methods' ability to capture relevant information for genetic disease risk, as evidenced in the UK Biobank study, opens new avenues for biomedical research. The significant associations between the PLS representations and genetic risk measures for various disorders provide valuable insights into the genetic mechanisms underlying diseases and brain morphometry.

6 Future Work: Proximal Gradient Descent for Regularized GEPs

While our current work focuses on unregularized GEPs, future research will extend our approach to incorporate complex regularization terms efficiently. We propose using proximal gradient descent, a method well-suited for optimizing loss functions comprising a smooth, differentiable component and a non-smooth regularization term.

6.1 Regularized Objective for GEP-EY

We aim to extend the GEP-EY framework by integrating specific regularization terms directly into the loss function. The revised loss function, denoted as $\mathcal{L}_{\text{Reg-GEP-EY}}(\theta)$, will incorporate regularization terms $R_i(\theta^{(i)})$ for each view i:

$$\mathcal{L}_{\text{Reg-GEP-EY}}(\theta) = \mathcal{L}_{\text{EY}}(\theta) + \sum_{i=1}^{I} \lambda_i R_i(\theta^{(i)}), \tag{V.30}$$

where $\mathcal{L}_{\text{EY}}(\theta)$ is our original GEP-EY loss function, λ_i are regularization parameters, and $R_i(\theta^{(i)})$ are view-specific regularization terms. This formulation allows for the inclusion of non-smooth penalties such as L1-norm or Total Variation (TV), which can enforce sparsity or structural constraints on the learned representations.

6.2 Proximal Gradient Descent Algorithm

The proximal gradient descent algorithm for optimizing the regularized GEP-EY objective alternates between a gradient step on the smooth part of the loss and a proximal step for the non-smooth regularization terms:

$$\theta_{t+1}^{(i)} = \text{prox}_{\alpha\lambda_i R_i}\left(\theta_t^{(i)} - \alpha \nabla_{\theta^{(i)}} \mathcal{L}_{\text{EY}}(\theta_t)\right), \tag{V.31}$$

where $\text{prox}_{\alpha\lambda_i R_i}(\cdot)$ denotes the proximal operator for the regularization term R_i with parameter λ_i, and α is the learning rate. The proximal operator is defined as:

$$\text{prox}_{\alpha\lambda_i R_i}(v) = \arg\min_u \left(R_i(u) + \frac{1}{2\alpha}\|u - v\|_2^2\right). \tag{V.32}$$

This approach effectively balances the influence of the smooth loss function gradient and the geometry imposed by the regularization, making it robust to the inclusion of complex constraints in GEP optimization.

6.3 Potential Applications and Benefits

The incorporation of proximal gradient descent into our GEP-EY framework opens up several exciting possibilities:

1. **Sparse CCA:** By using L1 regularization, we can encourage sparsity in the learned representations, leading to more interpretable results.
2. **Structured CCA:** Total Variation regularization can enforce spatial or temporal smoothness in the representations, which is particularly useful for neuroimaging or time-series data.
3. **Multi-view learning with heterogeneous regularization:** Different regularization terms can be applied to different views, allowing for more flexible modeling of complex multi-view data.
4. **Improved generalization:** Appropriate regularization can help prevent overfitting, especially in high-dimensional, low-sample size scenarios.

Future work will focus on implementing this proximal gradient descent approach, developing efficient algorithms for specific regularization terms, and evaluating its performance on various real-world datasets.

6.4 Conclusion

In summary, this chapter contributes to the fields of machine learning and multiview data analysis by introducing scalable and efficient solutions for CCA and PLS, applicable in a variety of domains, including but not limited to neuroimaging and genetics. Our work not only addresses significant computational challenges but also lays the groundwork for future research and practical applications in analyzing large-scale, high-dimensional datasets.

```
def compute_unbiased_stochastic_loss(Z, Z_prime, alpha=None):
    # Compute C
    C = sum(torch.cov(Z[i], Z[j]) for i in range(len(Z)) for j in range(i+1, len(Z)))

    # Compute V and V_prime
    V = sum(alpha[i] * torch.matmul(Z[i].T, Z[i]) +
            (1 - alpha[i]) * torch.var(Z[i], dim=0)
            for i in range(len(Z)))
    V_prime = sum(alpha[i] * torch.matmul(Z_prime[i].T, Z_prime[i]) +
                  (1 - alpha[i]) * torch.var(Z_prime[i], dim=0)
                  for i in range(len(Z_prime)))

    # Compute loss
    loss = -2 * torch.trace(C) + torch.sum(V * V_prime)
    return loss

def unbiased_stochastic_gep_ey(data_stream, lr, T, f):
    theta = initialize_parameters(f)
    optimizer = torch.optim.SGD([theta], lr=lr)

    for t in range(T):
        X_b, X_b_prime = next(data_stream), next(data_stream)
        Z_b, Z_b_prime = f(X_b, theta), f(X_b_prime, theta)

        optimizer.zero_grad()
        loss = compute_unbiased_stochastic_loss(Z_b, Z_b_prime)
        loss.backward()
        optimizer.step()

    return theta
```

Listing 2: PyTorch-style pseudocode for the unbiased stochastic GEP-EY algorithm

Chapter VI

Deep Stochastic CCA: Bridging Multiview and Self-Supervised Learning

Contents

Preface

This chapter is based on work presented in Chapman and Wells (2023) and Chapman, Wells, and Aguila (2024).

1 Introduction

Deep CCA (Andrew et al., 2013) secured a runner-up position for the test-of-time award at ICML 2023 (ICML, 2023). However, its direct application has been limited in large datasets due to biased gradients in the stochastic minibatch setting. There have since been proposals to scale-up Deep CCA in the stochastic case with adaptive whitening (W. Wang, Arora, Livescu, and Srebro, 2015) and regularization (Chang, Xiang, and T. M. Hospedales, 2018), but these techniques are highly sensitive to hyperparameter tuning.

Self-Supervised Learning (SSL) methods have reached state-of-the-art in tasks such as image classification (Balestriero, Ibrahim, et al., 2023), learning representations without labels that capture salient features so that they can be used for general downstream tasks. A family of SSL methods that are closely aligned with Canonical Correlation Analysis has garnered particular interest. This family notably includes Barlow Twins (Zbontar et al., 2021), VICReg (Bardes, Ponce, and LeCun, 2021), and W-MSE (Ermolov et al., 2021) and they aim to transform a pair of data views into similar representations, similar to the objective of CCA. Similarly, some generative approaches to SSL(Sansone and Manhaeve, 2022) bear a striking resemblance to Probabilistic CCA(Bach and Jordan, 2005). These connections have started to be explored in Balestriero and LeCun (2022).

In this chapter, we propose a novel formulation of Deep CCA that is unbiased in the stochastic setting and scales to large datasets. We also propose a novel SSL method, SSL-EY, that is competitive with existing methods on CIFAR-10 and CIFAR-100. We highlight the connections between our work and existing SSL methods, and show that our method is more robust to hyperparameter tuning.

2 Background: Deep Representation Learning

This section explores the evolution of deep representation learning techniques, from traditional autoencoders to more sophisticated approaches like Deep Canonical Correlation Analysis (Deep CCA) and Self-Supervised Learning (SSL).

2.1 Deep Learning and Neural Networks

Deep learning, a subfield of machine learning, utilizes neural networks to learn complex representations from data. Neural networks are composed of interconnected layers of artificial neurons, typically including an input layer, one or more hidden layers, and an output layer. Each neuron computes a weighted sum of its inputs, followed by a nonlinear activation function. A key component of modern neural networks is the rectified linear unit (ReLU) activation function, defined as:

$$\mathrm{ReLU}(x) = \max(0, x) \tag{VI.1}$$

ReLU and its variants have become popular due to their simplicity and effectiveness in mitigating the vanishing gradient problem during training. The power of neural networks lies in their ability to approximate complex functions. The Universal Approximation Theorem (Cybenko, 1989) states that a feedforward network with a single hidden layer containing a finite number of neurons can approximate any continuous function on compact subsets of $\mathbb{R}^n$, given certain mild assumptions about the activation function. Neural networks are typically trained using backpropagation and stochastic gradient descent (SGD) or its variants. The backpropagation algorithm efficiently computes gradients of the loss function with respect to the network parameters, while SGD allows for training on large datasets by updating parameters using small batches of data.

2.2 Autoencoders and Representation Learning

Autoencoders represent an early approach to unsupervised representation learning. These neural networks are designed to learn compact data representations by reconstructing input data through a bottleneck layer. The basic structure of an autoencoder consists of:

An encoder function $f_\theta : \mathcal{X} \rightarrow \mathcal{Z}$ that maps input data to a latent representation. A decoder function $g_\phi : \mathcal{Z} \rightarrow \mathcal{X}$ that attempts to reconstruct the input from the latent representation.

The autoencoder is trained to minimize a reconstruction loss:

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{x \sim p_{\text{data}}}[||x - g_\phi(f_\theta(x))\|^2] \tag{VI.2}$$

Autoencoders can be viewed as a nonlinear generalization of Principal Component Analysis (PCA). In fact, a linear autoencoder with a single hidden layer and mean squared error loss is equivalent to PCA when the hidden layer dimension is less than the input dimension. While autoencoders have been successful in various applications, they face limitations. The focus on reconstruction may lead to learning fine-grained details that are not necessarily useful for downstream tasks, and there's no explicit encouragement for the learned representations to capture meaningful features or disentangled factors of variation.

These limitations have motivated the development of more sophisticated autoencoder variants and alternative representation learning approaches.

2.3 Variational Autoencoders

Variational Autoencoders (VAEs) (Kingma and Welling, 2013) extend the autoencoder framework to learn probabilistic generative models. VAEs model the latent space as a probability distribution, typically a multivariate Gaussian, and are trained to maximize the evidence lower bound (ELBO):

$$\mathcal{L}_{ELBO}(\theta, \phi) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p(z)) \tag{VI.3}$$

where $q_\phi(z|x)$ is the encoder (or inference) network, $p_\theta(x|z)$ is the decoder (or generative) network, and $p(z)$ is a prior distribution over the latent space. VAEs provide a principled way to generate new samples and perform probabilistic inference, making them useful for both representation learning and generative modeling.

Research has extended Autoencoders to multiview settings, such as Multiview Autoencoders (Ngiam et al., 2011) which regularise autoencoders for each view to be somewhat similar to each other. Contributions of my own to this area include applications to neuroscience data (Lawry Aguila, Chapman, and Altmann, 2023) and the development of a software package for multiview autoencoders (Townsend, Chapman, and Cole, 2023).

However given the deep connection between Autoencoders and PCA, it is natural

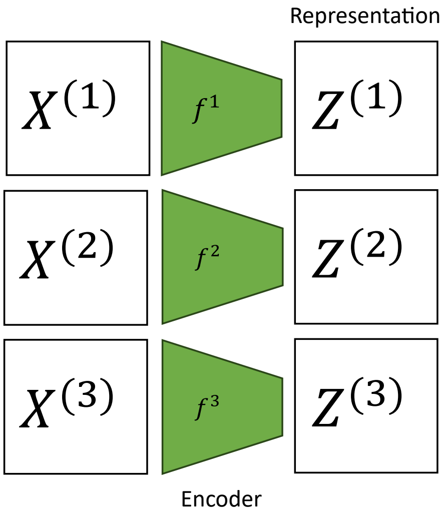


Figure VI.1: Schematic of the Deep CCA approach highlighting the nonlinear transformation of data into correlated views.

to consider the extension of CCA to the deep learning setting. This leads to the development of Deep CCA, which aims to learn nonlinear representations that are maximally correlated across different views.

2.4 From Autoencoders to Deep CCA

A natural objective of Deep CCA is to maximize the correlation between learned *representations* of different views:

$$\|\mathrm{MCCA}_K\left(Z^{(1)},\ldots,Z^{(I)}\right)\|_2 \tag{VI.4}$$

where $Z^{(i)} = f^{(i)}(X^{(i)};\theta^{(i)})$ are the learned representations for each view. Figure VI.1 illustrates how Deep CCA transforms data from different views through neural networks to achieve correlated representations. This approach allows for capturing complex, nonlinear relationships between views that linear methods might miss.

2.4.1 Full-batch Deep CCA

The full-batch approach of Deep CCA, formulated by Andrew et al. (2013) has been the foundation for subsequent work in this area and won the ICML 2023 test-of-time award. This method maximizes the correlation between learned representations using the following loss function:

$$T = \left(\text{cov}(Z^{(1)})\right)^{-\frac{1}{2}} Z^{(1)\top} Z^{(2)} \left(\text{cov}(Z^{(2)})\right)^{-\frac{1}{2}} \tag{VI.5}$$

$$\mathcal{L}_{\text{Rayleigh}} = -\operatorname{Tr}(T) \tag{VI.6}$$

We refer to this as the Rayleigh loss because it effectively maximizes the Generalized Rayleigh Quotient associated with the canonical correlation problem. While theoretically sound, this approach faces substantial scalability issues with large datasets.

The Rayleigh loss is only valid for full batch gradient descent, which becomes computationally infeasible for large datasets. This limitation has led to numerous issues reported on the GitHub implementation, with users experiencing problems with ill-conditioned matrices and exploding gradients. These problems arise because the covariance matrices can become singular or nearly singular when working with small batches or high-dimensional data, leading to instability in the matrix inversions required by the loss function.

2.4.2 DCCA-STOL

DCCA-STOL, proposed by W. Wang, Arora, Livescu, and Bilmes (2015), attempts to use the full-batch objective with large mini-batches but suffers from biased gradients due to the matrix inversions in Equation equation VI.5. This bias is fundamentally a consequence of Jensen's inequality applied to the inverse of a random variable.

For a positive random variable X, Jensen's inequality states that:

$$\frac{1}{E[X]} \leq E[\frac{1}{X}] \tag{VI.7}$$

This inequality arises because $f(x) = \frac{1}{x}$ is a convex function for $x > 0$.

To illustrate this, consider a simple example: Let X be a random variable that is 1 with probability 0.5 and 0.1 with probability 0.5. Then:

$$E[X] = 0.5 \cdot 1 + 0.5 \cdot 0.1 = 0.55 \tag{VI.8}$$

$$\frac{1}{E[X]} = \frac{1}{0.55} \approx 1.82 \tag{VI.9}$$

$$E[\frac{1}{X}] = 0.5 \cdot \frac{1}{1} + 0.5 \cdot \frac{1}{0.1} = 0.5 + 5 = 5.5 \tag{VI.10}$$

As we can see, $\frac{1}{E[X]} < E[\frac{1}{X}]$, confirming the inequality.

In the context of DCCA-STOL, this inequality leads to biased estimates when inverting covariance matrices estimated from mini-batches. The expectation of the inverse of the sample covariance matrix is not equal to the inverse of the expected covariance matrix. This bias necessitates batch sizes larger than the representation size, limiting the method's practical application.

The problem becomes even more severe when the batch size is smaller than the representation dimension. In this case, the sample covariance matrix is guaranteed to be singular. To understand why, consider that a batch of size n in a d-dimensional space (where $n < d$) can only span an n-dimensional subspace. Consequently, the sample covariance matrix will have at most rank n, making it singular in the d-dimensional space.

For singular matrices, the inverse is undefined, which is equivalent to division by zero in the scalar case. This causes Jensen's inequality to blow up to infinity, making the optimization problem ill-posed and impossible to solve.

Even when the batch size is slightly larger than the representation dimension, the covariance matrix can be nearly singular, leading to numerical instability and unreliable results.

A practical example illustrates the severity of this issue: Consider training deep neural networks on 3D MRI scans. Due to memory constraints of GPUs, it's often impossible to use large batch sizes for such high-dimensional data. If we attempt to use small batch sizes with the Rayleigh or STOL objective, we encounter singular (when batch size < representation dimension) or near-singular (when batch size $\approx$ representation dimension) matrices. The singularity effectively causes division by zero in Jensen's inequality, blowing up the expectation to infinity.

This problem has caused confusion for many GitHub users (including myself!) who attempted to implement these methods. The nuanced nature of this issue is not immediately apparent, leading to unexpected behavior and optimization failures. It's crucial to understand that to even get the optimization to run, we require batch sizes substantially larger than the representation dimension, before even considering problems of bias. This requirement severely limits the applicability of the method to high-dimensional data or large models, as it demands enormous batch sizes that often exceed available computational resources.

2.4.3 Deep MCCA and Deep GCCA

Extensions such as Deep MCCA (Somandepalli et al., 2019) and Deep GCCA (Benton et al., 2017) are multiview extensions of Deep CCA. Recalling the notation from 3.2.3, where $\hat{C}(\theta)$ and $\hat{V}(\theta)$ are the mini-batch estimates of the between-view and within-view covariance matrices, respectively, the loss function for Deep MCCA is given by:

$$T = \left(\hat{V}(\theta)^{-\frac{1}{2}}\hat{C}(\theta)\hat{V}(\theta)^{-\frac{1}{2}}\right) \tag{VI.11}$$

$$\mathcal{L}_{\text{DMCCA}} = -\operatorname{Tr}(T) \tag{VI.12}$$

However, these methods still require large batch sizes due to biased gradients from small mini-batch covariance matrices. This issue stems from the same problem as DCCA-STOL: the matrix inversions in the loss function lead to biased estimates when using small mini-batches, again due to Jensen's inequality.

2.4.4 Adaptive Whitening Methods

Adaptive whitening methods (W. Wang, Arora, Livescu, and Srebro, 2015; Chang, Xiang, and T. M. Hospedales, 2018) offer another solution to the scalability problem. These methods aim to approximate the matrix inversion in the loss function without actually inverting the matrix, functioning similarly to a preconditioner in optimization. By doing so, they attempt to mitigate the bias introduced by direct matrix inversion on small mini-batches.

One such method is DCCA-NOI (W. Wang, Arora, Livescu, and Bilmes, 2015), which uses Nonlinear Orthogonal Iteration to approximate the whitening transformation. The loss function of DCCA-NOI is:

$$\mathcal{L}_{\text{NOI}} = |\tilde{\Sigma}_{11}^{-\frac{1}{2}}Z^{(1)} - \tilde{\Sigma}_{22}^{-\frac{1}{2}}Z^{(2)}|_F^2 \tag{VI.13}$$

where $\tilde{\Sigma}_{11}$ and $\tilde{\Sigma}_{22}$ are estimates of the covariance matrices of $Z^{(1)}$ and $Z^{(2)}$. The adaptive whitening approach aims to iteratively refine these estimates over time, potentially allowing for smaller batch sizes. However, these methods introduce a time constant that complicates analysis and requires extensive tuning. This time constant represents the rate at which the whitening estimates are updated, and finding the right balance between stability and adaptivity can be challenging.

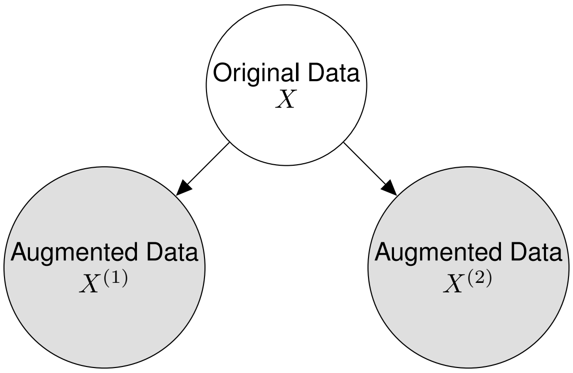


Figure VI.2: Joint Embedding Data Generation Process in SSL

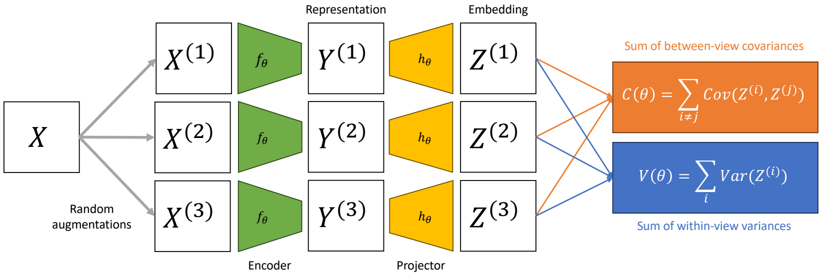


Figure VI.3: Encoder-Projector Model in SSL

2.5 Self-Supervised Learning and Joint Embedding

Self-Supervised Learning (SSL) has emerged as a powerful approach for learning representations from unlabeled data. A key strategy in SSL involves creating joint embeddings of augmented data, typically images.

SSL methods often employ an encoder-projector model:

2.5.1 CCA-based SSL Methods: Barlow Twins and VICReg

Two prominent self-supervised learning (SSL) methods, Barlow Twins and VICReg (Variance-Invariance-Covariance Regularization), utilize objectives that share similarities with Canonical Correlation Analysis principles. These methods aim to learn representations that are invariant to data augmentations while maintaining informative and decorrelated features.

Barlow Twins Introduced by Zbontar et al. (2021), Barlow Twins is inspired by the redundancy reduction principle in neuroscience (Barlow et al., 1961). The method aims to create embeddings that are invariant to distortions of the input sample while avoiding the collapse of the learned representations.

Barlow Twins Loss:

$$\mathcal{L}_{\mathsf{BT}} = \underbrace{\gamma \mathbb{E}\|Z^{(1)} - Z^{(2)}\|^2}_{\text{Invariance}} + \underbrace{\beta \sum_{\substack{k,l=1 \\ k \neq l}}^{K} \operatorname{Cov}(\hat{Z}_k^{(i)}, \hat{Z}_l^{(i)})^2}_{\text{Redundancy Reduction}} \tag{VI.14}$$

The loss function consists of two terms:

- **Invariance term:** Encourages the embeddings of distorted versions of a sample to be similar.
- **Redundancy reduction term:** Minimizes the redundancy between the dimensions of the embedding vectors.

VICReg VICReg, proposed by Bardes, Ponce, and LeCun (2021), builds upon the ideas of Barlow Twins but introduces an additional variance term to explicitly encourage the embeddings to be diverse and avoid dimensional collapse.

VICReg Loss:

$$\begin{aligned}\mathcal{L}_{\mathsf{VR}} = &\underbrace{\gamma \mathbb{E}\|Z^{(1)} - Z^{(2)}\|^2}_{\text{Invariance}} + \\ &\underbrace{\sum_{i \in \{1,2\}} \alpha \sum_{k=1}^{K} \left(1 - \sqrt{\operatorname{Var}(Z_k^{(i)})}\right)_+}_{\text{Variance}} + \underbrace{\beta \sum_{\substack{k,l=1 \\ k \neq l}}^{K} \operatorname{Cov}(Z_k^{(i)}, Z_l^{(i)})^2}_{\text{Covariance}}\end{aligned} \tag{VI.15}$$

The VICReg loss function comprises three terms:

- **Invariance term:** Similar to Barlow Twins, it encourages the embeddings of augmented versions of a sample to be close.
- **Variance term:** Ensures that the variance of each embedding dimension is above a certain threshold, preventing dimensional collapse.
- **Covariance term:** Minimizes the covariance between different dimensions of the embeddings, promoting decorrelation.

Relation to CCA Both Barlow Twins and VICReg share similarities with CCA in their objectives:

- The invariance terms in both methods are analogous to maximizing correlation in CCA, as they encourage similar representations for related inputs.
- The redundancy reduction term in Barlow Twins and the covariance term in VICReg are similar to the orthogonality constraints in CCA, promoting decorrelated features.
- VICReg's variance term can be seen as a way to ensure that the learned representations capture meaningful variations in the data, which is implicitly achieved in CCA through its formulation.

These methods demonstrate the effectiveness of CCA-inspired principles in self-supervised learning, providing a promising direction for learning robust and informative representations without relying on labeled data.

3 Methods: DCCA-EY and SSL-EY, extending GEP-EY to Deep Learning

Building upon the Eckart-Young inspired objective introduced in the previous chapter, we now extend our approach to non-linear transformations of the data. The key idea is to replace linear representations with non-linear ones, effectively optimizing the representation of the original data to maximize canonical correlations.

Recall the objective function from the previous chapter:

$$\mathcal{L}_{\mathsf{EY}}(\theta) = -2 \operatorname{Tr} C(\theta) + \|V_\alpha(\theta)\|_F^2 \tag{VI.16}$$

In this chapter, we consider non-linear transformations of the data, defined as:

$$Z^{(i)} = f^{(i)}(X^{(i)}; \theta^{(i)}) \tag{VI.17}$$

where $f^{(i)}$ are non-linear functions (typically neural networks) parameterized by $\theta^{(i)}$.

3.1 Deep Multiview Canonical Correlation Analysis (DCCA-EY)

We first show that our objective recovers Deep Multi-view CCA at any local optimum, assuming a final linear layer in each neural network.

Lemma 3.1. *[Objective recovers Deep Multi-view CCA] Assume that there is a final linear layer in each neural network $f^{(i)}$. Then at any local optimum, $\hat{\theta}$, of the population problem, we have*

$$\mathcal{L}_{\mathit{EY}}(\hat{\theta}) = -\|\mathrm{MCCA}_K(\hat{Z})\|_2^2$$

where $\hat{Z} = f_{\hat{\theta}}(X)$. Therefore, $\hat{\theta}$ is also a local optimum of objectives from Andrew et al. (2013) and Somandepalli et al. (2019) as defined in Equation (VI.4)*.*

Proof sketch: Consider treating the penultimate-layer representations as fixed, and optimising over the weights in the final layer. This is precisely equivalent to optimising the Eckhart-Young loss for linear CCA where the input variables are the penultimate-layer representations. So by Proposition 3.1 the optimal value is the negative sum of squared generalised eigenvalues. □

This result shows that our objective, which we call **DCCA-EY**, is a valid generalization of Deep CCA and can be used to learn correlated non-linear representations.

3.2 Application to Self-Supervised Learning (SSL-EY)

We can directly apply the DCCA-EY approach to the self-supervised learning (SSL) setting, which we call **SSL-EY**. The key differences between DCCA-EY and SSL-EY are:

1. Data source: In SSL-EY, the two views are augmented versions of a single sample, whereas in DCCA-EY, they are separate views of the data.
2. Encoder architecture: SSL-EY uses the same encoder for both views as a regularization strategy, while DCCA-EY uses separate encoders for each view.

The use of a shared encoder in SSL-EY is motivated by the fact that the paired data are generated from applying independent, identically distributed (i.i.d.) augmentations to the same original input. This approach acts as a regularizer and is intuitively sensible given that the distributions of both views are identical.

Our loss function for both DCCA-EY and SSL-EY bears some resemblance to those of Barlow Twins and VICReg:

$$\mathcal{L}_{\mathsf{EY}}(\theta) = -2\,\mathrm{trace}\, C(\theta) + \|V_\alpha(\theta)\|_F^2$$

where $C(\theta)$ is the cross-covariance matrix between the representations of the views, and $V_\alpha(\theta)$ is a matrix involving the individual covariance matrices of each view.

This objective has two terms:

1. $-2\operatorname{trace} C(\theta)$: encourages correlated representations across views, similar to the invariance term in Barlow Twins and VICReg.

2. $\|V_\alpha(\theta)\|_F^2$: involves individual covariance matrices, analogous to the variance and covariance terms in VICReg.

The main difference is that our method is based on canonical correlation principles, which may offer additional benefits in terms of representation quality and interpretability.

3.3 PyTorch Implementation

We provide a unified PyTorch implementation for both DCCA-EY and SSL-EY in Listing 3. This implementation defines a single class that can be used for both methods, with the key difference being in how the encoders are initialized and used.

This unified implementation can be used for both DCCA-EY and SSL-EY by setting the `ssl_mode` parameter appropriately. When `ssl_mode=True`, it uses a single shared encoder for both views (SSL-EY), and when `ssl_mode=False`, it uses separate encoders for each view (DCCA-EY).

In the next section, we will present experiments demonstrating the effectiveness of our DCCA-EY and SSL-EY methods in their respective settings.

4 Experiments and Results

4.1 Deep CCA

This experiment aims to demonstrate the effectiveness of our DCCA-EY method compared to existing Deep Canonical Correlation Analysis approaches including DCCA-STOL and DCCA-NOI. We focus on three key aspects: correlation capture, convergence speed, and ease of hyperparameter tuning. Our experimental setup closely follows that of W. Wang, Arora, Livescu, and Srebro (2015), enabling a direct and fair comparison.

4.1.1 Datasets

We evaluate our method on two diverse datasets: Split MNIST and the X-Ray Microbeam Speech Production Database (XRMB).

The Split MNIST dataset is a modified version of the original MNIST dataset, designed to challenge models with partial information. Each 28x28 pixel grayscale image is split into left and right halves, creating two distinct views. The dataset consists of 50,000 training images and 10,000 test images. This splitting mechanism tests the model's ability to learn correlated representations from incomplete digit information.

The XRMB dataset provides a more complex, real-world challenge in the domain of speech production. It comprises approximately 40,000 spoken utterances from 47 American English speakers, offering a rich multiview perspective on articulatory speech data. The dataset presents two views: acoustic features and articulatory measurements. The acoustic features are represented by 273-dimensional vectors capturing spectral characteristics, while the articulatory measurements are 112-dimensional vectors describing the position and movement of speech articulators. The high dimensionality and complexity of XRMB make it an excellent testbed for assessing the scalability and robustness of DCCA methods.

4.1.2 Experimental Setup

Our experimental design is crafted to rigorously evaluate DCCA-EY against existing methods across a range of operational conditions. We employ a network architecture consisting of multilayer perceptrons with two hidden layers, each containing 800 units, followed by an output layer of 50 units. ReLU activations are used throughout the network, closely aligning with the architecture used by W. Wang, Arora, Livescu, and Srebro (2015). This architectural choice ensures a fair comparison with previous work while providing sufficient model capacity to capture complex correlations between views.

To thoroughly assess both the initial convergence behavior and long-term performance stability of each method, we train each model for 50 epochs. This extended training period allows us to observe how different approaches perform over time, particularly in terms of their ability to maintain and refine learned correlations.

The dimensionality of the learned representations (K) is set to 50, matching the output layer size. This choice facilitates direct comparison with previous work and ensures that we are evaluating each method's ability to capture a consistent number of canonical components.

To evaluate performance across different levels of stochasticity, we experiment with batch sizes of 20, 50, and 100. This range allows us to assess each method's robustness to varying degrees of sample randomness, with a particular focus on how well they perform with smaller batch sizes that introduce more stochasticity into the training process.

Our hyperparameter tuning process includes a grid search over learning rates (1e-3, 1e-4, 1e-5) for all methods, with additional tuning of ρ values (0.6, 0.8, 0.9) for DCCA-NOI. By exploring more hyperparameters for the benchmark DCCA-NOI method compared to our approach, we ensure a fair comparison that, if anything, favors the baseline method.

Through this carefully designed experimental setup, we aim to systematically evaluate DCCA-EY against existing methods, with a particular focus on its performance across different batch sizes. By closely following the methodology of W. Wang, Arora, Livescu, and Srebro (2015) while introducing our own innovations, we strive to provide a fair and comprehensive comparison that highlights the strengths of our approach, especially in terms of correlation capture, convergence speed, and ease of hyperparameter tuning.

4.1.3 Evaluation Metric: Total Correlation Captured (TCC)

We introduce the Total Correlation Captured (TCC) metric for evaluation:

$$\text{TCC} = \sum_{k=1}^{K} \rho_k$$

where ρ_k are the empirical correlations between the neural network-based representations $Z^{(i)} = f^{(i)}(X^{(i)})$ on a validation set. The TCC metric differs from previously used measures such as the Proportion of Correlation Captured (PCC) in that it does not require ground truth correlations for its computation, making it applicable to datasets where such ground truth is unavailable. By evaluating on a validation set, TCC measures how well the learned correlations generalize to unseen data.

TCC provides a sum of the estimated correlations across views: a higher TCC value indicates stronger correlations between the learned representations in the validation set. This allows for quantitative comparison between different methods and across various experimental conditions.

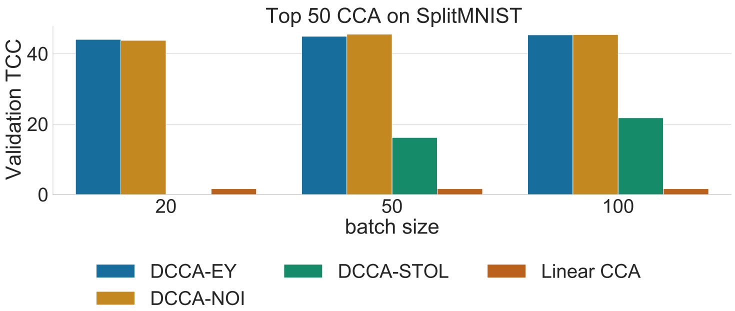


Figure VI.4: Deep CCA on SplitMNIST: Comparison of methods across varying batch sizes.

4.1.4 Objectives

This experiment compares DCCA-EY with existing DCCA methods across three dimensions: correlation capture (measured by TCC scores), convergence speed, and behavior across varying batch sizes. We examine how each method performs under different computational constraints, focusing particularly on scalability and stability during training. Through these comparisons, we aim to understand the practical trade-offs between DCCA-EY and current approaches when applied to different dataset sizes and computational settings.

4.1.5 Observations on SplitMNIST

For the SplitMNIST dataset, Figure VI.4 shows the comparison of methods across different batch sizes. We observe that DCCA-STOL captures substantially less correlation than the other methods and breaks down when the mini-batch size is smaller than the dimension $K = 50$. Figure VI.5 illustrates the learning progress over 50 epochs, where DCCA-NOI, despite performing similarly to DCCA-EY, requires more careful hyperparameter tuning and demonstrates a slower convergence speed.

4.1.6 Observations on XRMB

On the XRMB dataset, as seen in Figure VI.6, similar trends are evident. DCCA-STOL struggles with smaller mini-batch sizes, while DCCA-NOI, though comparable to DCCA-EY in performance, lags in convergence speed, as shown in Figure VI.7.

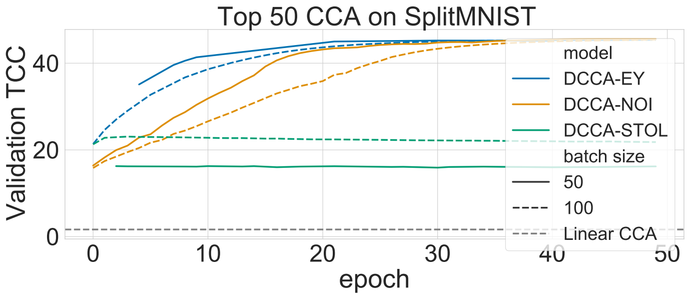


Figure VI.5: Deep CCA on SplitMNIST: Learning progress over 50 epochs.

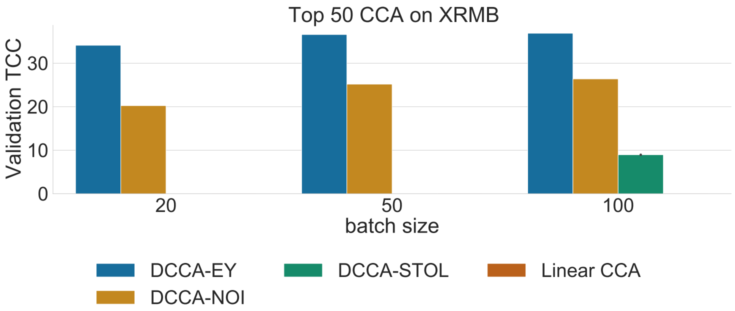


Figure VI.6: Deep CCA on XRMB: Comparison of methods across varying batch sizes.

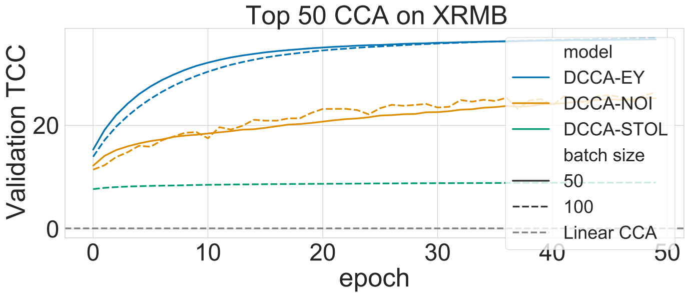


Figure VI.7: Deep CCA on XRMB: Learning progress over 50 epochs.

4.2 Deep Multiview CCA: Robustness Across Different Batch Sizes

In this experiment, we evaluate the performance and adaptability of our DCCA-EY method in a multiview context, comparing it against established multiview DCCA methods including Deep MCCA and Deep GCCA. Our primary goal is to assess how these methods perform across various batch sizes, with a particular focus on their ability to capture correlations between multiple views.

4.2.1 Dataset: mfeat

For this experiment, we utilize the mfeat dataset (Duin, n.d.), which provides an ideal testbed for multiview learning methods. This dataset consists of 2,000 handwritten numeral patterns, each represented by six distinct feature sets. These feature sets include Fourier coefficients, profile correlations, Karhunen-Love coefficients, pixel averages in 2x3 windows, Zernike moments, and morphological features.

The mfeat dataset is particularly valuable for our study due to its diverse feature types, which challenge the algorithms' ability to find correlations across heterogeneous data representations. This diversity allows us to assess the robustness and flexibility of our DCCA-EY method in dealing with varied data characteristics, a crucial aspect in real-world multiview learning scenarios.

4.2.2 Experimental Setup

Our experimental design aims to comprehensively evaluate the performance of various multiview Deep CCA methods across a range of batch sizes, with a particular focus on small batch scenarios. We hypothesize that our method will demonstrate superior performance across all batch sizes, but with a marked advantage in smaller batch conditions where existing methods are expected to struggle.

The core of our experiment involves training each Deep CCA variant to learn a shared representation space of 50 dimensions (K = 50). To ensure a thorough assessment of each method's convergence properties and final performance, we extend the training process over 100 epochs. This duration allows us to observe both the initial learning dynamics and the long-term stability of the learned representations.

A key aspect of our experimental design is the variation in batch sizes. We explore a wide spectrum, ranging from very small batches of 5 samples to larger batches of 200 samples. Specifically, we test batch sizes of 5, 10, 20, 50, 100, and

200. This range enables us to closely examine how each method's performance scales with increasing batch size, with particular attention to the challenging small-batch regime.

To ensure a fair comparison and to optimize each method's performance, we conduct a comprehensive hyperparameter search. The key hyperparameters and their ranges are summarized in Table 4.1.

Table 4.1: Hyperparameter ranges for Deep CCA methods

Parameter	Values
Representation Dimensionality	50
Training Epochs	100
Batch Sizes	5, 10, 20, 50, 100, 200
Learning Rates	0.01, 0.001, 0.0001, 0.00001

By systematically varying these parameters, particularly the batch size, we aim to provide a nuanced understanding of each method's strengths and limitations. Our analysis will focus on how performance metrics evolve across different batch sizes, with special attention to the small-batch scenarios where we anticipate our method to show significant advantages.

This experimental setup allows us to rigorously test our hypothesis that our proposed Deep CCA variant offers superior performance across a range of operational conditions, with a particular emphasis on its robustness in challenging small-batch scenarios. Through this comprehensive evaluation, we seek to demonstrate the practical advantages of our approach in real-world applications where small batch processing is often necessary due to computational or data streaming constraints.

4.2.3 Evaluation Metric

To effectively compare the performance of different methods in this multiview setting, we introduce a novel metric: the Total Multiview Correlation Captured (TMCC). This metric extends the concept of Total Correlation Captured to accommodate multiple views. The TMCC is defined as:

$$\text{TMCC} = \sum_{k=1}^{K} \frac{1}{I(I-1)} \sum_{\substack{i,j \leq I \\ i \neq j}} \text{corr}(Z_k^{(i)}, Z_k^{(j)}),$$

where $Z_k^{(i)}$ represents the k-th dimension of the i-th view's representation, and I is the total number of views. This metric quantifies the average correlation across all pairs of views for each dimension of the learned representations. A higher TMCC indicates better capture of inter-view correlations, reflecting the method's effectiveness in learning a shared multiview representation.

4.2.4 Objectives

Our experiment is designed with several key objectives. First, we aim to evaluate DCCA-EY's performance in capturing multiview correlations, comparing it directly with Deep MCCA and Deep GCCA. Second, we seek to assess the robustness of these methods across a wide range of batch sizes, from very small (5) to relatively large (200).

4.2.5 Observations

Figure VI.8 illustrates the comparison of DCCA-EY with Deep GCCA and Deep MCCA across different mini-batch sizes, using the validation TMCC metric. DCCA-EY consistently outperforms both Deep GCCA and Deep MCCA, showcasing its superior ability to capture validation TMCC. Notably, Deep MCCA encounters issues when the batch size is smaller than $K = 50$, likely due to singular empirical covariances. Deep GCCA, while not breaking down, underperforms with smaller batch sizes, highlighting limitations in scalability and efficiency for large-scale data applications.

In Figure VI.9, we observe the learning curves for batch sizes 50 and 100. Both Deep MCCA and Deep GCCA demonstrate rapid initial learning of correlations but reach a plateau relatively quickly. In contrast, DCCA-EY exhibits a consistent improvement over time and notably outperforms the other methods by the end of the training period. This behavior underscores the enhanced learning capability and efficiency of DCCA-EY, especially in the context of large-scale, high-dimensional data.

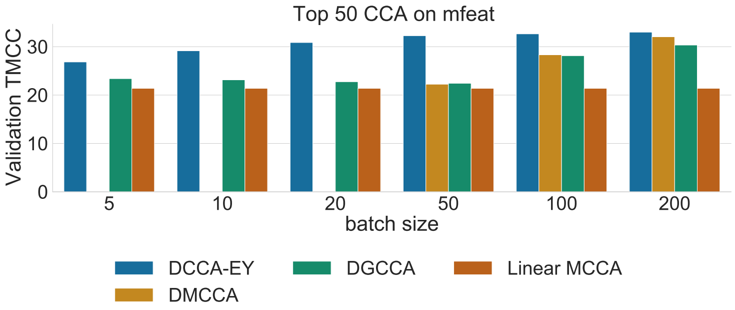


Figure VI.8: Deep Multi-view CCA on mfeat: Comparison across various mini-batch sizes using the Validation TMCC metric.

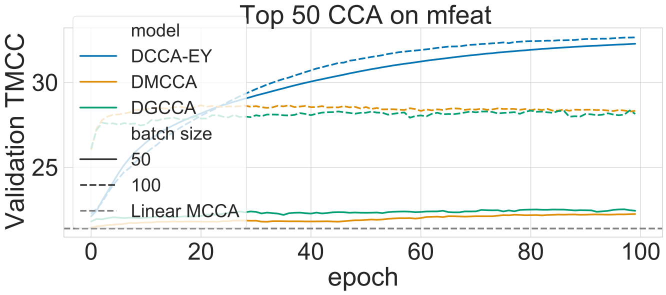


Figure VI.9: Deep Multi-view CCA on mfeat: Learning progress over 100 epochs for batch sizes 50 and 100.

4.3 Self-Supervised Learning with SSL-EY

To evaluate the effectiveness of our proposed SSL-EY method in self-supervised learning tasks, we conduct a comprehensive benchmark comparison against two established baselines: Barlow Twins and VICReg. This experiment aims to assess SSL-EY's performance in learning meaningful representations from unlabeled data.

4.3.1 Datasets

We evaluate our method on two widely-used benchmark datasets: CIFAR-10 and CIFAR-100. Both datasets consist of 60,000 32x32 color images, divided into 50,000 training images and 10,000 test images.

CIFAR-10 contains images from 10 classes, with 6,000 images per class. This dataset provides a balanced representation of basic object categories, offering a good starting point for evaluating self-supervised learning methods.

CIFAR-100, on the other hand, presents a more challenging scenario with 100 classes and only 600 images per class. This increased number of classes and reduced per-class sample size tests the methods' ability to learn discriminative features in a more fine-grained classification setting.

These datasets provide a diverse range of image complexities and class structures, allowing for a comprehensive evaluation of SSL-EY's performance across different scenarios. The contrast between CIFAR-10 and CIFAR-100 enables us to assess how well each method scales with increasing task complexity.

4.3.2 Experimental Setup

We follow a standardized experimental design as outlined in Tong et al. (2023), utilizing the sololearn library (Da Costa et al., 2022). This library provides optimized setups specifically tailored for VICReg and Barlow Twins, ensuring a fair comparison.

Key aspects of the setup include:

- **Encoder**: ResNet-18 architecture
- **Projector**: Bi-layer network
- **Training Duration**: 1,000 epochs
- **Batch Size**: 256 images
- **SSL-EY Hyperparameters**: Adopted from Barlow Twins optimization

For SSL-EY, we intentionally use the hyperparameters that were optimized for Barlow Twins. This choice aims to demonstrate SSL-EY's robustness and adaptability rather than seeking to outperform existing methods through extensive tuning.

4.3.3 Evaluation Methodology

To assess the quality of the learned representations, we employ a linear probe approach:

1. Train the self-supervised models on the unlabeled training data.
2. Fix the encoder weights and train a linear classifier on top of the learned representations using the labeled training data.
3. Evaluate the classifier's performance on the validation set using two metrics:
    - **Top-1 Accuracy**: Percentage of samples where the highest probability prediction matches the true label.
    - **Top-5 Accuracy**: Percentage of samples where the true label is among the top 5 highest probability predictions.

This approach allows us to gauge how well the self-supervised methods capture meaningful and discriminative features from the unlabeled data. It is important to note that in our figures, we actually present log accuracy rather than raw accuracy. Log accuracy provides a more revealing representation of performance differences, especially when comparing models with high accuracy scores. This is because small improvements in raw accuracy at higher levels can represent substantial performance gains, which are more clearly visible on a logarithmic scale.

By comparing SSL-EY against Barlow Twins and VICReg under these controlled conditions, we aim to provide insights into its effectiveness as a self-supervised learning algorithm, particularly its ability to learn robust and transferable representations across different datasets and task complexities.

4.3.4 Observations

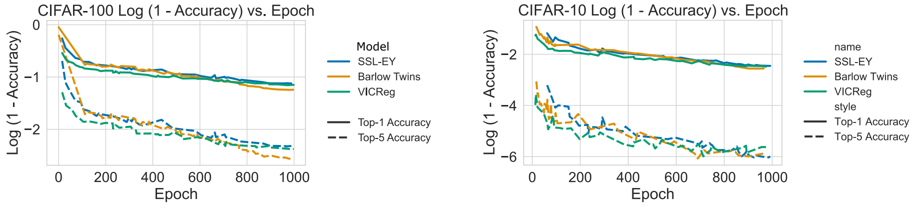
As Table 4.2 demonstrates, SSL-EY rivals Barlow Twins and VICReg, despite employing general hyperparameters as opposed to the latter's specifically optimized ones.

Table 4.2: Comparison of SSL method performance on CIFAR-10 and CIFAR-100 datasets.

Method	CIFAR-10 Top-1	CIFAR-10 Top-5	CIFAR-100 Top-1	CIFAR-100 Top-5
Barlow Twins	**92.1**	99.73	**71.38**	**92.32**
VICReg	91.68	99.66	68.56	90.76
SSL-EY	91.43	**99.75**	67.52	90.17

4.3.5 Model Convergence

Figure VI.10 illustrates the learning curve analysis for both CIFAR-10 and CIFAR-100 datasets.



(a) CIFAR-100

(b) CIFAR-10

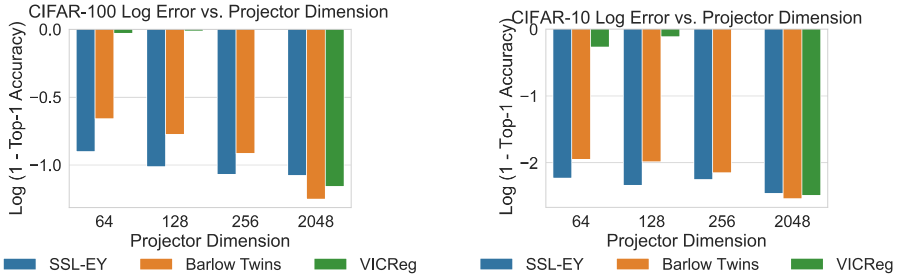
Figure VI.10: Learning curves for SSL-EY, Barlow Twins, and VICReg, depicting 1,000-epoch performance.

The learning curves in Figure VI.10 reveal a crucial insight: all compared SSL methods converge at remarkably slow rates, with hundreds of epochs required for substantial performance improvements. Even after 1,000 epochs, the models continue to learn. This observation highlights an important advantage of SSL-EY: its ability to achieve competitive performance without extensive hyperparameter tuning.

The performance variations at 1,000 epochs, as shown in Table 4.2, primarily stem from optimization noise, with convergence speeds being comparable among methods. This slow convergence underscores the importance of SSL-EY's robustness to hyperparameter choices, as it can achieve competitive results without the need for time-consuming fine-tuning.

4.3.6 Projector Size Variation and No-Projector Performance

We hypothesized that SSL-EY's architecture might exhibit robustness to projector size, potentially allowing for efficient performance even with smaller projectors or without one entirely. To investigate this, we conducted experiments varying the projector dimensions across SSL-EY, Barlow Twins, and VICReg, including a scenario with no projector. Figure VI.11 presents our findings on both CIFAR-100 and CIFAR-10 datasets.



(a) CIFAR-100

(b) CIFAR-10

Figure VI.11: Impact of projector size on SSL-EY, Barlow Twins, and VICReg performance for CIFAR-100 and CIFAR-10 datasets. The y-axis shows log error (inversely related to log accuracy), while the x-axis represents projector dimensions.

Figure VI.11 reveals that SSL-EY demonstrates remarkable stability across various projector sizes, maintaining high performance even as dimensions are reduced. In contrast, Barlow Twins and VICReg show more substantial performance degradation with smaller projector sizes, particularly on the more complex CIFAR-100 dataset.

To further investigate the impact of removing the projector entirely, we conducted additional experiments applying the linear probe directly to the encoder outputs. Table 4.3 presents these results alongside the projector-based performance for all three methods.

Table 4.3: Comparison of SSL method performance with and without projector on CIFAR-10 and CIFAR-100 datasets.

Method	Projector	CIFAR-10 Top-1	CIFAR-10 Top-5	CIFAR-100 Top-1	CIFAR-100 Top-5
Barlow Twins	Yes	**92.1**	99.73	**71.38**	**92.32**
Barlow Twins	No	89.99	99.21	63.51	86.99
VICReg	Yes	91.68	99.66	68.56	90.76
VICReg	No	**90.99**	99.46	63.82	86.39
SSL-EY	Yes	91.43	**99.75**	67.52	90.17
SSL-EY	No	90.98	99.69	**65.21**	**88.09**

The results in Table 4.3 provide several key insights:

- Both Barlow Twins and VICReg show a notable drop in performance when the projector is removed, particularly on the more complex CIFAR-100 dataset.
- SSL-EY demonstrates remarkable resilience, maintaining competitive performance even without a projector. This is especially evident on CIFAR-100, where SSL-EY's no-projector performance surpasses that of Barlow Twins and VICReg.
- On CIFAR-10, all methods show less dramatic changes when the projector is removed, likely due to the dataset's lower complexity.

These findings strongly support our initial hypothesis, highlighting SSL-EY's unique capacity to learn robust and efficient representations with minimal architectural overhead. The method's consistent performance across datasets of varying complexity and its resilience to projector removal underscore its versatility and potential for broad applicability in self-supervised learning tasks.

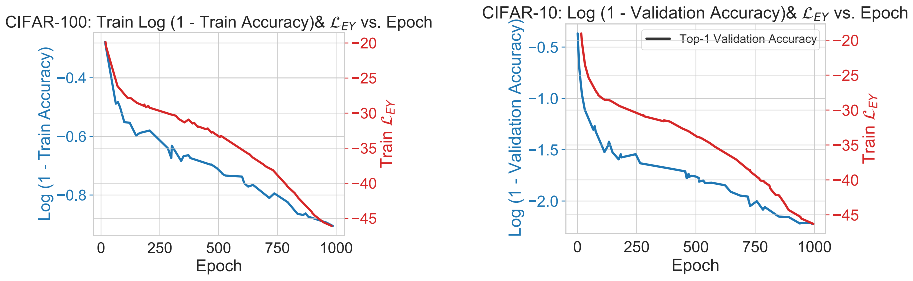
Despite the similarities in the underlying motivations across all three methods, our results suggest that a projector may not be a mandatory component for correlation-based models, at least in the context of our proposed method. By achieving competitive results with reduced architectural requirements, SSL-EY not only offers computational advantages but also suggests a more fundamental efficiency in its approach to representation learning.

4.3.7 $\mathcal{L}_{\text{EY}}$ as an Informative Metric

An important challenge in self-supervised learning (SSL) is tuning models to ensure their learned representations are robust and generalizable across various potential downstream tasks. Traditionally, SSL methods are often tuned using classification accuracy as a proxy metric. However, this approach risks overfitting the model to the specific classification task, potentially compromising its versatility for other applications. This practice contradicts the fundamental goal of SSL: to create representations that are adaptable to a wide range of unforeseen challenges.

To address this issue, we propose that our EY loss ($\mathcal{L}_{\text{EY}}$) could serve as a more appropriate surrogate metric for tuning and monitoring SSL models. By using $\mathcal{L}_{\text{EY}}$ instead of classification accuracy, we aim to maintain the model's generality while still providing a meaningful measure of representational quality. This approach aligns more closely with the core principles of SSL, potentially leading to more versatile and robust learned representations.

To explore this possibility, we investigated the relationship between EY loss and classification accuracy. Our findings, presented in Figure VI.12, offer compelling evidence for the potential of $\mathcal{L}_{\text{EY}}$ as an informative metric in SSL training.



(a) CIFAR-100

(b) CIFAR-10

Figure VI.12: Correlation between EY loss and classification accuracy for CIFAR-100 and CIFAR-10 datasets.

The correlation between EY loss and classification accuracy, shown in Figure VI.12, offers two key insights:

1. The strong relationship between EY loss and classification accuracy highlights the potential of maximizing canonical correlation as a pretext task in SSL.

2. The sustained increase in correlation over 1,000 epochs suggests that even with reduced projector dimensionality, SSL-EY does not reach full capacity,

indicating untapped potential in its representation capacity.

The evolution of the correlation, measured by $\mathcal{L}_{\text{EY}}$, suggests a new avenue for monitoring model training, potentially eliminating the need for a separate validation task. These results collectively demonstrate SSL-EY's robustness, efficiency, and potential for simplifying the self-supervised learning process across different dataset complexities.

This analysis not only supports the use of $\mathcal{L}_{\text{EY}}$ as a valuable monitoring tool for SSL training but also highlights its potential to streamline the development process. By providing a metric that correlates strongly with downstream performance without being tied to a specific task, $\mathcal{L}_{\text{EY}}$ could offer a more principled approach to model tuning in self-supervised learning contexts.

5 Discussion and Limitations

5.1 Discussion

This chapter introduces DCCA-EY and SSL-EY, novel approaches that address key limitations in Deep CCA and Self-Supervised Learning methods. These innovations offer substantial advantages in terms of scalability, convergence speed, and robustness to hyperparameter tuning, making them valuable tools for practitioners in representation learning.

One of the most notable strengths of DCCA-EY is its superior performance across various mini-batch sizes, as evidenced by our experiments on the Split MNIST and XRMB datasets. This scalability is crucial for handling large-scale datasets and diverse computational environments, a common challenge in modern machine learning applications. Moreover, both DCCA-EY and SSL-EY demonstrate faster convergence compared to existing methods, a particularly valuable trait in SSL tasks where training often requires a large number of epochs.

The reduced sensitivity to hyperparameter choices shown by our methods is another advantage. In practical applications, where extensive tuning can be prohibitively time-consuming or computationally expensive, this robustness can greatly streamline the deployment process. SSL-EY's ability to maintain performance with reduced or absent projectors suggests more efficient representation learning, potentially leading to simpler architectures and reduced computational requirements.

Furthermore, the observed correlation between EY loss and classification accuracy in SSL-EY provides a novel way to monitor training progress without relying on labeled validation data. This insight opens up new possibilities for unsupervised learning scenarios and could lead to more efficient training protocols.

5.2 Limitations

However, despite these advancements, our work is not without limitations. While our experiments were comprehensive, they were limited to specific datasets (Split MNIST, XRMB, CIFAR-10, CIFAR-100). Future research should explore performance on a broader range of datasets, including those from different domains such as natural language processing and time series data. This would provide a more complete picture of the methods' generalizability and robustness across diverse data types.

Additionally, while we have demonstrated improved scalability, further research is needed to assess performance on very large models and extremely high-dimensional datasets, which are increasingly common in cutting-edge machine learning applications. A more in-depth theoretical analysis of why DCCA-EY and SSL-EY perform well, particularly in stochastic settings, could provide valuable insights and guide future improvements.

Although our methods show improved convergence, a detailed study of their computational requirements compared to existing methods would be beneficial, especially for resource-constrained environments. Exploring how DCCA-EY and SSL-EY can be combined with other advanced techniques in deep learning, such as attention mechanisms or adaptive learning rates, could potentially lead to even better performance.

While our methods improve performance, further work on enhancing the interpretability of the learned representations could increase their utility in domains where model explainability is crucial. Our experiments, while extensive, were limited to 1,000 epochs. Investigating the long-term stability and performance of our methods over even longer training periods could provide insights into their asymptotic behavior.

Conclusion

In conclusion, DCCA-EY and SSL-EY represent advancements in Deep CCA and Self-Supervised Learning, offering improved scalability, faster convergence, and robustness to hyperparameter choices. These qualities make them valuable tools for researchers and practitioners working with complex, high-dimensional data. The strong performance of our methods, even without extensive tuning, suggests that they capture fundamental aspects of correlation and representation learning. This not only enhances their practical utility but also opens up new avenues for theoretical exploration in the fields of multiview learning and self-supervised learning.

```python
import torch
import torch.nn as nn

class UnifiedEY(nn.Module):
    def __init__(self, encoders, ssl_mode=False):
        super(UnifiedEY, self).__init__()
        self.ssl_mode = ssl_mode
        if ssl_mode:
            assert len(encoders) == 1, "SSL mode requires a single encoder"
            self.encoder = encoders[0]
        else:
            self.encoders = nn.ModuleList(encoders)

    def forward(self, Xs):
        if self.ssl_mode:
            Zs = [self.encoder(X) for X in Xs]
        else:
            Zs = [encoder(X) for encoder, X in zip(self.encoders, Xs)]
        return Zs

    def loss(self, Zs, alphas=[0, 0]):
        # Compute total between-view covariance matrix
        C = torch.zeros(Zs[0].shape[1], Zs[0].shape[1])
        for i in range(len(Zs)):
            for j in range(i + 1, len(Zs)):
                C += torch.matmul(Zs[i].T, Zs[j]) / Zs[i].shape[0]

        # Compute total within-view variance matrix
        V = torch.zeros(Zs[0].shape[1], Zs[0].shape[1])
        for i, alpha in enumerate(alphas):
            Vi = torch.matmul(Zs[i].T, Zs[i]) / Zs[i].shape[0]
            V += alpha * torch.eye(Vi.shape[0]) + (1 - alpha) * Vi

        # Compute loss
        loss = -2 * torch.trace(C) + torch.norm(V, p='fro') ** 2

        return loss
```

Listing 3: Unified PyTorch implementation for DCCA-EY and SSL-EY.

Chapter VII

CCA-Zoo: A collection of Regularized, Deep Learning-based, Kernel, and Probabilistic methods in a scikit-learn style framework

Contents

Preface

This work was published in the Journal of Open Source Software (Chapman and H.-T. Wang, 2021). I have been the lead developer of the `CCA-Zoo` package since its inception in 2020. All of the methods we have described in this thesis are implemented in `CCA-Zoo` and are immediately available for use by the research community.

1 Introduction

This chapter presents `CCA-Zoo`, a comprehensive Python library for multiview learning that was developed as a key contribution of this thesis. `CCA-Zoo` brings together a wide range of methods for canonical correlation analysis (CCA), partial least squares (PLS), and related techniques, providing efficient and user-friendly implementations that integrate seamlessly with the Python data science ecosystem.

The development of `CCA-Zoo` was motivated by the recognition that the lack of well-developed and widely available software has been a major barrier to the adoption and advancement of multiview learning methods, particularly in the Python community. While popular libraries like `scikit-learn` (Pedregosa et al., 2011) offer basic implementations of classical techniques like CCA and PLS, they lack support for many of the important extensions and variants that have been proposed in the literature to handle challenges such as high-dimensional data, non-linearity, sparsity, and deep learning.

`CCA-Zoo` aims to fill this gap by providing a unified framework for multiview learning that is both comprehensive and accessible. The library includes implementations of both classical and state-of-the-art methods, ranging from regularized and kernel-based extensions of CCA and PLS to modern deep learning and probabilistic approaches. These implementations are designed to be efficient, scalable, and easy to use, with a consistent API that follows the conventions of `scikit-learn`.

In addition to its core algorithms, `CCA-Zoo` provides a range of tools and utilities to support the entire multiview learning workflow, from data preprocessing and feature selection to model evaluation and visualization. The library also includes a collection of example datasets and pre-trained models, making it easy for users to get started and explore different techniques on real-world problems.

Throughout the development of this thesis, `CCA-Zoo` has played a central role as both a research tool and a means of disseminating our methodological contributions to the wider community. The experiments and case studies presented in the previous chapters have all relied on `CCA-Zoo` implementations, ensuring reproducibility and comparability of our results. At the same time, by releasing `CCA-Zoo` as an open-source library on GitHub and PyPi, we have enabled other researchers and practitioners to easily build upon and extend our work.

We also discuss the impact that `CCA-Zoo` has had so far, both within the context of this thesis and in the broader research community, and outline directions for future development and improvement. Our hope is that `CCA-Zoo` will serve as a valuable resource and catalyst for advancing the state-of-the-art in multiview learning, and for bridging the gap between methodological research and practical application.

2 Background: Software for Multiview Learning

The field of multiview learning has recently witnessed a surge of interest from the research community. This growth can be attributed to the increasing availability of multi-modal data across various domains, from bioinformatics to social media analysis, and the recognition that integrating multiple views can often lead to better insights and predictions than relying on a single perspective.

Traditionally, the development of multiview learning methods has been dominated by researchers in the statistical learning community, who have primarily relied on programming languages like R and MATLAB. These platforms have served as fertile ground for the creation and dissemination of many state-of-the-art algorithms.

However, this has created a challenge for researchers and practitioners who prefer to work in the Python programming language, which has become increasingly popular for machine learning tasks due to its simplicity, flexibility, and rich ecosystem of libraries. Python users have been faced with two suboptimal options: either port existing R or MATLAB implementations into Python, which can be a time-consuming and error-prone process requiring significant domain expertise, or make do with the limited set of multiview methods available in general-purpose Python libraries like scikit-learn (Pedregosa et al., 2011).

This fragmentation of the multiview learning landscape across different programming languages has created significant barriers to entry for Python users, potentially hindering the widespread adoption and application of these powerful techniques. Moreover, it has made it difficult for researchers to compare and benchmark different methods on a level playing field, as implementations may vary widely in terms of performance, scalability, and ease of use.

The CCA-Zoo package aims to address these challenges by providing a comprehensive and unified platform for multiview learning in Python. By offering a wide range of algorithms spanning both classical and state-of-the-art approaches, CCA-Zoo enables researchers and practitioners to easily explore and apply these techniques to their own data and problems, without the need for extensive domain expertise or cumbersome porting of code.

Through its scikit-learn compatible API, modular design, and efficient implementations, CCA-Zoo seamlessly integrates with the existing Python machine learning ecosystem, lowering the barriers to entry and accelerating the pace of research and application in multiview learning. By bringing together methods from different research communities and programming languages under a common framework, CCA-Zoo also facilitates fair and reproducible comparisons of different approaches, helping to advance our understanding of their strengths and limitations.

In the following sections, we will consider the design principles, key features, and implementation details of CCA-Zoo, showcasing how it can be used to streamline and enhance multiview learning workflows in Python.

3 Methods: Design and Implementation of CCA-Zoo

CCA-Zoo is designed to be a comprehensive and user-friendly library for multiview learning in Python. In this section, we describe the key design decisions and implementation details that underpin its functionality, flexibility, and performance. Figure VII.1 provides an overview of the library's structure and its integration with the wider Python machine learning ecosystem.

3.1 API Design and Scikit-Learn Compatibility

A central goal in the development of `CCA-Zoo` was to ensure maximum compatibility and interoperability with the existing Python machine learning ecosystem. To this end, we adopted the API design principles and conventions of the widely-used `scikit-learn` library (Pedregosa et al., 2011). `Scikit-learn` has become the de facto standard for machine learning in Python, thanks to its consistent, user-friendly API and its extensive collection of tools for data preprocessing, model selection, evaluation, and visualization.

By adhering to the `scikit-learn` API, `CCA-Zoo` inherits these benefits and ensures that users can seamlessly integrate multiview learning methods into their existing workflows. All estimators in `CCA-Zoo` follow the fit-transform pattern, where the `fit()` method learns model parameters from training data, and the `transform()` method applies the learned transformation to new data. Hyperparameters are specified as constructor arguments, allowing easy model creation and configuration.

This design choice not only makes `CCA-Zoo` intuitive and easy to use for anyone familiar with `scikit-learn`, but also enables the use of `scikit-learn`'s powerful model selection and evaluation tools, such as cross-validation and grid search, directly with `CCA-Zoo` estimators. Users can construct complex pipelines that include data preprocessing, feature selection, and multiview learning steps, all with a consistent, declarative syntax.

Figure VII.1 illustrates this integration, highlighting `CCA-Zoo`'s compatibility with key components of the Python machine learning stack.

3.2 Modular Architecture and Extensibility

Another key design principle of `CCA-Zoo` is modularity. The library is organized into distinct submodules, each focusing on a specific aspect of the multiview learning workflow:

- `datasets`: Classes for generating synthetic data and loading real-world datasets.
- `preprocessing`: Tools for data normalization, scaling, and dimensionality reduction.
- `model_selection`: Wrappers for `scikit-learn`'s cross-validation and hyperparameter tuning utilities, adapted for multiview settings.
- `linear`: Estimators for linear CCA, PLS, and their variants.

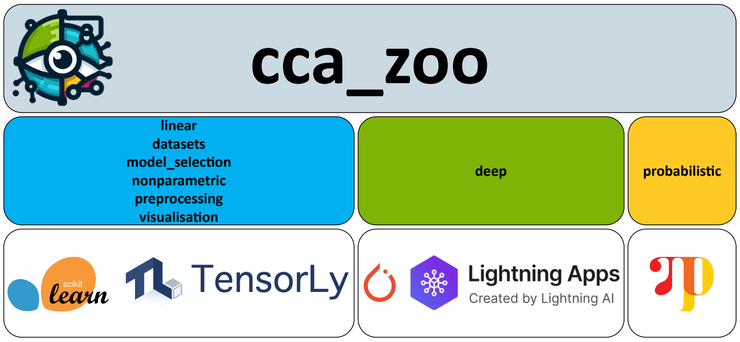


Figure VII.1: The CCA-Zoo compatibility map showcases integration with various machine learning packages. The deep learning module is built upon PyTorch and Lightning, reflecting their status as industry standards for neural network implementations. The probabilistic module employs NumPyro for its Bayesian inference capabilities, enhancing the application of probabilistic approaches in CCA.

- deep: Deep learning-based approaches, built on top of PyTorch (Paszke et al., 2019) and PyTorch Lightning (Falcon, 2019).
- probabilistic: Bayesian multiview learning methods, implemented with NumPyro (Phan, Pradhan, and Jankowiak, 2019), a probabilistic programming library built on top of JAX (DeepMind, 2020).
- visualization: Functions for visualizing model parameters, latent spaces, and performance metrics.

This modular structure makes the codebase more maintainable and easier to navigate. It also facilitates extensibility: new methods and features can be added to each submodule without affecting the rest of the library, as long as they adhere to the common API conventions.

Furthermore, the use of well-established libraries like PyTorch, PyTorch Lightning, and NumPyro for the deep learning and probabilistic modules ensures that CCA-Zoo can benefit from the latest advancements in these rapidly evolving fields. Developers can easily experiment with new architectures, loss functions, and inference techniques, while still leveraging the data handling and model evaluation capabilities of the core CCA-Zoo framework.

3.3 New methods included from this thesis

CCA-Zoo includes all of the methods we have described in this thesis, including the Elastic Net regularized CCA, methods for simulating data, as well as gradient descent based CCA and DCCA-EY. We also include all of the relevant baselines.

3.4 Flexibility and Ease of Use

CCA-Zoo is designed to be flexible and easy to use for a wide range of multiview learning tasks. The library provides a unified interface for working with both linear and nonlinear methods, unsupervised and semi-supervised settings, and two-view and multi-view scenarios.

The choice of default hyperparameters and architectural choices for deep learning models has been carefully considered to ensure good performance on a variety of datasets without the need for extensive tuning. At the same time, users have full control over these settings and can easily customize them for specific tasks.

CCA-Zoo also includes a range of utility functions and classes that simplify common tasks and help users avoid boilerplate code. For example, the datasets module provides a consistent interface for accessing and sampling from both synthetic and real-world datasets, handling data loading, splitting, and formatting behind the scenes.

Similarly, the model_selection module extends scikit-learn's cross-validation and grid search tools to handle the multi-view setting seamlessly. Users can perform model selection and hyperparameter tuning with just a few lines of code, without having to worry about the intricacies of indexing and reshaping views.

Listing 4 demonstrates this simplicity and flexibility, showing a complete workflow for training and evaluating a regularized CCA model with cross-validated hyperparameter selection:

This example showcases several key features of CCA-Zoo:

- The LatentVariableData class allows easy generation of synthetic multiview data with a specified number of features and latent dimensions.

- The rCCA class provides a regularized CCA estimator with a scikit-learn-compatible API, supporting both fit-transform and inverse transform operations.

- The GridSearchCV class wraps scikit-learn's grid search functionality, automatically handling the multi-view parameter grid and cross-validation

```python
from cca_zoo.datasets import LatentVariableData
from cca_zoo.linear import rCCA
from cca_zoo.model_selection import GridSearchCV
from cca_zoo.visualisation import SeparateRepresentationScatterDisplay

#Generate synthetic multi-view data
data = LatentVariableData(view_features=[10, 10], latent_dims=2)
X, Y = data.sample(n_samples=200, seed=42)

#Define hyperparameter grid
param_grid = {
'c': ([0.1, 0.3, 0.7, 0.9], [0.1, 0.3, 0.7, 0.9]),
}

#Perform cross-validated grid search
model = GridSearchCV(rCCA(latent_dimensions=2),
param_grid=param_grid,
cv=5).fit(X, Y)

#Visualize latent space
SeparateRepresentationScatterDisplay.from_estimator(model.best_estimator_)
```

Listing 4: A complete example of training and evaluating a regularized CCA model with CCA-Zoo.

splitting.

- The SeparateRepresentationScatterDisplay class provides a high-level interface for visualizing the learned latent space, with separate plots for each view.

By providing such high-level abstractions and adhering to familiar API conventions, CCA-Zoo aims to make multiview learning methods accessible to a wide audience, from seasoned machine learning practitioners to domain experts in fields like bioinformatics, computer vision, and natural language processing.

3.5 Performance and Scalability

In addition to ease of use and flexibility, CCA-Zoo is designed with performance and scalability in mind. The library is implemented in pure Python, with computationally intensive operations delegated to optimized libraries like NumPy (Harris et al., 2020), SciPy (Virtanen et al., 2020), and PyTorch.

For linear methods like CCA and PLS, CCA-Zoo leverages the randomized SVD and other matrix approximation techniques to efficiently handle high-dimensional data. These techniques allow the library to scale to datasets with millions of features and samples, without sacrificing accuracy or numerical stability.

In the deep learning module, CCA-Zoo takes advantage of PyTorch's GPU acceleration and automatic differentiation capabilities to enable fast training of complex models on large-scale datasets. The use of PyTorch Lightning further streamlines the training process, providing a high-level interface for distributed training, checkpointing, and logging.

For probabilistic methods, CCA-Zoo leverages the power of NumPyro and JAX to perform efficient variational inference and MCMC sampling on both CPUs and GPUs. The use of modern probabilistic programming techniques allows users to easily specify and train complex Bayesian models, while still benefiting from the performance and scalability of the underlying libraries.

CCA-Zoo's performance and scalability claims are backed by extensive benchmarking and testing on a variety of synthetic and real-world datasets. In Section 4, we present a selection of these experiments, comparing CCA-Zoo's performance to other popular multiview learning libraries and demonstrating its ability to handle large-scale, high-dimensional data.

3.6 Development and Maintenance

CCA-Zoo is developed as an open-source project, with its source code and documentation hosted on GitHub at https://github.com/jameschapman19/cca_zoo. The library follows modern software development best practices, including version control, continuous integration, and automated testing.

The development team is committed to maintaining and improving CCA-Zoo over the long term. This includes fixing bugs, adding new features and methods, and keeping dependencies up to date. The team also welcomes contributions from the community in the form of bug reports, feature requests, and pull requests.

To ensure the library's quality and reliability, CCA-Zoo includes a comprehensive test suite that covers all major functionality. These tests are automatically run on each commit and pull request, using continuous integration services like Travis CI and GitHub Actions. This helps catch regressions and ensures that new features are properly integrated and documented.

CCA-Zoo's documentation is another key aspect of its maintenance and development. The library includes extensive API documentation, generated automatically from docstrings using tools like Sphinx and Read the Docs. The documentation also includes user guides, tutorials, and examples to help users get started and make the most of the library's features.

In addition to the API documentation, `CCA-Zoo`'s GitHub repository includes a wiki and issue tracker where users can find additional information, ask questions, and report bugs. The development team is responsive to user feedback and strives to address issues in a timely manner.

By adhering to these development and maintenance practices, `CCA-Zoo` aims to provide a stable, reliable, and well-documented library that can serve as a foundation for multiview learning research and applications for years to come.

In summary, the key design decisions and implementation details of `CCA-Zoo` are:

- Adherence to the `scikit-learn` API for maximum compatibility and ease of use.
- Modular architecture for maintainability and extensibility.
- Flexibility and unified interface for both linear and deep learning methods.
- Use of optimized libraries and techniques for performance and scalability.
- Open-source development with modern software engineering practices.
- Comprehensive documentation and user support.

These choices reflect `CCA-Zoo`'s goal of providing a powerful yet accessible toolkit for multiview learning in Python, suitable for both research and practical applications.

4 Experiments and Results

Throughout this thesis, the `CCA-Zoo` package has been used extensively for conducting experiments and evaluating the proposed multiview learning methods. The library's comprehensive set of tools and its seamless integration with the Python data science ecosystem have greatly facilitated the implementation and assessment of these methods on a wide range of datasets and tasks.

In this section, we showcase the performance and versatility of `CCA-Zoo` through a series of benchmarking experiments. These experiments not only demonstrate the efficiency of the library's implementations but also highlight its ability to handle high-dimensional data, a crucial requirement in many real-world applications such as bioinformatics and natural language processing.

4.1 Benchmarking Setup

To assess the computational efficiency of `CCA-Zoo`, we compared its performance against the widely-used `scikit-learn` library. We focused on two fundamental multiview learning methods: Canonical Correlation Analysis (CCA) and Partial Least Squares (PLS). The experiments were conducted on synthetic datasets of varying dimensionality to evaluate the scalability of the implementations.

The datasets were generated as random matrices with a fixed number of samples (100) and a varying number of features (50, 100, 200, 400, and 800) for each view. The number of latent dimensions was set to 10 for both CCA and PLS. To obtain reliable performance metrics, each experiment was repeated 10 times, and the average execution time was reported.

The benchmarking experiments were performed using the following software versions:

- `CCA-Zoo` (version: 2.4.0)
- `Scikit-learn` (version: 1.3.0)

All experiments were run on a machine with an Intel Core i7-9700K CPU (3.60GHz) and 32GB of RAM, running Ubuntu 20.04.

4.2 Canonical Correlation Analysis

Figure VII.2 presents the comparison of execution times between `CCA-Zoo` and `scikit-learn` for CCA. Across all tested dimensionalities, `CCA-Zoo` demonstrates competitive performance, with execution times comparable to or better than those of `scikit-learn`.

The efficiency of `CCA-Zoo`'s CCA implementation can be attributed to its use of the principal component space for computing the canonical correlations. By first projecting the data onto a lower-dimensional space defined by the leading principal components, `CCA-Zoo` reduces the computational burden associated with high-dimensional covariance matrices, resulting in faster execution times without sacrificing accuracy.

This performance advantage is particularly relevant in real-world applications, where the number of features often greatly exceeds the number of samples. In such

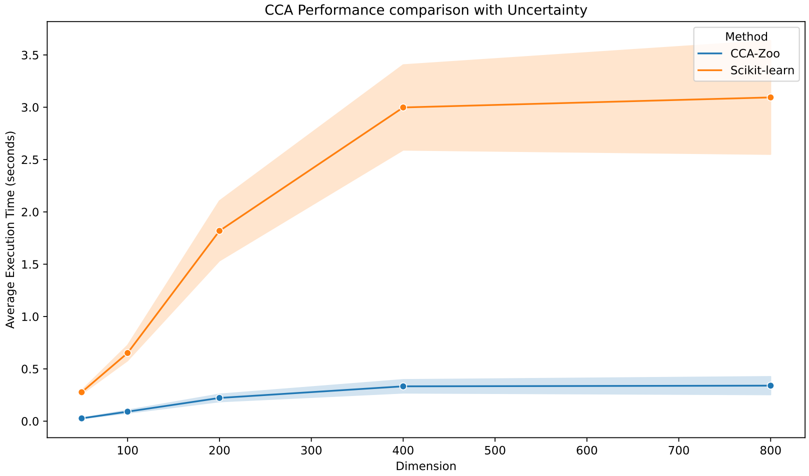


Figure VII.2: Performance comparison for CCA methods

scenarios, CCA-Zoo's ability to efficiently handle high-dimensional data can lead to significant time savings and enable the analysis of larger datasets.

4.3 Partial Least Squares

Figure VII.3 shows the execution time comparison for PLS. Similar to the CCA results, CCA-Zoo exhibits a robust performance profile, with execution times that are competitive with those of scikit-learn across all tested dimensionalities.

The competitive performance of CCA-Zoo's PLS implementation can be attributed to its use of efficient algorithms and data structures, such as the NIPALS algorithm and the deflation scheme for computing the latent components. These optimizations allow CCA-Zoo to scale well with increasing data dimensionality, making it a suitable choice for a wide range of applications.

4.4 Real-World Applications

In addition to the synthetic benchmarking experiments, CCA-Zoo has been used throughout this thesis to evaluate the proposed multiview learning methods on various real-world datasets. These datasets span multiple domains, including neuroscience, bioinformatics and computer vision and pose diverse challenges in terms of dimensionality, sparsity, and noise.

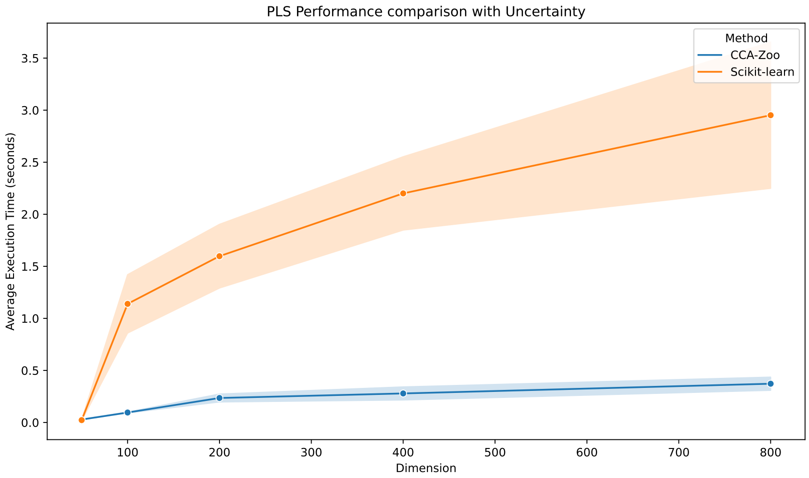


Figure VII.3: Performance comparison for PLS methods

5 Discussion and Limitations

5.1 Contributions and Impact

The development of `CCA-Zoo` represents a significant contribution to the field of multiview learning, particularly within the Python ecosystem. By providing a comprehensive and user-friendly library that integrates seamlessly with existing tools like scikit-learn and PyTorch, `CCA-Zoo` has the potential to greatly accelerate research and application of multiview methods.

By being fully compatible with `sklearn` pipelines, we believe that `CCA-Zoo` also ensures that deconfounding parameters can be estimated as part of the model fitting process rather than separate steps which introduce a subtle form of leakage. While we did not include this in the package itself to avoid feature creep, we recommend using sklearn pipelines for deconfounding such as Raamana (2020) which I was able to contribute to in a small way. This allows users to automatically identify and remove confounding factors from their data, improving the interpretability and generalization of multiview models. This is particularly important in applications like genomics and neuroimaging, where confounding variables can significantly impact the results.

Throughout this thesis, `CCA-Zoo` has played a central role in enabling the empirical studies and method development presented in the previous chapters. The library's efficient implementations of both classical and state-of-the-art multiview algorithms allowed us to conduct extensive experiments on real-world datasets, comparing the performance of different methods and gaining new insights into their behavior. Moreover, the modular design of `CCA-Zoo` facilitated rapid prototyping and testing of novel extensions and refinements to existing techniques.

Beyond its use in this thesis, `CCA-Zoo` has already begun to have an impact in the wider research community. The library has been well-received on GitHub, with over 150 stars and 30 forks to date, and has been downloaded nearly 500 times per month from the Python Package Index. Several published papers and ongoing projects in fields ranging from genomics to neuroscience have used CCA-Zoo, demonstrating its potential to enable new discoveries and applications.

5.2 Limitations and Future Work

While `CCA-Zoo` provides a solid foundation for multiview learning in Python, there are certainly areas where it could be improved and extended. One current limitation is the lack of GPU acceleration for some of the more computationally intensive methods, which could hamper their scalability to massive datasets. In future versions, we plan to leverage libraries like cupy to enable seamless GPU support.

Another direction for future work is to expand the library's functionality to encompass an even wider range of multiview learning paradigms, such as multi-modal deep learning, multi-view clustering, and multi-view matrix factorization. By providing a unified interface to these diverse approaches, `CCA-Zoo` could serve as a powerful toolkit for exploring and combining different perspectives on data.

Finally, we are committed to the ongoing maintenance and development of `CCA-Zoo` as an open-source project. We welcome contributions from the community in the form of bug reports, feature requests, documentation improvements, and code contributions. By engaging with users and incorporating their feedback, we hope to continuously refine and enhance the library to better serve the needs of multiview learning researchers and practitioners.

5.3 Conclusion

In conclusion, `CCA-Zoo` fills an important gap in the Python ecosystem by providing a comprehensive, efficient, and user-friendly library for multiview learning. Through its extensive catalog of classical and modern multiview methods, seamless integration with popular machine learning tools, and flexible API, `CCA-Zoo` enables researchers and practitioners to easily explore and apply these powerful techniques to their own data and problems.

The development of `CCA-Zoo` has been a central contribution of this thesis, underpinning many of the empirical studies and methodological advances presented in earlier chapters. By making the library open-source and freely available to the community, we hope to accelerate progress in multiview learning and promote reproducible, extensible research.

Looking ahead, we see ample opportunities to expand and refine CCA-Zoo, in collaboration with its growing base of users and contributors. Through sustained development and community engagement, we believe `CCA-Zoo` has the potential to become an indispensable tool in the multiview learning toolkit, enabling new discoveries and applications across a wide range of domains.

Chapter VIII

Conclusion

This chapter provides a summary of the findings of this thesis, discusses their implications, and outlines potential directions for future work.

1 Summary of Contributions

1.1 Regularisation of CCA Models: A Flexible Framework based on Alternating Least Squares

This chapter presented the FRALS framework for CCA, addressing challenges in analyzing large-scale neuroimaging datasets from projects such as the Human Connectome Project and the Alzheimer's Disease Neuroimaging Initiative. Incorporating structured priors through regularization, particularly the elastic net penalty, FRALS enhanced the interpretability and generalizability of CCA models. This method, documented in the work presented at the OHBM (James Chapman, 2023), has been effective in uncovering significant brain-behavior associations, showing superior out-of-sample performance compared to traditional methods.

1.2 Insights From Generating Simulated Data for CCA

This chapter contributed to the debate on the interpretation of model weights versus loadings in CCA. By generating high-dimensional simulated data and categorizing methods into explicit and implicit latent variable models, the chapter highlights the robustness of loadings to columnwise transformations in data matrices, a feature not shared with weights. The simulated data strategies formed part of the analysis in Mihalik, Chapman, Adams, Winter, Ferreira, Shawe-Taylor, Mourao-Miranda, et al. (2022) and influenced the analysis in Adams et al. (2024).

1.3 Efficient Algorithms for the CCA Family: Unconstrained Losses with Unbiased Gradients

Focusing on scaling challenges for CCA and PLS in the context of large-scale biomedical datasets like the UK Biobank, this chapter introduces a new gradient descent algorithm tailored for generalized eigenvalue problems. The methods developed, informed by publications (Chapman, Aguila, and Wells, 2022; Chapman, Wells, and Aguila, 2024; Chapman and Wells, 2023), enable the application of multiview CCA and PLS to datasets with extensive dimensions and complex structures.

1.4 Deep CCA and Self-Supervised Learning

This chapter introduces a novel formulation of Deep CCA optimized for the stochastic minibatch setting and proposes SSL-EY, a new competitive SSL method. Grounded in findings from (Chapman and Wells, 2023) and (Chapman, Wells, and Aguila, 2024), the chapter demonstrates the robustness of these methods against hyperparameter sensitivity and elucidates connections between CCA-based SSL methods and other contemporary SSL approaches.

1.5 CCA-Zoo: A collection of Regularized, Deep Learning-based, Kernel, and Probabilistic methods in a scikit-learn style framework

Presenting `CCA-Zoo`, a Python library that consolidates and enhances the accessibility of multiview learning methods, this chapter details the development and capabilities of the library, which implements a variety of CCA, PLS, and related techniques. As detailed in (Chapman and H.-T. Wang, 2021), `CCA-Zoo` addresses gaps in existing software offerings and facilitates broader adoption and innovation within the research community.

In summary, we have demonstrated novel ways to introduce structured priors into CCA models, developed efficient algorithms for large-scale CCA, extended CCA to deep learning, and provide a unified interface for various CCA methods. Finally, we have made software implementations of these methods available to the research community through the `CCA-Zoo` package which have already been well-received by the community.

2 Limitations and Challenges

It is important to note that while this thesis presents significant methodological advancements and software contributions, the new applied results are somewhat limited. This limitation was largely due to unforeseen circumstances, including significant bureaucratic challenges in accessing the UK Biobank dataset and the global impact of the COVID-19 pandemic. These factors restricted our ability to conduct extensive applied research as initially planned. However, this limitation has been partially offset by the broad adoption and application of our software tools by other researchers in various fields, extending the impact of our work beyond our own direct applications.

3 Future Work

3.1 Applications

3.1.1 Large-Scale Neuroimaging Datasets

While the applications presented in this thesis, particularly the UK Biobank analysis in Chapter VI, have demonstrated the potential of our methods, there is still vast untapped potential in applying these techniques to even larger and more diverse datasets. The ABCD dataset, for instance, offers a rich source of multimodal data that could benefit from the regularized and scalable CCA methods developed in this thesis. Preliminary results on this dataset have shown promise, and we believe that further exploration will yield valuable insights into brain development and its associated factors.

3.1.2 Wearable Devices

The rise of wearable devices and the proliferation of biometric data present new opportunities for applying multiview learning techniques to personal health monitoring. By integrating data streams from devices such as smartwatches, continuous glucose monitors, and sleep trackers, we can gain insights into an individual's physical and mental well-being that were previously inaccessible. I strongly believe that the development of interpretable and scalable methods for analyzing these diverse data sources will be crucial for unlocking the full potential of wearable technology in personalized healthcare.

3.2 Methods

3.2.1 Proximal Gradient Descent for Regularized CCA

Preliminary experiments suggest that the proximal gradient descent approach discussed in chapter V is much faster than existing methods, making it a promising direction for future research. We anticipate that this methodology will significantly enhance the scalability and applicability of CCA, PCA, and PLS in the era of big data and complex regularization schemes.

4 Closing Remarks

My PhD journey has been a fascinating exploration of the world of multiview learning, with Canonical Correlation Analysis at its core. What began as an endeavor to apply deep learning to uncover brain-behavior associations evolved into a multifaceted exploration, leading to the development of scalable algorithms for linear CCA and the investigation of connections between CCA and self-supervised learning.

Confronting the scalability bottleneck head-on led me to develop these efficient algorithms, pushing the boundaries of CCA and PLS methods. This exploration also unveiled intriguing connections between CCA and self-supervised learning, bringing the journey full circle.

Witnessing the rapid growth and evolution of multiview learning during my PhD has been incredibly stimulating. From the emergence of powerful multimodal language models to the increasing adoption of self-supervised learning techniques, being part of this dynamic field has been a privilege. This journey has transformed me as a researcher, honing my skills in algorithm development, software engineering, and bridging theoretical concepts with practical applications.

One of the most rewarding aspects of this work has been its impact, particularly through the CCA-Zoo package. Seeing researchers across various fields use these tools to tackle diverse problems has been immensely gratifying, underscoring the importance of accessible method implementations. Moreover, the potential applications of these methods in areas like large-scale neuroimaging studies and wearable device data analysis hint at broader impacts on our understanding of human health and behavior.

Looking ahead, I believe the future of multiview learning holds immense promise. The integration of deep learning with CCA and the application of these methods to ever-larger and more diverse datasets open exciting avenues for research. Particularly promising are the applications to large-scale neuroimaging datasets and the analysis of multimodal data from wearable devices for personalized healthcare.

Appendices

Appendix A

HCP and ADNI Loadings

This appendix builds upon the results presented in Chapter III, where we introduced a method to regularize CCA using structured priors on model weights, demonstrated with Human Connectome Project (HCP) and Alzheimer's Disease Neuroimaging Initiative (ADNI) data. In light of the insights gained from Chapter IV, which examined the relationship between loadings and weights in CCA using simulated data, we revisit the HCP and ADNI results to further explore the interpretability of the models.

1 Human Connectome Project (HCP) Data

This section presents the loadings for the HCP data, which were obtained using the models described in Chapter III. The loadings are visualized using chord diagrams for each model. The chord diagrams show the top 8 positive and negative brain loadings for each model. The loadings are ordered by their absolute value, with the largest loadings at the top of the diagram. The chord diagrams provide a visual representation of the relationship between the brain regions and the latent variables in each model. The loadings are color-coded to indicate the strength and direction of the relationship between the brain regions and the latent variables. Positive loadings are shown in blue, and negative loadings are shown in red. The width of the chords indicates the strength of the relationship between the brain regions and the latent variables. The chord diagrams provide a concise and intuitive way to compare the loadings across the different models and identify patterns in the relationships between the brain regions and the latent variables.

1.1 Brain Connectivity Weights and Loadings

Figure A.1 shows chord diagrams of the top 8 positive and negative brain loadings for each model.

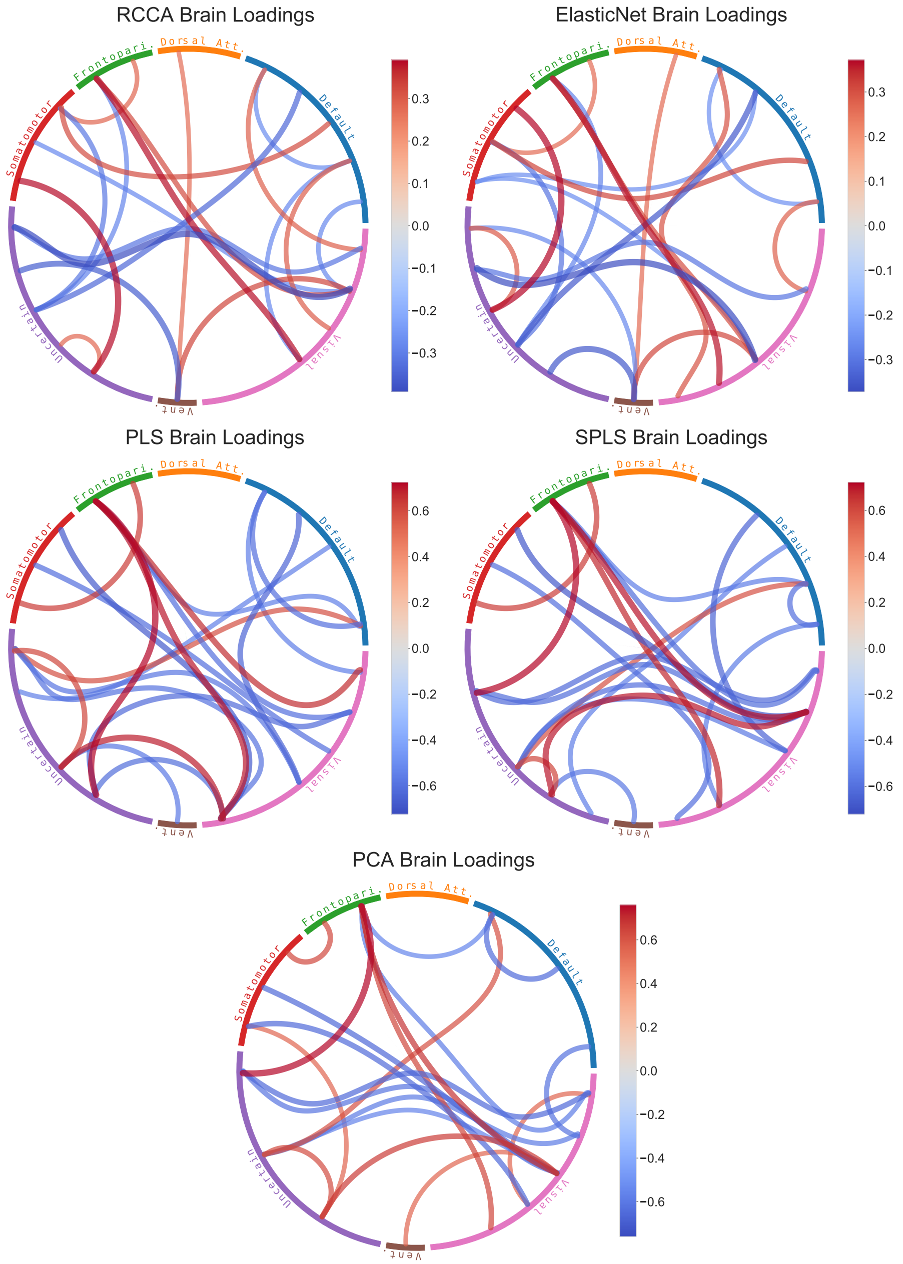


Figure A.1: Chord diagrams of the top 8 positive and negative brain loadings for each model.

2 Alzheimer’s Disease Neuroimaging Initiative (ADNI) Data

This section presents the loadings for the ADNI data, which were obtained using the models described in Chapter III. The loadings are visualized using statistical maps for each model. The statistical maps show the brain structure loadings for each model. The loadings are color-coded to indicate the strength and direction of the relationship between the brain structures and the latent variables. Positive loadings are shown in blue, and negative loadings are shown in red. The statistical maps provide a visual representation of the relationship between the brain structures and the latent variables in each model. The maps allow us to compare the loadings across the different models and identify patterns in the relationships between the brain structures and the latent variables.

2.1 Brain Structure Weights and Loadings

Figure A.2 shows statistical maps of brain structure loadings and weights for each model.

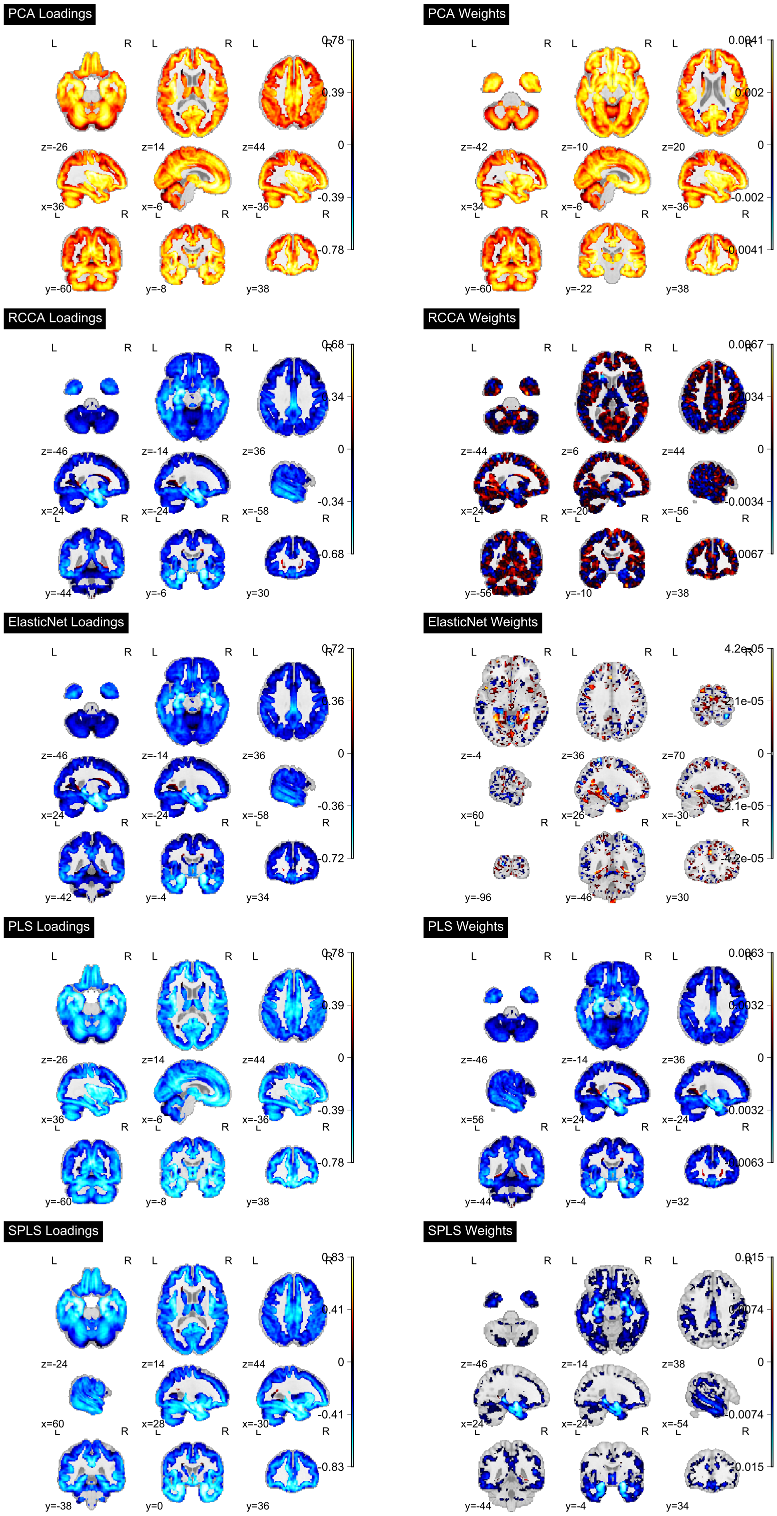


Figure A.2: Statistical maps of brain structure loadings and weights for each model.

References

Adams, R. A. et al. (2024). “Voxel-wise multivariate analysis of brain-psychosocial associations in adolescents reveals six latent dimensions of cognition and psychopathology”. In: *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*. ISSN: 2451-9022. DOI: `https://doi.org/10.1016/j.bpsc.2024.03.006`. URL: `https://www.sciencedirect.com/science/article/pii/S2451902224000855`.

Ali, A., E. Dobriban, and R. Tibshirani (2020). “The implicit regularization of stochastic gradient flow for least squares”. In: *International conference on machine learning*. PMLR, pp. 233–244.

Alpert, M. I. and R. A. Peterson (1972). “On the interpretation of canonical analysis”. In: *Journal of marketing Research* 9.2, pp. 187–192.

Andrew, G. et al. (2013). “Deep canonical correlation analysis”. In: *International conference on machine learning*. PMLR, pp. 1247–1255.

Arora, R., A. Cotter, et al. (2012). “Stochastic optimization for PCA and PLS”. In: *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, pp. 861–868.

Arora, R., P. Mianjy, and T. Marinov (2016). “Stochastic optimization for multiview representation learning using partial least squares”. In: *International Conference on Machine Learning*. PMLR, pp. 1786–1794.

Ashburner, J. et al. (2014). “SPM12 manual”. In: *Wellcome Trust Centre for Neuroimaging, London, UK* 2464.4.

Bach, F. R. and M. I. Jordan (2005). “A probabilistic interpretation of canonical correlation analysis”. In: URL: `https://statistics.berkeley.edu/sites/default/files/tech-reports/688.pdf`.

Balakrishnama, S. and A. Ganapathiraju (1998). “Linear discriminant analysis-a brief tutorial”. In: *Institute for Signal and information Processing* 18.1998, pp. 1–8.

Baldassarre, L., J. Mourao-Miranda, and M. Pontil (2012). “Structured sparsity models for brain decoding from fMRI data”. In: *2012 Second International Workshop on Pattern Recognition in NeuroImaging*. IEEE, pp. 5–8.

Balestriero, R., M. Ibrahim, et al. (2023). “A Cookbook of Self-Supervised Learning”. In: *arXiv preprint arXiv:2304.12210*.

Balestriero, R. and Y. LeCun (2022). “Contrastive and non-contrastive self-supervised learning recover global and local spectral embedding methods”. In: *arXiv preprint arXiv:2205.11508*.

Bardes, A., J. Ponce, and Y. LeCun (2021). “Vicreg: Variance-invariance-covariance regularization for self-supervised learning”. In: *arXiv preprint arXiv:2105.04906*.

Barlow, H. B. et al. (1961). “Possible principles underlying the transformation of sensory messages”. In: *Sensory communication* 1.01, pp. 217–233.

Benton, A. et al. (2017). “Deep generalized canonical correlation analysis”. In: *arXiv preprint arXiv:1702.02519*.

Bilenko, N. Y. and J. L. Gallant (2016). “Pyrcca: regularized kernel canonical correlation analysis in python and its applications to neuroimaging”. In: *Frontiers in neuroinformatics* 10, p. 49. DOI: 10.3389/fninf.2016.00049.

Biobank, U. (2014). “About UK Biobank”. In: *Available at h ttps://www. ukbiobank. ac. uk/a bout-biobank-uk*.

Bogdan, P. C. et al. (2023). “ConnSearch: A framework for functional connectivity analysis designed for interpretability and effectiveness at limited sample sizes”. In: *NeuroImage* 278, p. 120274.

Bogdan, R. et al. (2017). “Imaging genetics and genomics in psychiatry: a critical review of progress and potential”. In: *Biological psychiatry* 82.3, pp. 165–175.

Borga, M. (1998). “Learning Multidimensional Signal Processing”. eng. Publisher: Linköping University Electronic Press. PhD thesis. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-54341 (visited on 10/13/2022).

Boyd, S. et al. (2011). “Distributed optimization and statistical learning via the alternating direction method of multipliers”. In: *Foundations and Trends® in Machine learning* 3.1, pp. 1–122.

Button, K. S. et al. (2013). “Power failure: why small sample size undermines the reliability of neuroscience”. In: *Nature reviews neuroscience* 14.5, pp. 365–376.

Bzdok, D., T. E. Nichols, and S. M. Smith (2019). “Towards algorithmic analytics for large-scale datasets”. In: *Nature Machine Intelligence* 1.7, pp. 296–306.

Bzdok, D. and B. T. Yeo (2017). “Inference in the age of big data: Future perspectives on neuroscience”. In: *Neuroimage* 155, pp. 549–564.

Carroll, J. D. (1968). “Generalization of canonical correlation analysis to three or more sets of variables”. In: *Proceedings of the 76th annual convention of the American Psychological Association*. Vol. 3. Washington, DC, pp. 227–228.

Chang, X., T. Xiang, and T. M. Hospedales (2018). “Scalable and effective deep CCA via soft decorrelation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1488–1497.

Chapman, J., A. L. Aguila, and L. Wells (2022). “A Generalized EigenGame with Extensions to Multiview Representation Learning”. In: *arXiv preprint arXiv:2211.11323*.

Chapman, J. and H.-T. Wang (2021). “CCA-Zoo: A collection of Regularized, Deep Learning based, Kernel, and Probabilistic CCA methods in a scikit-learn style framework”. In: *Journal of Open Source Software* 6.68, p. 3823.

Chapman, J. and L. Wells (2023). “CCA with Shared Weights for Self-Supervised Learning”. In: *NeurIPS 2023 Workshop: Self-Supervised Learning - Theory and Practice*. URL: https://openreview.net/forum?id=7rYseRZ7Z3.

Chapman, J., L. Wells, and A. L. Aguila (2024). *Unconstrained Stochastic CCA: Unifying Multiview and Self-Supervised Learning*.

Chen, M.-S. et al. (2022). “Representation learning in multi-view clustering: A literature review”. In: *Data Science and Engineering* 7.3, pp. 225–241.

Chen, M. et al. (2013). “Sparse CCA via precision adjusted iterative thresholding”. In: *arXiv preprint arXiv:1311.6186*.

Chen, Z. et al. (2019). “On constrained nonconvex stochastic optimization: A case study for generalized eigenvalue decomposition”. In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 916–925.

Chi, E. C. et al. (2013). “Imaging genetics via sparse canonical correlation analysis”. In: *2013 IEEE 10th International Symposium on Biomedical Imaging*, pp. 740–743. DOI: 10.1109/ISBI.2013.6556581.

Chun, H. and S. Keleş (2010). “Sparse partial least squares regression for simultaneous dimension reduction and variable selection”. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 72.1, pp. 3–25.

Cruciani, F. et al. (2022). “What PLS can still do for Imaging Genetics in Alzheimer’s disease”. In: *2022 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*. IEEE, pp. 1–4.

Cuingnet, R. et al. (2012). “Spatial and anatomical regularization of SVM: a general framework for neuroimaging data”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.3, pp. 682–696.

Curth, A., A. Jeffares, and M. van der Schaar (2023). “A U-turn on Double Descent: Rethinking Parameter Counting in Statistical Learning”. In: *arXiv preprint arXiv:2310.18988*.

Cybenko, G. (1989). “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4, pp. 303–314.

Da Costa, V. G. T. et al. (2022). “solo-learn: A Library of Self-supervised Methods for Visual Representation Learning.” In: *J. Mach. Learn. Res.* 23.56, pp. 1–6.

De Pierrefeu, A. et al. (2017). “Structured sparse principal components analysis with the TV-elastic net penalty”. In: *IEEE transactions on medical imaging* 37.2, pp. 396–407.

DeepMind (2020). *The DeepMind JAX Ecosystem*. URL: http://github.com/deepmind.

Demontis, D. et al. (Feb. 2023). “Genome-wide analyses of ADHD identify 27 risk loci, refine the genetic architecture and implicate several cognitive domains”. en. In: *Nat. Genet.* 55.2, pp. 198–208.

Deng, L. et al. (2021). “Sparse PLS-based method for overlapping metabolite set enrichment analysis”. In: *Journal of proteome research* 20.6, pp. 3204–3213.

Dinga, R. et al. (2019). “Evaluating the evidence for biotypes of depression: Methodological replication and extension of”. In: *NeuroImage: Clinical* 22, p. 101796.

Dohmatob, E. D. et al. (2014). “Benchmarking solvers for TV-L1 least-squares and logistic regression in brain imaging”. In: *2014 International Workshop on Pattern Recognition in Neuroimaging*. IEEE, pp. 1–4.

Drysdale, A. T. et al. (2017). “Resting-state connectivity biomarkers define neurophysiological subtypes of depression”. In: *Nature medicine* 23.1, pp. 28–38.

Duin, R. (n.d.). *Multiple Features*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5HC70.

Eckart, C. and G. Young (1936). “The approximation of one matrix by another of lower rank”. In: *Psychometrika* 1.3, pp. 211–218.

Engl, H. W., M. Hanke, and A. Neubauer (1996). *Regularization of inverse problems*. Vol. 375. Springer Science & Business Media.

Ermolov, A. et al. (2021). “Whitening for self-supervised representation learning”. In: *International Conference on Machine Learning*. PMLR, pp. 3015–3024.

Euesden, J., C. M. Lewis, and P. F. O’Reilly (Dec. 2014). “PRSice: Polygenic Risk Score software”. In: *Bioinformatics* 31.9, pp. 1466–1468. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btu848. eprint: https://academic.oup.com/bioinformatics/article-pdf/31/9/1466/50306478/bioinformatics_31_9_1466.pdf. URL: https://doi.org/10.1093/bioinformatics/btu848.

Falcon, W. A. (2019). “Pytorch lightning”. In: *GitHub* 3.

Ferreira, F. S. et al. (2022). “A hierarchical Bayesian model to find brain-behaviour associations in incomplete data sets”. In: *NeuroImage* 249, p. 118854.

Fischl, B. (Aug. 2012). “FreeSurfer”. en. In: *Neuroimage* 62.2, pp. 774–781.

Fjell, A. M. et al. (2009). “High consistency of regional cortical thinning in aging across multiple samples”. In: *Cerebral cortex* 19.9, pp. 2001–2012.

Folstein, M. F., S. E. Folstein, and P. R. McHugh (1975). ““Mini-mental state”: a practical method for grading the cognitive state of patients for the clinician”. In: *Journal of psychiatric research* 12.3, pp. 189–198.

Fu, X. et al. (2017). “Scalable and flexible Max-Var generalized canonical correlation analysis via alternating optimization”. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 5855–5859.

Galton, F. (1907). “Vox populi”. In: *Nature* 75.1949, pp. 450–451.

Ge, R. et al. (2016). “Efficient algorithms for large-scale generalized eigenvector computation and canonical correlation analysis”. In: *International Conference on Machine Learning*. PMLR, pp. 2741–2750.

Gemp, I., C. Chen, and B. McWilliams (2022). “The Generalized Eigenvalue Problem as a Nash Equilibrium”. In: *arXiv preprint arXiv:2206.04993*.

Gemp, I., B. McWilliams, et al. (2021). *EigenGame Unloaded: When playing games is better than optimizing*. arXiv: 2102.04152 [stat.ML].

Gemp, I. M. et al. (2020). “EigenGame: PCA as a Nash Equilibrium”. In: *CoRR* abs/2010.00554. arXiv: 2010.00554. URL: https://arxiv.org/abs/2010.00554.

Genon, S., S. B. Eickhoff, and S. Kharabian (2022). “Linking interindividual variability in brain structure to behaviour”. In: *Nature Reviews Neuroscience* 23.5, pp. 307–318.

Ghojogh, B., F. Karray, and M. Crowley (2019). “Eigenvalue and generalized eigenvalue problems: Tutorial”. In: *arXiv preprint arXiv:1903.11240*.

Golub, G. H. and C. F. Van Loan (2013). *Matrix computations*. JHU press.

Golub, G. H. and H. Zha (1995). “The canonical correlations of matrix pairs and their numerical computation”. In: *Linear algebra for signal processing*. Springer, pp. 27–49. DOI: 10.1007/978-1-4612-4228-4_3.

Gonen, M. and E. Alpaydin (2011). “Multiple kernel learning algorithms”. In: *The Journal of Machine Learning Research* 12, pp. 2211–2268.

Goyal, P. et al. (2019). “Scaling and benchmarking self-supervised visual representation learning”. In: *Proceedings of the ieee/cvf International Conference on computer vision*, pp. 6391–6400.

Greenacre, M. et al. (2022). “Principal component analysis”. In: *Nature Reviews Methods Primers* 2.1, p. 100.

Grosenick, L. et al. (2013). “Interpretable whole-brain prediction analysis with Graph-Net”. In: *NeuroImage* 72, pp. 304–321.

Gu, F. and H. Wu (2018). “Simultaneous canonical correlation analysis with invariant canonical loadings”. In: *Behaviormetrika* 45, pp. 111–132.

Guo, W., J. Wang, and S. Wang (2019). “Deep multimodal representation learning: A survey”. In: *Ieee Access* 7, pp. 63373–63394.

Hardoon, D. R., J. Mourao-Miranda, et al. (2007). “Unsupervised analysis of fMRI data using kernel canonical correlation”. In: *NeuroImage* 37.4, pp. 1250–1259.

Hardoon, D. R., S. Szedmak, and J. Shawe-Taylor (2004). “Canonical correlation analysis: An overview with application to learning methods”. In: *Neural computation* 16.12, pp. 2639–2664. DOI: 10.1162/0899766042321814.

Harris, C. R. et al. (2020). “Array programming with NumPy”. In: *Nature* 585.7825, pp. 357–362.

Hastie, T. et al. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer.

Helmer, M. et al. (2020). “On stability of Canonical Correlation Analysis and Partial Least Squares with application to brain-behavior associations”. In: *bioRxiv*.

Höskuldsson, A. (1988). “PLS regression methods”. In: *Journal of chemometrics* 2.3, pp. 211–228.

Hotelling, H. (1933). “Analysis of a complex of statistical variables into principal components.” In: *Journal of educational psychology* 24.6, p. 417.

— (1935). “Canonical correlation analysis (cca)”. In: *Journal of Educational Psychology*, p. 10.

— (1992). “Relations between two sets of variates”. In: *Breakthroughs in statistics*. Springer, pp. 162–190. DOI: 10.2307/2333955.

ICML (2023). *ICML 2023*. URL: https://icml.cc/Conferences/2023/Test-of-Time (visited on 09/21/2023).

International League Against Epilepsy Consortium on Complex Epilepsies (Dec. 2018). “Genome-wide mega-analysis identifies 16 loci and highlights diverse biological mechanisms in the common epilepsies”. en. In: *Nat. Commun.* 9.1, p. 5269.

Jack Jr, C. R. et al. (2008). “The Alzheimer’s disease neuroimaging initiative (ADNI): MRI methods”. In: *Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine* 27.4, pp. 685–691.

James Chapman Janaina Mourao-Miranda, J. S.-T. (2023). *A Framework for Regularised Canonical Correlation Analysis by Alternating Least Squares*.

Johnstone, I. M. (2001). “On the distribution of the largest eigenvalue in principal components analysis”. In: *The Annals of statistics* 29.2, pp. 295–327.

Kanai, R. and G. Rees (2011). “The structural basis of inter-individual differences in human behaviour and cognition”. In: *Nature Reviews Neuroscience* 12.4, pp. 231–242.

Kanatsoulis, C. I. et al. (2018). “Structured SUMCOR multiview canonical correlation analysis for large-scale data”. In: *IEEE Transactions on Signal Processing* 67.2, pp. 306–319.

Kettenring, J. R. (1971). “Canonical analysis of several sets of variables”. In: *Biometrika* 58.3, pp. 433–451.

Kingma, D. P. and J. Ba (2014). “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980*.

Kingma, D. P. and M. Welling (2013). “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114*.

Klami, A. et al. (2014). “Group factor analysis”. In: *IEEE transactions on neural networks and learning systems* 26.9, pp. 2136–2147.

Krishnan, A. et al. (2011). “Partial Least Squares (PLS) methods for neuroimaging: a tutorial and review”. In: *Neuroimage* 56.2, pp. 455–475.

Lambert, J. C. et al. (Dec. 2013). “Meta-analysis of 74,046 individuals identifies 11 new susceptibility loci for Alzheimer’s disease”. en. In: *Nat. Genet.* 45.12, pp. 1452–1458.

Lawry Aguila, A., J. Chapman, and A. Altmann (2023). “Multi-modal Variational Autoencoders for Normative Modelling Across Multiple Imaging Modalities”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer Nature Switzerland Cham, pp. 425–434.

Lê Cao, K.-A. et al. (2008). “A sparse PLS for variable selection when integrating omics data”. In: *Statistical applications in genetics and molecular biology* 7.1.

Lindenbaum, O. et al. (2021). “L0-sparse canonical correlation analysis”. In: *International Conference on Learning Representations*.

Liu, J. and V. D. Calhoun (2014). “A review of multivariate analyses in imaging genetics”. In: *Frontiers in neuroinformatics* 8, p. 29.

Liu, Z. et al. (2022). “Improved Interpretability of Brain-Behavior CCA With Domain-Driven Dimension Reduction”. In: *Frontiers in Neuroscience* 16, p. 851827.

Luo, C. et al. (2018). “Cosine normalization: Using cosine similarity instead of dot product in neural networks”. In: *Artificial Neural Networks and Machine*

Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27. Springer, pp. 382–391.

Lyu, Q. et al. (2021). "Understanding latent correlation-based multiview learning and self-supervision: An identifiability perspective". In: *arXiv preprint arXiv:2106.07115*.

Ma, Z., Y. Lu, and D. Foster (2015). "Finding linear structure in large datasets with scalable canonical correlation analysis". In: *International conference on machine learning*. PMLR, pp. 169–178.

Mackay, D. J. C. (1998). "Introduction to monte carlo methods". In: *Learning in graphical models*. Springer, pp. 175–204.

Mackey, L. (2008). "Deflation methods for sparse PCA". In: *Advances in neural information processing systems* 21.

Mai, Q. and X. Zhang (2019). "An iterative penalized least squares approach to sparse canonical correlation analysis". In: *Biometrics* 75.3, pp. 734–744. DOI: 10.1111/biom.13043.

Matkovic, A. et al. (2023). "The contribution of diverse and stable functional connectivity edges to brain-behavior associations". In: *bioRxiv*, pp. 2023–11.

Matkovič, A. et al. (2023). "Static and dynamic fMRI-derived functional connectomes represent largely similar information". In: *Network Neuroscience* 7.4, pp. 1266–1301.

McIntosh, A. R. (2021). "Comparison of Canonical Correlation and Partial Least Squares analyses of simulated and empirical data". In: *arXiv preprint arXiv:2107.06867*.

Meng, Z., R. Chakraborty, and V. Singh (2021). "An Online Riemannian PCA for Stochastic Canonical Correlation Analysis". In: *Advances in Neural Information Processing Systems* 34, pp. 14056–14068.

Meredith, W. (1964). "Canonical correlations with fallible data". In: *Psychometrika* 29.1, pp. 55–65.

Michel, V. et al. (2011). "Total variation regularization for fMRI-based prediction of behavior". In: *IEEE transactions on medical imaging* 30.7, pp. 1328–1340.

Mihalik, A., J. Chapman, R. A. Adams, N. R. Winter, F. S. Ferreira, J. Shawe-Taylor, J. Mourao-Miranda, et al. (2022). "Canonical correlation analysis and partial least squares for identifying brain-behaviour associations: a tutorial and a comparative study". In: *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*.

Mihalik, A., J. Chapman, R. A. Adams, N. R. Winter, F. S. Ferreira, J. Shawe-Taylor, and J. Mourão-Miranda (Aug. 2022). "Canonical Correlation Analysis and Partial Least Squares for identifying brain-behaviour associations: a tutorial and a comparative study". en. In: *Biological Psychiatry: Cognitive Neuroscience and Neu-*

roimaging. ISSN: 2451-9022. DOI: 10.1016/j.bpsc.2022.07.012. URL: https://www.sciencedirect.com/science/article/pii/S2451902222001859 (visited on 08/29/2022).

Mihalik, A., F. S. Ferreira, M. Moutoussis, et al. (2020). “Multiple holdouts with stability: Improving the generalizability of machine learning analyses of brain–behavior relationships”. In: *Biological psychiatry* 87.4, pp. 368–376.

Mihalik, A., F. S. Ferreira, M. J. Rosa, et al. (2019). “Brain-behaviour modes of covariation in healthy and clinically depressed young people”. In: *Scientific reports* 9.1, pp. 1–11.

Mills-Curran, W. C. (1988). “Calculation of eigenvector derivatives for structures with repeated eigenvalues”. In: *AIAA journal* 26.7, pp. 867–871.

Miranda, L. et al. (2021). “Systematic review of functional MRI applications for psychiatric disease subtyping”. In: *Frontiers in Psychiatry* 12, p. 665536.

Mirsky, L. (1960). “Symmetric gauge functions and unitarily invariant norms”. In: *The quarterly journal of mathematics* 11.1, pp. 50–59.

Monteiro, J. M. et al. (2016). “A multiple hold-out framework for Sparse Partial Least Squares”. In: *Journal of neuroscience methods* 271, pp. 182–194.

Mullins, N. et al. (June 2021). “Genome-wide association study of more than 40,000 bipolar disorder cases provides new insights into the underlying biology”. en. In: *Nat. Genet.* 53.6, pp. 817–829.

Nalls, M. A. et al. (Dec. 2019). “Identification of novel risk loci, causal insights, and heritable risk for Parkinson’s disease: a meta-analysis of genome-wide association studies”. en. In: *Lancet Neurol.* 18.12, pp. 1091–1102.

Ngiam, J. et al. (2011). “Multimodal deep learning”. In: *ICML*.

Nguyen, N. D. and D. Wang (2020). “Multiview learning for understanding functional multiomics”. In: *PLoS computational biology* 16.4, e1007677.

Oja, E. (1982). “Simplified neuron model as a principal component analyzer”. In: *Journal of mathematical biology* 15.3, pp. 267–273.

OpenAI (2021). *ChatGPT: A Large-Scale Generative Model for Open-Domain Chat*. https://github.com/openai/gpt-3.

Park, S., E. Ceulemans, and K. Van Deun (2023). “A critical assessment of sparse PCA (research): why (one should acknowledge that) weights are not loadings”. In: *Behavior Research Methods*, pp. 1–20.

Parkhomenko, E., D. Tritchler, and J. Beyene (2009). “Sparse canonical correlation analysis with application to genomic data integration”. In: *Statistical applications in genetics and molecular biology* 8.1, pp. 1–34. DOI: 10.2202/1544-6115.1406.

Paszke, A. et al. (2019). “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in neural information processing systems* 32, pp. 8026–8037.

Pearl, J. (2009). *Causality*. Cambridge university press.

Pedregosa, F. et al. (2011). “Scikit-learn: Machine learning in Python”. In: *the Journal of machine Learning research* 12, pp. 2825–2830.

Phan, D., N. Pradhan, and M. Jankowiak (2019). “Composable effects for flexible and accelerated probabilistic programming in NumPyro”. In: *arXiv preprint arXiv:1912.11554*.

Price, A. L. et al. (2006). “Principal components analysis corrects for stratification in genome-wide association studies”. In: *Nature genetics* 38.8, pp. 904–909.

Purcell, S. et al. (Sept. 2007). “PLINK: a tool set for whole-genome association and population-based linkage analyses”. en. In: *Am. J. Hum. Genet.* 81.3, pp. 559–575.

Qi, J. and J. Tejedor (2016). “Deep multi-view representation learning for multimodal features of the schizophrenia and schizo-affective disorder”. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 952–956.

Raamana, P. R. (Mar. 2020). *Conquering confounds and covariates in machine learning with the python library confounds*. Version 0.1.1. DOI: `10.5281/zenodo.3701528`. URL: `https://doi.org/10.5281/zenodo.3701528`.

Reichenbach, H. (1956). *The direction of time*. Vol. 65. Univ of California Press.

Rheenen, W. van et al. (Dec. 2021). “Common and rare variant association analyses in amyotrophic lateral sclerosis identify 15 risk loci with distinct genetic architectures and neuron-specific biology”. en. In: *Nat. Genet.* 53.12, pp. 1636–1648.

Riffenburgh, R. H. (1957). “Linear discriminant analysis”. PhD thesis. Virginia Polytechnic Institute.

Rosipal, R. and N. Krämer (2005). “Overview and recent advances in partial least squares”. In: *International Statistical and Optimization Perspectives Workshop" Subspace, Latent Structure and Feature Selection"*. Springer, pp. 34–51.

Ruigrok, A. N. et al. (2014). “A meta-analysis of sex differences in human brain structure”. In: *Neuroscience & Biobehavioral Reviews* 39, pp. 34–50.

Rypma, B. and M. D’Esposito (2001). “Age-related changes in brain–behaviour relationships: Evidence from event-related functional MRI studies”. In: *European Journal of Cognitive Psychology* 13.1-2, pp. 235–256.

Sameni, R., C. Jutten, and M. B. Shamsollahi (2009). “A deflation procedure for subspace decomposition”. In: *IEEE Transactions on Signal Processing* 58.4, pp. 2363–2374.

Sanger, T. D. (1989). “Optimal unsupervised learning in a single-layer linear feedforward neural network”. In: *Neural networks* 2.6, pp. 459–473.

Sansone, E. and R. Manhaeve (2022). “GEDI: GEnerative and DIscriminative Training for Self-Supervised Learning”. In: *arXiv preprint arXiv:2212.13425*.

Smith, S. M. and T. E. Nichols (2018). “Statistical Challenges in “Big Data” Human Neuroimaging”. In: *Neuron* 97.2, pp. 263–268. ISSN: 0896-6273. DOI: `https://doi.org/10.1016/j.neuron.2017.12.018`. URL: `https://www.sciencedirect.com/science/article/pii/S0896627317311418`.

Smith, S. M., T. E. Nichols, et al. (2015). “A positive-negative mode of population covariation links brain connectivity, demographics and behavior”. In: *Nature neuroscience* 18.11, pp. 1565–1567.

Snoek, C. G. et al. (2005). “Mediamill: Exploring news video archives based on learned semantics”. In: *Proceedings of the 13th annual ACM international conference on Multimedia*, pp. 225–226.

Somandepalli, K. et al. (2019). “Multimodal representation learning using deep multiset canonical correlation”. In: *arXiv preprint arXiv:1904.01775*.

Sudlow, C. et al. (2015). “UK biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age”. In: *PLoS medicine* 12.3, e1001779.

Sun, L., S. Ji, and J. Ye (2008). “A least squares formulation for canonical correlation analysis”. In: *Proceedings of the 25th international conference on Machine learning*, pp. 1024–1031.

Suo, X. et al. (2017). “Sparse canonical correlation analysis”. In: *arXiv preprint arXiv:1705.10865*.

Tenenhaus, A. and M. Tenenhaus (2011). “Regularized generalized canonical correlation analysis”. In: *Psychometrika* 76.2, p. 257. DOI: `10.1007/s11336-011-9206-8`.

Tipping, M. E. and C. M. Bishop (1999). “Probabilistic principal component analysis”. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 61.3, pp. 611–622.

Tong, S. et al. (2023). “EMP-SSL: Towards Self-Supervised Learning in One Training Epoch”. In: *arXiv preprint arXiv:2304.03977*.

Townsend, F., J. Chapman, and J. Cole (Nov. 2023). *florencejt/fusilli: Fusilli v1.0.0*. Version v1.0.0. DOI: 10.5281/zenodo.10228564. URL: https://doi.org/10.5281/zenodo.10228564.

Trubetskoy, V. et al. (Apr. 2022). “Mapping genomic loci implicates genes and synaptic biology in schizophrenia”. en. In: *Nature* 604.7906, pp. 502–508.

Tuzhilina, E., L. Tozzi, and T. Hastie (2023). “Canonical correlation analysis in high dimensions with structured regularization”. In: *Statistical modelling* 23.3, pp. 203–227.

Uurtio, V. et al. (2017). “A tutorial on canonical correlation methods”. In: *ACM Computing Surveys (CSUR)* 50.6, pp. 1–33.

Van Essen, D. C. et al. (2013). “The WU-Minn human connectome project: an overview”. In: *Neuroimage* 80, pp. 62–79.

Vapnik, V. (1999). *The nature of statistical learning theory*. Springer science & business media.

Vaswani, A. et al. (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30.

Vinod, H. D. (1976). “Canonical ridge and econometrics of joint production”. In: *Journal of econometrics* 4.2, pp. 147–166.

Virtanen, P. et al. (2020). “SciPy 1.0: fundamental algorithms for scientific computing in Python”. In: *Nature methods* 17.3, pp. 261–272.

Voevodskaya, O. et al. (2014). “The effects of intracranial volume adjustment approaches on multiple regional MRI volumes in healthy aging and Alzheimer’s disease”. In: *Frontiers in aging neuroscience* 6, p. 264.

Waaijenborg, S., P. C. V. de Witt Hamer, and A. H. Zwinderman (2008). “Quantifying the association between gene expressions and DNA-markers by penalized canonical correlation analysis”. In: *Statistical applications in genetics and molecular biology* 7.1.

Wang, H.-T. et al. (2018). “Finding the needle in high-dimensional haystack: A tutorial on canonical correlation analysis”. In: *arXiv preprint arXiv:1812.02598*.

— (2020). “Finding the needle in a high-dimensional haystack: Canonical correlation analysis for neuroscientists”. In: *NeuroImage* 216, p. 116745.

Wang, W., R. Arora, K. Livescu, and J. A. Bilmes (2015). “Unsupervised learning of acoustic features via deep canonical correlation analysis”. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 4590–4594.

Wang, W., R. Arora, K. Livescu, and N. Srebro (2015). “Stochastic optimization for deep CCA via nonlinear orthogonal iterations”. In: *2015 53rd Annual Aller-*

ton Conference on Communication, Control, and Computing (Allerton). IEEE, pp. 688–695.

Wilms, I. and C. Croux (2015). “Sparse canonical correlation analysis from a predictive point of view”. In: *Biometrical Journal* 57.5, pp. 834–851.

Winkler, A. M. et al. (2020). “Permutation inference for Canonical Correlation Analysis”. In: *arXiv preprint arXiv:2002.10046*.

Witten, D. M., R. Tibshirani, and T. Hastie (2009). “A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis”. In: *Biostatistics* 10.3, pp. 515–534.

Wold, H. (1975). “Path models with latent variables: The NIPALS approach”. In: *Quantitative sociology*. Elsevier, pp. 307–357.

Yao, Y., L. Rosasco, and A. Caponnetto (2007). “On early stopping in gradient descent learning”. In: *Constructive Approximation* 26, pp. 289–315.

Yeo, B. T. et al. (2011). “The organization of the human cerebral cortex estimated by intrinsic functional connectivity”. In: *Journal of neurophysiology*.

Zbontar, J. et al. (2021). “Barlow twins: Self-supervised learning via redundancy reduction”. In: *arXiv preprint arXiv:2103.03230*.

Zhuang, X., Z. Yang, and D. Cordes (2020). “A technical review of canonical correlation analysis for neuroscience applications”. In: *Human Brain Mapping* 41.13, pp. 3807–3833.

Zong, Y., O. Mac Aodha, and T. Hospedales (2023). “Self-Supervised Multimodal Learning: A Survey”. In: *arXiv preprint arXiv:2304.01008*.

Zou, H., T. Hastie, and R. Tibshirani (2006). “Sparse principal component analysis”. In: *Journal of computational and graphical statistics* 15.2, pp. 265–286.

Zou, H. and L. Xue (2018). “A selective overview of sparse principal component analysis”. In: *Proceedings of the IEEE* 106.8, pp. 1311–1320.