Advances in Probabilistic Generative Models: Normalizing Flows, Multi-View Learning, and Linear Dynamical Systems

by

Mahdi Karami

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Statistical Machine Learning

Department of Computing Science

University of Alberta

Abstract

This thesis considers some aspects of generative models including my contributions in deep probabilistic generative architectures and linear dynamical systems.

First, some advances in deep probabilistic generative models are contributed. Flow-based generative modelling is an emerging and highly applicable method to construct complex probability density. Herein, I investigate a set of invertible convolutional flows based on the circular and symmetric convolutions with efficient Jacobian determinant computation and inverse mapping (deconvolution) in $\mathcal{O}(N \log N)$ time. Further, an analytic approach to designing nonlinear elementwise bijectors is proposed that induces special properties in the intermediate layers, by implicitly introducing specific regularizers in the loss. It is demonstrated that these transforms allow more effective normalizing flow models to be developed for generative image models.

In the second part, a deep generative framework is expanded to multi-view learning. This model is composed of a linear probabilistic multi-view layer in the latent space in conjunction with deep generative networks as observation models where the variations of each view is captured by a shared latent representation and a set of view-specific factors. To approximate the posterior distribution of the latent probabilistic multi-view layer, a variational inference approach is developed that results in a scalable algorithm for training deep generative multi-view neural networks. Empirical studies confirm that the proposed deep

generative multi-view model can efficiently integrate the relationship between multiple views.

Finally, the thesis considers maximum likelihood estimation of linear dynamical systems (LDS) and develops an optimization based strategy for recovering the latent states and transition parameters. Key to the approach is a two-view reformulation of maximum likelihood estimation for linear dynamical systems that enables the use of global optimization algorithms for matrix factorization. It is shown that the proposed estimation strategy outperforms widely-used identification algorithms such as subspace identification methods, both in terms of accuracy and runtime.

Preface

The research included in this thesis is my original work. Chapter 2 has been presented as the following conference and workshop papers:

- Mahdi Karami, Dale Schuurmans, Jascha Sohl-Dickstein, Laurent Dinh, and Daniel Duckworth. Invertible convolutional flow. In *Advances in Neural Information Processing Systems*, pages 5636–5646, 2019a

- Mahdi Karami, Dale Schuurmans, Jascha Sohl-Dickstein, Laurent Dinh, and Daniel Duckworth. Symmetric convolutional flow. In *Workshop on Invertible Neural Nets and Normalizing Flows(INNF), ICML 2019*, 2019b

- Mahdi Karami, Laurent Dinh, Daniel Duckworth, Jascha Sohl-Dickstein, and Dale Schuurmans. Generative convolutional flow for density estimation. In *Workshop on Bayesian Deep Learning NeurIPS 2018*, 2018

Chapter 3 is basis of a conference submission currently under review which is also available in:

- Mahdi Karami and Dale Schuurmans. Variational inference for deep probabilistic canonical correlation analysis. *arXiv preprint arXiv:2003.04292*, 2020a

Chapter 4 has been published as the following conference paper:

- Mahdi Karami, Martha White, Dale Schuurmans, and Csaba Szepesvári. Multi-view matrix factorization for linear dynamical system estimation. In *Advances in Neural Information Processing Systems*, pages 7092–7101, 2017

The work presented in appendix C has been accepted in the *Data and Machine Learning Advances with Multiple Views workshop in ECML/PKDD* and published as:

- Mahdi Karami. Deep generative multi-view learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 465–477. Springer, 2019

Acknowledgements

First and foremost, I would like to thank my supervisor Dale Schuurmans for his precious guidance, constructive discussions, constant encouragement, and support in research and life. It was my privilege to take his exciting grad courses and continue my research with him.

I would like to extend my appreciation to my thesis committee Csaba Szepesvari for his inspiring attitude and friendly comments, Martha White for her helpful discussion during my thesis and Nilanjan Ray for his time and questions throughout my defense. Moreover, I sincerely thank Aaron Courville for his detailed reading and valuable comments on my thesis. I am also grateful to Ying Tsui whose thoughtful guidance and support was central during my transition to the CS department.

My appreciation goes out to each of my colleagues, collaborators in the department of CS and ECE, and friends in Edmonton and Delft. Special thanks to Amir Alizad whose wonderful friendship and the joyeful time we spent together were the cherry on the cake.

I would like to express my countless thanks to my family: my beloved mother and father for their unconditional love and patience, my siblings and my parents-in-law whose support and kindness pushed me forward.

Finally, I cannot give enough gratitude to my beloved wife, Fatemeh. Thank you for your genuine love and incentive encouragement during this long journey; you and my dear Minoo made it wonderful and sweet for me.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

2D	Two Dimensional
3D	Three Dimensional
ACC	Clustering Accuracy Rate
AE	Autoencoder
AR	Auto-regressive
ARI	Adjusted Rand Index
CCA	Canonical Correlation Analysis
CNN	Convolutional Neural Network
CONF	Convolutional Coupling Flow
DCT	Discrete Cosine transform
DCT-II	Discrete Cosine transform of type two
DFT	Discrete Fourier transform
DGMV	Deep Generative Multi-View
DNN	Deep Neural Network
DSC	Deep Subspace Clustering
DTT	Discrete Trigonometric transform
ELBO	Evidence Lower Bound
EM	Expectation- Maximization
FA	Factor Analysis
FISTA	Fast Iterative Shrinkage-Thresholding Algorithm (accelerated proximal gradient algorithm)
GAN	Generative Adversarial Network
GCG	Generalized Conditional Gradient
GFA	Group Factor Analysis

GOF	Goodness Of Fit
HMM	Hidden Markov Model
IAF	Inverse Autoregressive Flow
ISTA	Iterative Shrinkage-Thresholding Algorithm (proximal gradient algorithm)
KL	KullbackLeibler (divergence)
LDA	Linear Discriminant Analysis
LDS	Linear Dynamical System
LDS-DV	Linear Dynamical System - Duo View
MBFA	Multi-battery Factor Analysis
ML	Maximum Likelihood
MLE	Maximum Likelihood Estimation
MSE	Mean Squared Error
MV	Multi-view
N4SID	Subspace State Space System Identification algorithm
NF	Normalizing Flow
NMI	Normalizing Mutual Information
NMSE	Normalized Mean Square Error
NN	Neural Network
PCA	Principal Component Analysis
PCCA	Principal Canonical Correlation Analysis
PEM	Prediction Error Method
S-Log	Symmetric Logarithmic shape gate
SGD	Symmetric Gradient Descent algorithm
VAE	Variational Autoencoder
VCCA	Variational Canonical Correlation Analysis
VPCCA	Variational Probabilistic Canonical Correlation Analysis

Chapter 1

Introduction

The field of data science has experienced growing demand for the ability to process and understand high-dimensional and highly structured data, requiring algorithms that can handle increasingly complex problems. The success of machine learning in the era of big data not only relies on growing computational power but also on advances in models and training algorithms. Building highly flexible yet scalable learning models, and designing associated training algorithms, has driven significant research in the past decade. New representation learning methods are at the core of this movement [Bengio et al., 2013]. In contrast to feature engineering, which depends highly on human expertise, *representation learning* aims to develop algorithms that are sufficiently powerful to automatically identify meaningful feature representations and useful implicit structure that captures the essence of data. Generative models are among the most successful schemes in this context. The main theme of this thesis is to consider new advances in generative models, particularly from a probabilistic point of view.

Depending on the context, generative models can refer to different notions. In supervised learning, where features $\boldsymbol{x}$ and targets, $\boldsymbol{y}$ (also known as labels in classification), are provided, generative models can be defined in contrast to the discriminative models. In this context, the generative models describe the statistical model of all observed variables, *i.e.* learning the joint distribution over both features and targets $p(\mathbf{x}, \mathbf{y})$, while the discriminative models specify

the conditional probability $p(\mathbf{y}|\mathbf{x})$, such as linear or logistic regression, or, in non-probabilistic view, learns a forward mapping from features to targets such as support vector machines. Some instances of supervised generative models are Naive Bayes and linear discriminant analysis (LDA) [Ngiam et al., 2011].

In unsupervised learning, the notion of generative models is extended to a broader family of learning algorithms. Unsupervised probabilistic generative models, explicitly or implicitly, estimate the underlying distribution of data, $p_{data}(\mathbf{x})$, with an approximate model density, $p_{model}(\mathbf{x})$, using large pools of unlabeled training samples. Therefore, these models offer a mechanism to generate new realistic looking samples according to the estimated data distribution, hence are referred to as generative models. Many classical unsupervised algorithms such as PCA, CCA or Auto-encoders (AEs) can be given generative probabilistic interpretations, elaborated in chapter 3. Besides their classical application, due to the success and flexibility in modeling high dimensional and highly complex problems, generative unsupervised models have been recently adopted in a wide range of applications, including, but not limited to, generating new contents or synthesizing image or voice [Radford et al., 2015, Oord et al., 2016a], denoising [Ballé et al., 2015] inpainting or matrix completion [Oord et al., 2016b, Yeh et al., 2017], image or audio super-resolution [Ledig et al., 2017, Kuleshov et al., 2017] extracting semantically meaningful and interpretable latent representation [Kingma and Welling, 2013, Dinh et al., 2016], dimensionality reduction or data compression [Gregor et al., 2016, Ballé et al., 2018, Theis et al., 2017, Tschannen et al., 2018, Townsend et al., 2019] and modeling time series [Karami et al., 2017, Luo et al., 2018, Kumar et al., 2019, Babaeizadeh et al., 2017]; as well as applications in other fields such as astronomy [Fergus et al., 2014, Regier et al., 2015] chemistry [Gómez-Bombarelli et al., 2018] and neuroscience [Wiltschko et al., 2015, Linderman et al., 2016].

1.1 Background and Context

The main trends in probabilistic generative models that provide some of the relevant context for this thesis are briefly reviewed.

Autoregressive density estimation is an approach to probabilistic modeling that decomposes the joint density of visible variables, $p(\mathbf{x})$, into a product of one-dimensional conditionals according to the chain rule of probability, $p(\mathbf{x}) = \prod_i p(x_i|\mathbf{x}_{1:i-1})$. Modeling the conditionals by neural networks provides highly flexible learning algorithms that together with techniques such as masked autoencoders [Germain et al., 2015] enables fast training on parallel computational hardware. Some recent works in this domain, *e.g.* PixelCNN [Oord et al., 2016c] PixelRNN [Oord et al., 2016b], demonstrate the merit of the approach in achieving state-of-the-art in density estimation and image generation. Since no latent variable modeling is involved, tractable conditionals of these classes of *fully visible* generative models offers exact log-likelihood evaluation and scalable training, but on the other hand the absence of latent variables limits the interpretablity and applicability of such models in downstream tasks that rely on rich latent representation modeling. Also, their sequential nature makes sample generation from these models non-parallelizable and computationally inefficient, especially for high-dimensional and real-time applications such as video or speech synthesis. In addition, these algorithms are sensitive to sequence of factorizations whereby the appropriate choice of ordering can play an important role in the expressiveness of the learned model.

In contrast to fully visible models, the *partially visible* generative models associate a latent variable for each observation that represents the data in a simple and meaningful way. These models, also known as *deep latent variable models*, enable inference of the latent representations from data as natural features that explain the underlying factors of high-dimensional observations, and hence alleviate the difficulty of learning in downstream tasks. These benefits come at a price. Maximum likelihood learning for complex partially visible models, such as deep generative networks, is hard, often intractable in its standard form, which has motivated a large amount of research effort in the past few years.

In this context, a generative probabilistic model describes the joint probability of the observation and latent variables, $p(\boldsymbol{x}, \boldsymbol{z})$, and thus the generative network maps the latent variable $\mathbf{z} \sim p(\boldsymbol{z})$ to a sample in observation space

$\mathbf{x} \sim p(\boldsymbol{x})$ or by specifying the conditional probability $p(\boldsymbol{x}|\boldsymbol{z})$.

To overcome the difficulty of maximum likelihood training in a directed deep generative model, and the intractability of posterior inference, $p(\mathbf{z}|\mathbf{x})$, a standard approach is to follow the variational principle of [Jordan et al., 1999] and introduce an approximate variational posterior $q_\eta(\mathbf{z}|\mathbf{x})$, which leads to

$$\log p_\theta(\mathbf{x}) = \underbrace{\mathbb{E}_{q_\eta}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{\mathrm{KL}}[q_\eta(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})]}_{ELBO} + D_{\mathrm{KL}}[q_\eta(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}|\mathbf{x})] \quad (1.1)$$

In the above, the last term measures the Kullback-Leibler (KL) divergence between the true posterior and its approximatation. The non-negative property of the KL divergence implies that the summation of the first two terms forms a variational lower bound on the marginal log-likelihood that are known as *evidence lower bound (ELBO)*. Variational inference maximizes the ELBO instead of likelihood and once *stochastic backpropagation* and a parameterized inference network are deployed, the approach is referred to as *amortized variational inference* [Rezende et al., 2014]. A schematic of VAE is illustrated in Figure 1.1.

One popular and successful approach to scaling such probabilistic generative learning is the *variational auto-encoder* (VAE), which simultaneously trains the approximate variational inference network, also called the encoder, and the generative network, also called the decoder, leveraging both the stochastic backpropagation and reparameterization trick [Kingma and Welling, 2013]. According to this perspective, classical auto-encoders (AEs) can be viewed as a special case of variational auto-encoders when the probabilistic models are reduced to deterministic variants and both the encoder and decoder are trained in an end-to-end architecture. It has been shown in [Kingma and Welling, 2013] that different dimensions of the latent variable encode underlying explanatory factors of variation, therefore VAEs are considered powerful models for discovering and disentangling the natural and interpretable representations of observed inputs.

The gap between the true posterior and its approximation, however, hinders

the performance of VAE, which has recently motivated many researchers to enhance the expressiveness of approximate inference network to reduce this gap. One such development is to construct flexible parametrized probability density functions using normalizing flows, which is considered further in chapter 2.

Another recent approach to probabilistic latent variable modeling is to associate a discriminator network to a generative network then formulate the learning process as a two-player minimax game rather than the maximum likelihood optimization. In such a two-player game, each player tries to optimize its objective; the discriminator tries to distinguish the real data from the fake samples while the generator tries to trick the discriminator by producing more realistic samples, hence resulting in a minimax adversarial training algorithm known as *Generative Adversarial Networks* (GANs). In contrast to AR and VAE, which explicitly model the data distribution, GANs are more focused on the generated samples by implicitly learning the underlying distribution of the date, hence successfully trained GANs are able to achieve state-of-the-art results in generating new realistic-looking, high-resolution images. Nevertheless, training stability and sample generation diversity remain core challenges in GANs applications. The minimax game nature of GAN training makes the process unstable and prone to suffering from number of issues such as vanishing gradient, converging to poor local optima and mode collapse [Wiatrak and Albrecht, 2019]. Moreover, they do not naturally offer an inference network to recover the latent representation from the sample points, and lack a tractable and comprehensive metric to evaluate the diversity of generated samples [Theis et al., 2015]. Research on tackling these issues and improving GANs has surged in the past few years.

Another line of work that has received a large amount of interest recently is to directly estimate the distribution of the data by *normalizing flows*. The normalizing flow is a chain of smooth and invertible transformations (bijections) to construct a complex probability density by transforming a simple base density, such as a standard normal distribution, exploiting the change of variable formula. Given a random variable $\mathbf{z} \sim p(\mathbf{z})$ and an invertible and differentiable mapping $g : \mathbb{R}^n \to \mathbb{R}^n$, with inverse mapping $f = g^{-1}$, the probability density function

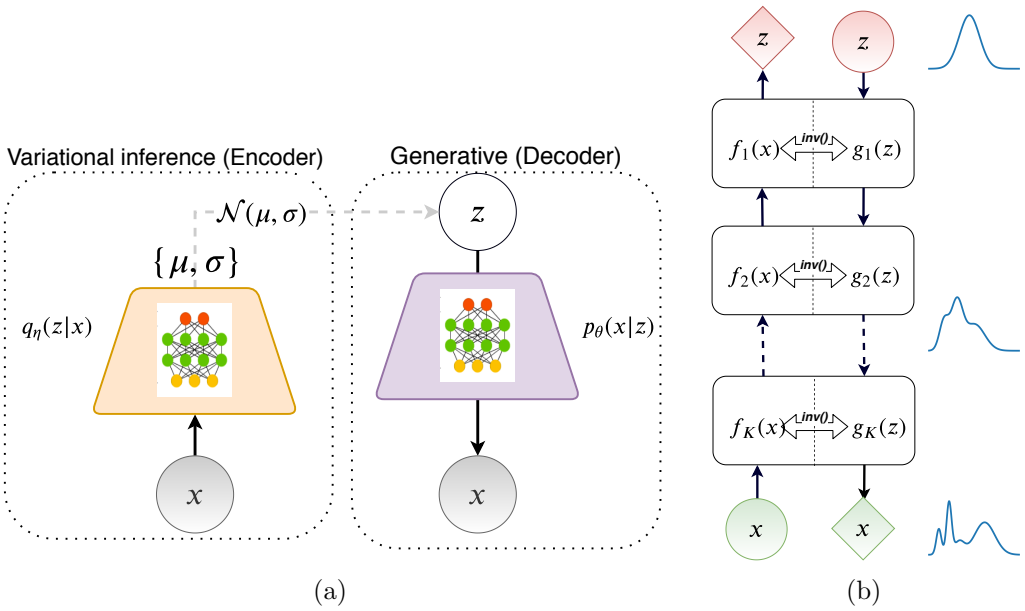


Figure 1.1: Schematic representation of (a) a vanilla Variational Auto-Encoder (VAE) model, and (b) a Normalizing Flow model.

of the transformed variable $\boldsymbol{x} = g(\mathbf{z})$ can be described by the *change of variable formula* as

$$\begin{aligned} p(\boldsymbol{x}) &= p(\mathbf{z}) \left|\det \boldsymbol{J}_g\right|^{-1} \\ &= p(f(\boldsymbol{x})) \left|\det \boldsymbol{J}_f\right| \end{aligned} \tag{1.2}$$

This formula provides a framework for probabilistic generative modeling. Training such models necessitates evaluation of Jacobian determinant in formula 1.2, which is a non-trivial computation and merits special consideration. By carefully designing the transformations, one can obtain exact and scalable maximum likelihood training and exact inference. Recently, many researchers have focused on designing highly flexible and expressive flows with scalable Jacobian determinant evaluation, which is also the main focus of chapter 2.

Besides generative density estimation, flow based models have been adopted in variational inference and representation learning for hybrid modeling and reinforcement learning [Papamakarios et al., 2019]. Moreover, it has been shown that the autoregressive models, if invertible transformations are employed,

correspond to a form of normalizing flows [Papamakarios et al., 2017]. A schematic of a normalizing flow model for density estimation is illustrated in Figure 1.1.

1.2 Overview and Contributions

The main contributions of this thesis and abstracts of the main chapters are summarized as follows. First, chapters 2 and 3 focus on designing *deep generative models* based on the recent advances in this domain.

Chapter 2 is dedicated to deep invertible transformations, called *normalizing flows*, that have recently been extensively adopted in the design of deep generative models. Normalizing flows construct a complex probability density by transforming a simple base density, such as a standard normal distribution, via a chain of smooth, invertible mappings (bijections). Flow-based generative networks can be used to construct high quality generative probabilistic models, but training and sample generation require repeated evaluation of Jacobian determinants and function inverses. To make such computations feasible, current approaches employ highly constrained architectures that produce diagonal, triangular, or low rank Jacobian matrices. As an alternative, we investigate a set of novel normalizing flows based on *circular* and *symmetric* convolutions. We show that these transforms admit efficient Jacobian determinant computation and inverse mapping (deconvolution) in $\mathcal{O}(N \log N)$ time. Based on these invertible convolution filters, we have proposed an alternative nonlinear convolution layer, the *nonlinear data-adaptive convolution* transformation, where expressiveness is increased by allowing a layers kernel to adapt to the layers input. Additionally, element-wise multiplication, widely used in normalizing flow architectures, can be combined with these transforms to increase modeling flexibility. We further propose an analytic approach to designing nonlinear elementwise bijectors that induce special properties in the intermediate layers, by implicitly introducing specific regularizers in the loss. We show that these transforms allow more effective normalizing flow models to be developed in many applications especially for images. This work has been published in

[Karami et al., 2019a].

The focus of chapter 3 shifts to deep generative muli-view models. This chapter begins by reviewing probabilistic PCA, probabilistic CCA and a general probabilistic multi multi-view model. We then propose a deep probabilistic multi-view model that is composed of a linear multi-view layer, based on probabilistic canonical correlation analysis (CCA) description in the latent space, together with deep generative networks as observation models. The network is designed to decompose the variations of all views into a *shared latent representation* and a set of *view-specific components* where the shared latent representation is intended to describe the common underlying sources of variation among the views. The proposed method is then extended to a deep generative probabilistic multil-view model with an arbitrary number of views. An efficient variational inference procedure is developed that approximates the posterior distributions of the latent probabilistic multi-view layer while taking into account the solution of probabilistic CCA. The proposed model is particularly suitable for subspace clustering whereby the shared representation can be utilized to extract the underlying cluster memberships. Empirical studies confirm that the proposed deep generative multi-view model can successfully extend deep variational inference to multi-view learning while it efficiently integrates the relationship between multiple views to alleviate the difficulty of learning. The ideas of this work are presented in [Karami and Schuurmans, 2020b].

In chapter 4, we consider maximum likelihood estimation of classical linear dynamical systems with generalized-linear observation models. Linear dynamical systems (LDS) provide a fundamental model for estimation and forecasting in discrete-time multi-variate time series. In an LDS, each observation is associated with a latent state; these unobserved states evolve as a Gauss-Markov process where each state is a linear function of the previous state plus noise. Such a model of a partially observed dynamical system has been widely adopted, particularly due to its efficiency for prediction of future observations using Kalman filtering. Maximum likelihood is typically considered to be hard in this setting since latent states and transition parameters must be inferred jointly.

Given that expectation-maximization (EM), a classical iterative approach to likelihood maximization, does not scale and is prone to local minima, moment-matching approaches from the subspace identification literature have become standard. These methods provide closed form solutions, often involving a singular value decomposition of a matrix constructed from the empirical moments of observations. Recent evidence, however, suggests that these moment-matching approaches may suffer from weak statistical efficiency, performing particularly poorly with small sample sizes. In chapter 4, we instead reconsider likelihood maximization and develop an optimization based strategy for recovering the latent states and transition parameters under exponential family observation noise. Key to the proposed approach is a two-view reformulation of maximum likelihood estimation for linear dynamical systems which allows us to approximate the estimation task as a form of matrix factorization, and apply recent global optimization techniques for such models. To extend these optimization algorithms to this setting, we provide a novel proximal mapping update for the two-view approach that significantly simplifies the algorithm. We show that the proposed estimation strategy outperforms widely-used identification algorithms such as subspace identification methods, both in terms of accuracy and runtime, while scaling better with increasing sample size and dimensions. The contributions of this chapter was published in [Karami et al., 2017].

Chapter 2

Invertible Convolutional Flow

2.1 Introduction

Flow-based generative networks have shown tremendous promise for modeling complex observations in high dimensional datasets. In flow-based models, a complex probability density is constructed by transforming a simple base density, such as a standard normal distribution, via a chain of smooth, invertible mappings (bijections), to yield a *normalizing flow* [Rezende and Mohamed, 2015]. Such models are employed in various contexts, including approximating a complex posterior distribution in variational inference [Rezende and Mohamed, 2015], or for density estimation with generative models [Dinh et al., 2016].

Using a complex transformation (bijective function) to define a normalized density requires the computation of a Jacobian determinant, which is generally impractical for arbitrary neural network transformations. To overcome this difficulty and enable fast computation, previous work has carefully designed architectures that produce simple Jacobian forms. For example, [Rezende and Mohamed, 2015, van den Berg et al., 2018] consider transformations with a Jacobian that corresponds to low rank perturbations of a diagonal matrix, enabling the use of Sylvester's determinant lemma. Other works, such as [Dinh et al., 2014, 2016, Kingma et al., Papamakarios et al., 2017], use a constrained transformation where the Jacobian has a triangular structure. The latter approach has proved particularly successful, since this constraint

is easy to enforce without major sacrifices in expressiveness or computational efficiency. More recently, Kingma and Dhariwal [2018] proposed the use of 1×1 convolutions for cross channel mixing in a multi-channel signal, achieving tractability via a block diagonal Jacobian. Nevertheless, these models have overlooked some opportunities for formulating tractable normalizing flows that can enhance expressiveness and better capture the structure of natural data, such as images and audio. Also, a new line of work based on ordinary differential equations has emerged recently that offers promising continuous dynamics based flows [Grathwohl et al., 2019].

In this work, we propose an alternative nonlinear convolution layer, the *nonlinear adaptive convolution filter*, where expressiveness is increased by allowing a layer's kernel to adapt to the layer's input. The idea is to partition the input of a layer $\boldsymbol{x}$ into $\{\boldsymbol{x}_1,\ \boldsymbol{x}_2\}$, where the convolution updates $\boldsymbol{x}_2$ as $\boldsymbol{w}(\boldsymbol{x}_1) * \boldsymbol{x}_2$, while the kernel $\boldsymbol{w}(\boldsymbol{x}_1)$ is a function of $\boldsymbol{x}_1$ that can be expressed by a deep neural network. We present invertible convolution operators whose Jacobian can be computed efficiently, making this approach practical for normalizing flow. Unlike the causal convolution employed in [Oord et al., 2016c] to generate audio waveforms, or in [Zheng et al., 2017] to approximate the posterior in a variational autoencoder, the proposed transformations are not constrained to depend only on the preceding input variables and also offer efficient inverse mapping, also known as deconvolution, analytically. It is worth noting that circular convolution has been recently adopted in [Karami et al., 2018] as a normalizing flow for density estimation and in [Hoogeboom et al., 2019] to design invertible periodic convolution for (almost) periodic data. Furthermore, we propose an analytic approach to add invertible pointwise nonlinearity in the flow that implicitly induces specific regularizers on the intermediate layers.

2.2 Background

Given a random variable $\mathbf{z} \sim p(\mathbf{z})$ and an invertible and differentiable mapping $g : \mathbb{R}^n \to \mathbb{R}^n$, with inverse mapping $f = g^{-1}$, the probability density function

of the transformed variable $\boldsymbol{x} = g(\mathbf{z})$ can be recovered by the *change of variable rule* as $p(\boldsymbol{x}) = p(\mathbf{z}) \left|\det \boldsymbol{J}_g\right|^{-1} = p(f(\boldsymbol{x})) \left|\det \boldsymbol{J}_f\right|$. Here $\boldsymbol{J}_g = \frac{\partial g}{\partial \mathbf{z}^\top}$ and $\boldsymbol{J}_f = \frac{\partial f}{\partial \boldsymbol{x}^\top}$ are the Jacobian matrices of functions g and f, respectively. One can use these to build a complex mapping g by composing a chain of simple bijective maps, $g = g^{(1)} \circ g^{(2)} \circ ... \circ g^{(K)}$, that preserve invertibility, with the inverse mapping being $f = f^{(K)} \circ f^{(K-1)} \circ ... \circ f^{(1)}$. By applying the chain rule to the Jacobian of the composition, and using the fact that $\det \boldsymbol{A}\boldsymbol{B} = \det \boldsymbol{A} \det \boldsymbol{B}$, the log-likelihood equality (LLE) can be written as

$$\log p(\boldsymbol{x}) = \log p(\mathbf{z}) + \sum_{k=1}^{K} \log \left|\det \boldsymbol{J}_{f_k}\right| . \tag{2.1}$$

Evaluating the Jacobian determinant is the main computational bottleneck in (2.1) since, in general, its scaling is cubic in the size of input. It is therefore natural to seek structured transformations that mitigate this cost while retaining useful modeling flexibility.

Notation definition: Throughout the chapter, invertible flows are denoted by f, while $f(\boldsymbol{x})$ is used for unconditional flows, and conditional (data-parameterized) flows are identified by $f(\boldsymbol{x}_2; \boldsymbol{x}_1)$ or $f(\boldsymbol{x}_2; \theta(\boldsymbol{x}_1))$ where the flow warps $\boldsymbol{x}_2$ conditioned on $\boldsymbol{x}_1$. Subscripts are intended to specify the type of flow or its parameters while superscripts enumerate the order of flows in the chain. For example, f_* denotes the convolutional flow in general and σ_α is used to specify the pointwise nonlinear bijectors with its inverse being ϕ_α. Also, in general, $\boldsymbol{y}$ and $\boldsymbol{x}$ indicate the output and input of a flow, respectively and when referring to the k^{th} flow in the chain, we use $\boldsymbol{y}^{(k)}$ and $\boldsymbol{x}^{(k)}$ where $\boldsymbol{x}^{(k)} = \boldsymbol{y}^{(k-1)}$. Moreover, *circular convolution* and *symmetric convolution* are denoted by $\circledast$ and $*_s$, respectively, while $*$ denotes an invertible convolution in general, and $\boldsymbol{x}_\mathcal{F}$, $\boldsymbol{x}_\mathcal{C}$ and $\boldsymbol{x}_\mathcal{T}$ denote *DFT*, *DCT* and *trigonometric transform* of sample $\boldsymbol{x}$, respectively.

2.2.1 Toeplitz structure and Circular Convolution

Although available methods have typically considered bijections whose Jacobians have block-diagonal or triangular forms, these are not the only useful possibilities. In fact, various other transformations exist whose Jacobian has sufficient structure to allow computationally efficient determinant calculation. One such structure is the *Toeplitz* property, where all the elements along each diagonal of a square matrix are identical (Figure 2.1(a)). The calculation of the determinant can then be simplified significantly. Let $\boldsymbol{J}_T$ be a Toeplitz matrix of size $N \times N$; its determinant can be evaluated in $\mathcal{O}(N^2)$ time in general [Monahan, 2011]. More specifically, if $\boldsymbol{J}_T$ has a limited bandwidth size of $K = r + s$, as depicted in Figure 2.1(a), then the determinant computation can be reduced to $\mathcal{O}(K^2 \log N + K^3)$ time [Cinkir, 2011]. Moreover, Toeplitz matrices can be inverted efficiently [Martinsson et al., 2005]. The fact that the discrete convolution can be expressed as a product of a Toeplitz matrix and the input [Gray et al., 2006] highlights that the Toeplitz property is of particular interest in *convolutional neural networks (CNNs)*.

In this chapter, we consider a particular transformation whose Jacobian is a *circulant matrix*, a special form of Toeplitz structure where the rows (columns) are cyclic permutations of the first row (column), *i.e.* $J_{l,m} = J_{1,(l-m) \bmod N}$. See Figure 2.1(b) for an illustration. This structure allows certain computationally expensive algebraic operations, such as determinant calculation, inversion and eigenvalue decomposition, to be performed efficiently in $\mathcal{O}(N \log N)$ time by exploiting the fact that a square circulant matrix can be diagonalized by a discrete Fourier transform (DFT) [Gray et al., 2006]. Define the circular convolution as $\boldsymbol{y} := \boldsymbol{w} \circledast \boldsymbol{x}$ where $\boldsymbol{y}(i) := \sum_{n=0}^{N-1} \boldsymbol{x}(n)\boldsymbol{w}(i-n) \bmod N$, which is equivalent to the linear convolution of two sequences when one is padded cyclically, also known as periodic padding, as illustrated in Figure 2.2(a). The key property we exploit in developing an efficient normalizing layer is that the Jacobian of this convolution forms a circulant matrix, hence its determinant and inverse mapping (deconvolution) can be computed efficiently. Some useful properties of this operation are needed:

$$\boldsymbol{J}_T = \begin{bmatrix} w_0 & w_{-1} & \dots & w_{-s} & & \mathbf{0} \\ w_1 & w_0 & & & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ w_r & \ddots & \ddots & \ddots & \ddots & w_{-s} \\ & \ddots & \ddots & \ddots & w_0 & w_{-1} \\ \mathbf{0} & & w_r & \dots & w_1 & w_0 \end{bmatrix}$$

(a)

$$\boldsymbol{J}_C = \begin{bmatrix} w_0 & w_{N-1} & \dots & w_2 & w_1 \\ w_1 & w_0 & \ddots & \ddots & w_2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ w_{N-2} & \ddots & \ddots & w_0 & w_{N-1} \\ w_{N-1} & w_{N-2} & \dots & w_1 & w_0 \end{bmatrix}$$

(b)

Figure 2.1: (a) $\boldsymbol{J}_T$ is a Toeplitz matrix with limited bandwidth size of $K = r + s$, (b) $\boldsymbol{J}_C$ is the Jacobian of circular convolution that is a circulant matrix.

Proposition 2.1 *Let $\boldsymbol{y} := \boldsymbol{w} \circledast \boldsymbol{x}$ be a circular convolution on the input vector $\boldsymbol{x}$ with its DFT transform $\boldsymbol{x}_\mathcal{F} := \mathcal{F}_{DFT}\{\boldsymbol{x}\}$ whereas $\boldsymbol{y}_\mathcal{F} := \mathcal{F}_{DFT}\{\boldsymbol{y}\}$ and $\boldsymbol{w}_\mathcal{F} := \mathcal{F}_{DFT}\{\boldsymbol{w}\}$ are the DFT transform of $\boldsymbol{y}$ and $\boldsymbol{w}$, respectively. Then:*

a) *The circular convolution operation can be expressed as a vector-matrix multiplication $\boldsymbol{y} = \boldsymbol{C}_w \boldsymbol{x}$ where $\boldsymbol{C}_w$ is a circulant square matrix having the convolution kernel $\boldsymbol{w}$ as its first row.*

b) *The Jacobian of the mapping is $\boldsymbol{J}_y = \boldsymbol{C}_w$.*

c) *The matrix $\boldsymbol{C}_w$ can be diagonalized using DFT basis with its eigenvalues being equal to the DFT of $\boldsymbol{w}$, hence $\log|\det \boldsymbol{J}_y| = \sum_{n=0}^{N-1} \log|\boldsymbol{w}_\mathcal{F}(n)|$.*

d) *The circular convolution can be expressed by element-wise multiplication in the frequency domain, $\boldsymbol{y}_\mathcal{F}(k) = \boldsymbol{w}_\mathcal{F}(k)\ \boldsymbol{x}_\mathcal{F}(k)$, a.k.a. the circular convolution-multiplication property.*

e) *If $\boldsymbol{w}_\mathcal{F}(n) \neq 0\ \forall n$, this linear operation is invertible with inverse $\boldsymbol{x}_\mathcal{F}(n) = \boldsymbol{w}_\mathcal{F}^{-1}(n)\ \boldsymbol{y}_\mathcal{F}(n)$. Moreover, its inverse mapping (deconvolution) is also a circular convolution operation with kernel $\boldsymbol{w}^{inv} := \mathcal{F}_N^{-1}\{\boldsymbol{w}_\mathcal{F}^{-1}\}$. On the other hand, the log determinant Jacobian also acts as a log-barrier in the objective function that in turn prevents the $\boldsymbol{w}_\mathcal{F}(n)$ from becoming zero hence enforces the invertibility of the convolution filter.*

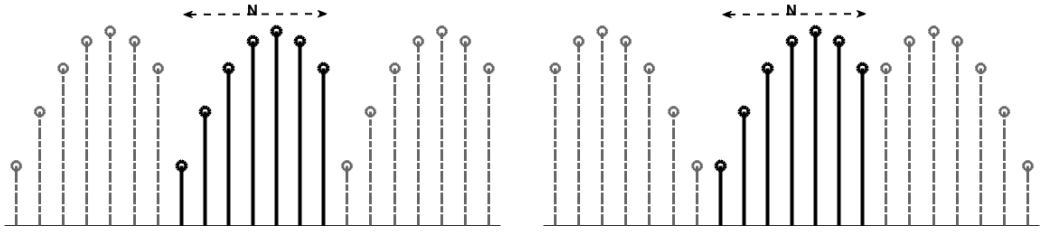


Figure 2.2: (a) Cyclic (periodic) extension and (b) even-symmetric extension of the base sequence, where the base sequence specified by dark solid lines.

f) *The circular convolution, its inverse, and Jacobian determinant can all be efficiently computed in $\mathcal{O}(N \log N)$ time in the frequency domain, exploiting Fast Fourier Transform (FFT) algorithms.*

These properties are based on the results in [Gray et al., 2006, Gonzalez and Woods, 1992].

2.2.2 Symmetric convolution

Circular convolution is not a unique operation with such properties, *symmetric convolution* is another form of structured filtering operation that can be adopted to achieve interesting desirable properties. A family of symmetric extension (padding) patterns and their corresponding discrete trigonometric transforms (DTT) are outlined in [Martucci, 1994], based on which alternative symmetric convolution filters can be defined that satisfy the convolution-multiplication property. Among this family, we choose an even-symmetric extension that can be readily interpreted. Define an even-symmetric extension of a base sequence of length N around $N - 1/2$ as

$$\hat{\boldsymbol{x}}(n) = \varepsilon\{\boldsymbol{x}(n)\} := \begin{cases} \boldsymbol{x}(n) & n = 0, 1, ..., N-1 \\ \boldsymbol{x}(-n-1) & n = -N, ..., -1 \end{cases}. \tag{2.2}$$

This even-symmetric extension is illustrated in Figure 2.2(b). The *symmetric convolution* of two sequences, denoted by $*_s$, can then be defined by the

$$\boldsymbol{J}_S = \begin{bmatrix} w_0 & w_0 & \dots & w_{N-3} & w_{N-2} \\ w_1 & w_0 & \ddots & w_{N-4} & w_{N-3} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ w_{N-2} & w_{N-3} & \ddots & w_0 & w_0 \\ w_{N-1} & w_{N-2} & \dots & w_1 & w_0 \end{bmatrix} + \begin{bmatrix} w_1 & w_2 & \dots & w_{N-1} & w_{N-1} \\ w_2 & w_3 & \iddots & w_{N-1} & w_{N-2} \\ \vdots & \iddots & \iddots & \iddots & \vdots \\ w_{N-1} & w_{N-1} & \iddots & w_2 & w_1 \\ w_{N-1} & w_{N-2} & \dots & w_1 & w_0 \end{bmatrix}$$

(a)

Figure 2.3: $\boldsymbol{J}_S$ is the Jacobian of symmetric convolution that can be expressed as summation of a Toeplitz matrix and an upside-down Toeplitz matrix (also called a Hankel matrix where its skew-diagonal elements are identical).

circular convolution of their corresponding even-symmetric extensions, as $\boldsymbol{y} = \boldsymbol{w} \ *_s \ \boldsymbol{x} := \mathcal{R}\{\hat{\boldsymbol{x}} \circledast \hat{\boldsymbol{w}}\}$, where $\mathcal{R}\{.\}$ is a rectangular window operation that retains the base sequence of interest in an extended sequence; that is, it inverts the symmetric extension operation (2.2). Now, since the sequences are extended by an even-symmetric pattern, the cosine functions provide the appropriate basis for the Fourier transform, giving rise to the discrete cosine transform of type two (DCT-II):

$$\boldsymbol{x}_{\mathcal{C}}(k) = \mathcal{F}_{dct}\{\boldsymbol{x}\}_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} \frac{\sqrt{2}}{\sqrt{1_{n=0}+1}} \boldsymbol{x}(n) \cos\left(\frac{\pi k}{N}(n+\tfrac{1}{2})\right). \tag{2.3}$$

The convolution-multiplication property holds for this convolution, which implies that the symmetric convolution of two sequences in the spatial domain can be expressed as a pointwise multiplication in the transform domain, after a forward DCT of its operands, *i.e.* $\boldsymbol{y}_{\mathcal{C}} = \boldsymbol{w}_{\mathcal{C}} \odot \boldsymbol{x}_{\mathcal{C}}$. This property also offers and alternative definition for the symmetric convolution: the inverse DCT of pointwise multiplication of the forward DCT of its operands [Martucci, 1994].

One can also show that the symmetric convolution provides a structured Jacobian that can be specified by Toeplitz matrices; see Figure 2.3(a) for an illustration. Analogous to the results presented in Proposition 2.1 for circular convolution, the symmetric convolution-multiplication property implies that the Jacobian of the symmetric convolution can be diagonalized by a DCT basis, with eigenvalues being the DCT of the convolution kernel. Similarly, the inverse

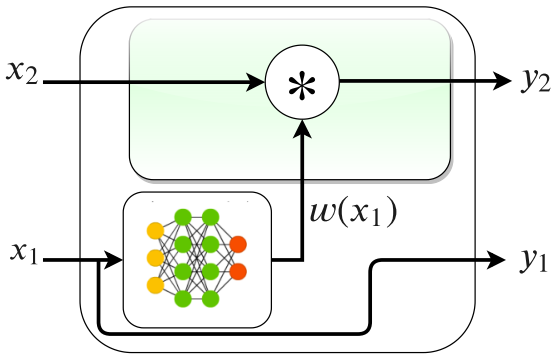


Figure 2.4: The diagram of the data-adaptive convolution sub-flow.

filter (*deconvolution*) can be obtained by inverting the kernel coefficients in the transform domain, i.e. $\boldsymbol{w}^{inv} := \mathcal{F}_{dct}^{-1}\{1./\boldsymbol{w}_{\mathcal{C}}\}$, where, again, the *invertibility* of the convolution is guaranteed by the fact that it log determinant Jacobian in the objective function keeps the elements of $\boldsymbol{w}_{\mathcal{C}}$ away from zero (as a log-barrier). On the other hand, since the DCT can be defined in terms of a DFT of the symmetric extension of the original sequences, the symmetric convolution, its inverse, and Jacobian determinant can exploit available fast Fourier algorithms with $\mathcal{O}(N \log N)$ complexity.[1]

2.3 Convolutional normalizing flow

2.3.1 Data adaptive convolution layer

The special convolutional forms introduced above appear to be particularly well suited to capturing structure in images and audio signals, therefore we seek to design more expressive normalizing flows using the convolution bijections as building blocks. To increase flexibility, we propose a *data-adaptive convolution* filter with a filter kernel that is a function of the input of the layer.

Inspired by the idea of the coupling layer in [Dinh et al., 2016], a modular bijection can be formed by splitting the input $\boldsymbol{x} \in \mathbb{R}^d$ into two disjoint parts $\{\boldsymbol{x}_1 \in \mathbb{R}^{d_1}, \boldsymbol{x}_2 \in \mathbb{R}^{d_2} : \ d_1 + d_2 = d\}$, referred to as the *base input* and *update input*, respectively, and only updating $\boldsymbol{x}_2$ by an invertible convolution operation

[1] All bijective convolutions in experiments were performed in transform domain using a fast Fourier transform algorithm.

with a data-parameterized kernel that depends on $\boldsymbol{x}_1$. The data-adaptive convolution sub-flow can then be expressed as

$$f_*(\boldsymbol{x}_2; \boldsymbol{x}_1) = \boldsymbol{w}(\boldsymbol{x}_1) * \boldsymbol{x}_2. \tag{2.4}$$

In the above transformation $*$ is an invertible convolution operation and can be one of the invertible convolutions introduced in last section. Here, the kernel $\boldsymbol{w}(\boldsymbol{x}_1)$ can be any nonlinear function, which leads to a *nonlinear adaptive convolution* filtering scheme. Its diagram is depicted in Figure 2.4.

2.3.2 Pointwise nonlinear bijections

Adding pointwise nonlinear bijections in the chain of normalizing flows can further enhance expressiveness. More specifically, focusing on the Jacobian determinant introduced by the nonlinearities in log-likelihood equation (2.1), one can observe that these terms can be interpreted as regularizers on the latent representation. In other words, specific structures on intermediate activations can be encouraged by designing customized pointwise nonlinear gates; these structures encode various prior knowledge into the design of the model. Let $\sigma^{(k)}$ denote the k^{th} bijection in the chain of normalizing flows that is assumed to be a pointwise nonlinear operation, *i.e.* $\boldsymbol{y}_i^{(k)} = \sigma^{(k)}(\boldsymbol{x}_i^{(k)})$. Dropping the indices, this mapping can be simply written as $y = \sigma(x)$ with inverse $x = \phi(y) = \sigma^{-1}(y)$. Since the nonlinearity operates elementwise, its Jacobian is diagonal, hence the log determinant reduces to $\log|\det \boldsymbol{J}_y| = \sum_{i=1}^{d} \log\left|\frac{\partial\sigma(x_i)}{\partial x_i}\right|$. Then, an analytic approach to designing nonlinear invertible gates are derived in the following.

Proposition 2.2 *Assume we want to induce a specific structure, formulated by a regularizer $\gamma(y)$, on the intermediate activation $y := \boldsymbol{y}_i^{(k)}$. Then the elementwise bijection can be defined as the solution to the differential equation:*

$$\left|\frac{\partial\sigma^{-1}}{\partial y}\right| = \left|\frac{\partial\phi}{\partial y}\right| = e^{\gamma(y)}. \tag{2.5}$$

In other words, the contribution to the $-\log|\det \boldsymbol{J}_\sigma|$ term in the negative

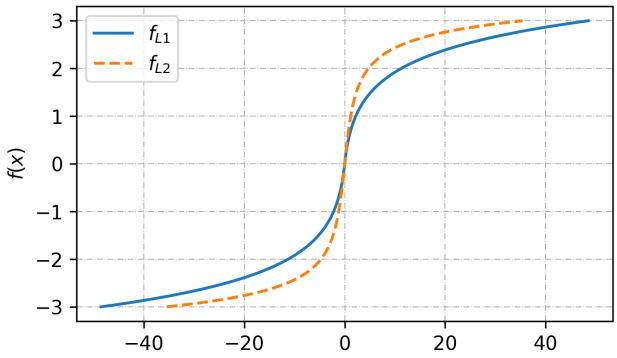


Figure 2.5: Nonlinear gates corresponding to $l1$ and $l2$ regularizers.

log-likelihood from this unit will then reduces to $\log|\frac{\partial\phi}{\partial y}| = \gamma(y)$.

Solving the above equation and deriving the nonlinear bijection for two well established $l1$ and $l2$ regularizers leads to the following.

- $l1$ regularization: $\gamma(y) = \alpha|y|$ which corresponds to Laplace distribution assumption on y:

$$\phi_\alpha(y) = \tfrac{\text{sign}(y)}{\alpha}(e^{\alpha|y|} - 1), \quad \sigma_\alpha(x) = \tfrac{\text{sign}(x)}{\alpha}\ln(\alpha|x| + 1). \tag{2.6}$$

  Due to its symmetric logarithmic shape, we call the forward function $\sigma_\alpha(x)$ an *S-Log* gate parameterized by positive-valued α.

- $l2$ regularization: $\gamma(y) = \alpha y^2$ which corresponds to Gaussian distribution assumption on y:

$$\phi_\alpha(y) = \sqrt{\tfrac{\pi}{4\alpha}}\text{erfi}(\sqrt{\alpha}y), \quad \sigma_\alpha(x) = \tfrac{1}{\sqrt{\alpha}}\text{erfi}^{-1}(\sqrt{\tfrac{4\alpha}{\pi}}x).$$

The proposed nonlinear gates, plotted in Figure 2.5, are not only differentiable by construction but also have unbounded domain and range, making them suitable choices for designing normalizing flows in many settings such as density estimation. Due to its simple analytical form and closed form inversion, the *S-Log* gate, (2.6), is adopted as nonlinear bijection in our model architecture. For multichannel inputs, we assume that the gates share the same parameter α over all spatial locations of a channel (feature map).

2.3.3 Combined convolution multiplication layer

The convolution operation spatially slides a filter and applies the same weighted summation at every location of its input, resulting in location invariant filtering. To achieve a more flexible and richer filtering scheme, we can combine an element-wise multiplication, indicated by $f_{\odot}$, and invertible convolution, indicated by f_*, so that the filtering scheme varies over space and frequency. The product of a diagonal matrix with a circulant matrix was also proposed in [Cheng et al., 2015] as a structured approximation for dense (fully connected) linear layers, while [Moczulski et al., 2015] showed that any $N \times N$ linear operator can be approximated to arbitrary precision by composing order N of such products.

Overall, the aforementioned components can be deployed to compose a *combined convolutional flow* as

$$\begin{aligned} f_{\boldsymbol{w},\boldsymbol{s}}(\boldsymbol{x}_2;\boldsymbol{x}_1) &= (\sigma_{\alpha'} \circ f_{\odot} \circ \sigma_{\alpha} \circ f_*)(\boldsymbol{x}_2;\boldsymbol{x}_1) \\ &= \sigma_{\alpha'}\big(\ \boldsymbol{s}(\boldsymbol{x}_1) \odot \sigma_{\alpha}(\boldsymbol{w}(\boldsymbol{x}_1) * \boldsymbol{x}_2)\ \big) \end{aligned} \tag{2.7}$$

We found that a more expressive network can be achieved by stacking M iterates of the combined convolutional flows and an additive coupling transform in each step of the network. Therefore, the *convolutional coupling flow* (*CONF*) can be written as

$$\begin{cases} \boldsymbol{y}_1 = \boldsymbol{x}_1 \\ \boldsymbol{y}_2 = (f^{(M)}_{\boldsymbol{w},\boldsymbol{s}} \circ ... \circ f^{(1)}_{\boldsymbol{w},\boldsymbol{s}})(\boldsymbol{x}_2;\boldsymbol{x}_1) + \boldsymbol{t}(\boldsymbol{x}_1). \end{cases} \tag{2.8}$$

The parameters of the flow $\{\boldsymbol{w}_1, \boldsymbol{s}_1, ..., \boldsymbol{w}_M, \boldsymbol{s}_M, \boldsymbol{b}\}$ can be any nonlinear function of the base input $\boldsymbol{x}_1$ and are not required to be invertible, hence they can be modeled by deep neural networks with an arbitrary number of hidden units, offering flexibility and rich representation capacity while preserving an efficient learning algorithm. These are also called *conditioning networks* in the context of normalizing flow. The model complexity can be significantly reduced

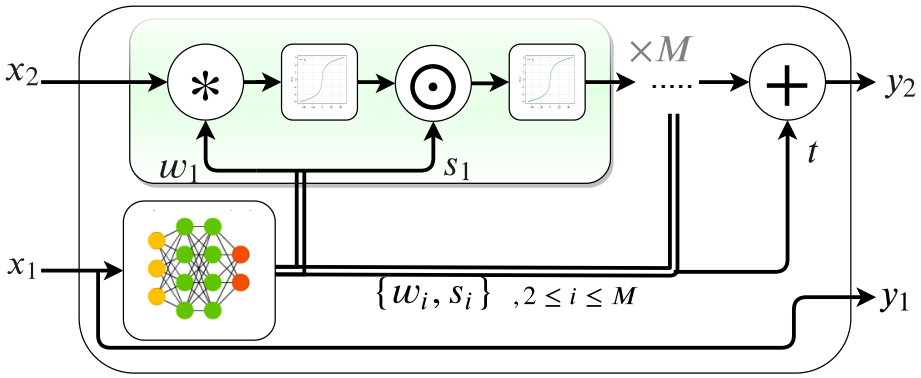


Figure 2.6: The diagram of one step of flow (CONF) that is composed of M combined convolutional flows defined in (2.7). In density estimation, the input to the conditioning neural network is the base input, $\boldsymbol{x}_1$, and the flow updates $\boldsymbol{x}_2$. In variational inference applications, the neural network is conditioned on the data points $\boldsymbol{x}$ while warping the latent random variable $\boldsymbol{z}$.

by using one conditioning neural network for all parameters of a coupling flow so that it shares all layers except the last one for generating the parameters of the flow. Consequently, we achieve a more expressive flow with the stack of bijectors in (2.8) without introducing too many extra NN layers in the model.

2.3.4 Jacobian determinant and inversion

The modular structure of coupling CONF modules (2.8) implies that its Jacobian determinant can be expressed in terms of its sub-flows. More precisely, its Jacobian is

$$\boldsymbol{J}_y = \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}^\top} = \begin{bmatrix} \boldsymbol{I}_{d_1} & \boldsymbol{0} \\ \frac{\partial \boldsymbol{y}_2}{\partial \boldsymbol{x}_1^\top} & \frac{\partial \boldsymbol{y}_2}{\partial \boldsymbol{x}_2^\top} \end{bmatrix}. \tag{2.9}$$

Noticeably, the Jacobian is a block triangular matrix, so its determinant can be readily computed as the product of determinant of the square diagonal blocks,

therefore

$$\begin{aligned}\log|\det \boldsymbol{J}_y| &= \sum_{i=1}^{M} \log\left|\det \boldsymbol{J}_{w,s}^{(i)}\right| \\ &= \sum_{i=1}^{M} \log\left|\det \boldsymbol{J}_{\phi_{\alpha'}}^{(i)}\right| + \log\left|\det \boldsymbol{J}_{\odot}^{(i)}\right| + \log\left|\det \boldsymbol{J}_{\phi_{\alpha}}^{(i)}\right| + \log\left|\det \boldsymbol{J}_{*}^{(i)}\right|\end{aligned} \tag{2.10}$$

where $\boldsymbol{J}_{w,s}^{(i)}$ denotes the Jacobian of $f_{w,s}^{(i)}$. According to the results presented for invertible convolutions in section 3.1, $\log\left|\det \boldsymbol{J}_{*}^{(i)}\right|$ can be computed efficiently in $\mathcal{O}(N \log N)$ times using the fast Fourier transform algorithm. Also, it is worth noting that this term plays the role of a *log-barrier* in the final loss function that prevents the eigenvalues of the Jacobian from falling to zero hence enforces the *invertibility* of the convolution filter. Then, the inverse model of (2.8) is[2]

$$\begin{cases} \boldsymbol{x}_1 = \boldsymbol{y}_1 \\ \boldsymbol{x}_2 = (g_{\boldsymbol{w},\boldsymbol{s}}^{(1)} \circ ... \circ g_{\boldsymbol{w},\boldsymbol{s}}^{(M)})(\boldsymbol{y}_2 - \boldsymbol{t}(\boldsymbol{x}_1); \boldsymbol{x}_1) \end{cases}$$
$$\text{where } g_{\boldsymbol{w},\boldsymbol{s}}(\boldsymbol{y}_2; \boldsymbol{x}_1) = \boldsymbol{w}^{inv} * \phi_\alpha(\boldsymbol{s}^{inv} \odot \phi_{\alpha'}(\boldsymbol{y}_2).)$$

Remark: Note that the guarantee holds for continuous time gradient descent. It is technically possible, though not observed in practice, that SGD could produce a non-invertible kernel. Additionally, the space of non-invertible kernels is measure zero in the space of kernels (its rare for an eigenvalue to be *exactly* zero), and so non-invertible kernels are unlikely to occur by chance.

Initialization of the parameters: Better data propagation is expected to be achieved for very deep normalizing flows if the combined flow (2.7) acts (approximately) as an identity mapping at initialization. Accordingly, the parameters of the nonlinear bijector pair, $\{\sigma_\alpha, \sigma_{\alpha'}\}$, are initialized sufficiently

[2]The inverse kernel $\boldsymbol{w}(\boldsymbol{y}_1)^{inv}$ can indeed be derived through the procedure explained in Proposition 2.1 for circular convolution or in a similar way for symmetric convolution.

close to zero so that they behave approximately as linear functions at the outset. Furthermore, the conditioning networks are initialized such that the scaling filters, $\boldsymbol{s}$, and the convolution kernels at the frequency domain, $\mathcal{F}\{\boldsymbol{w}\}$, are all initially identity filters.

Multi-dimensional extension: The multi-dimensional discrete Fourier transform can be expressed in separable forms, meaning that the operations can be performed by successively applying 1-dimensional transforms along each dimension [Gonzalez and Woods, 1992]. The separability property ensures the results mentioned so far can be extended to multi-dimensional settings. In this work, we are particularly interested in 2-D operations for image data. Based on the 2-D circular convolution definition, its equivalent block-circulant matrix form, and diagonalization method by 2-D DFT [Gonzalez and Woods, 1992, Ch. 5], the results of the circular convolution in Proposition 2.1 can be readily generalized to the 2-D case.[3] The same properties apply to the 2-D symmetric convolution, since the symmetric convolution-multiplication property can be generalized naturally to the 2-D setting [Foltz and Welsh, 1998].

2.4 Model architecture

A highly flexible and complex density approximation can be formed by composing a chain of the convolution coupling layers introduced in this work. As explained in Section 3.1, the determinant of the Jacobian and inverse of the composition can then be obtained readily. In addition to the invertible transformation introduced in this work, we use the following bijections in the final architecture of the normalizing flow.

Cross-channel mapping (mixing) For multi-channel setting, the invertible convolution operation is performed in a depthwise fashion *i.e.* each input channel is filtered by a separate convolution kernel. Then cross channel information

[3]Due to the separability property, the 2-D DFT of matrices of size $N_1 \times N_2$ can be computed in $\mathcal{O}(N_1 N_2(\log N_1 + \log N_2))$ time.

flow can be complemented by channel shuffling or using a 1×1 convolution. The latter offered significant improvement with small computational overhead in normalizing flows [Kingma and Dhariwal, 2018] hence, is applied after each convolutional coupling layer in our architecture. Also, for single channel inputs, assuming equal size splits $\{\boldsymbol{x}_1, \boldsymbol{x}_2\}$ (base input and update input), these can be treated as two separate channels of the input and the same technique can be applied to mix them after each coupling layer.

Multiscale architecture To achieve latent representations at multiple scales and obtain more fine-grained features, a subset of latent variables can be factored out at the intermediate layers. This technique is very useful for large image datasets and can significantly reduce the computational cost in very deep models [Dinh et al., 2016].

Normalization To improve the training in very deep normalizing flows, batch normalization was employed as a bijection after each coupling layer in [Dinh et al., 2016]. To overcome the adverse effect of small minibatch size in batch normalization, Kingma and Dhariwal [2018] proposed *actnorm*, as normalization, which applies an affine transformation and normalizes the activation per channel, similar to batch normalization but with larger mini-batch size, at initialization while the parameters of this bijection are freely updated during training with smaller minibatch size; this technique is called data dependent initialization [Salimans and Kingma, 2016]. Thus, in density estimation experiments, we employed the actnorm layers as bijections in the chain of normalizing flow and also in the deep conditioning neural networks.

2.5 Experiments

2.5.1 Density estimation

We first conduct experiments to evaluate the benefits of the proposed flow model (CONF). As observed in [Huang et al., 2018], expressiveness of the affine

Table 2.1: Average test negative log-likelihood (in nats) for tabular datasets and (in bits/dim) for MNIST and CIFAR using fully connected conditioning networks (lower is better). C-CONF and S-CONF stands for circular and symmetric convolutional coupling flow presented in (2.8), respectively. Error bars correspond to 2 standard deviations. The results of the benchmark methods are from [Grathwohl et al., 2019].

	POWER	GAS	BSDS300	MNIST	CIFAR10
MADE	3.08 ± .03	-3.56 ± .04	-148.85 ± .28	2.04 ± .01	5.67 ± .01
MAF	-0.24 ± .01	-10.08 ± .02	-155.69 ± .28	1.89 ± .01	4.31 ± .01
Real NVP	-0.17 ± .01	-8.33 ± .14	-153.28 ± 1.78	1.93 ± .01	4.53 ± .01
Glow	-0.17 ± .01	-8.15 ± .40	-155.07 ± .03	-	-
FFJORD	-0.46 ± .01	-8.59 ± .12	-157.40 ± .19	-	-
S-CONF	**-0.48** ± .01	**-10.98** ± .13	**-163.23** ± .13	**1.26** ± .01	**3.78** ± .03
C-CONF	-0.47 ± .01	-10.84 ± .06	**-163.23** ± .34	**1.25** ± .01	3.82 ± .00

Table 2.2: Results in bits per dimension for MNIST and CIFAR10 using CNN based conditioning networks. The results of the benchmark methods are from [Kingma and Dhariwal, 2018] and [Grathwohl et al., 2019]

	Real NVP	Glow	FFJORD	S-CONF
MNIST	1.06	1.05	0.99	1.00
CIFAR10	3.49	3.35	3.40	3.34

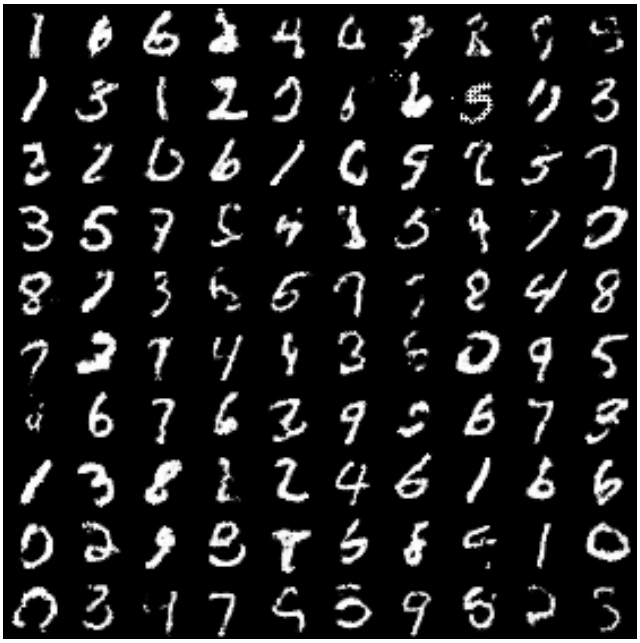
coupling flows and affine autoregressive flows stems from the complexity of the conditioning neural network that models flow parameters, and successive application of the flows. Therefore for fair comparison we follow [Papamakarios et al., 2017] and use a general-purpose neural network composed of fully connected layers in the design of conditioning networks. In this way we highlight the capacity of the flow itself, without relying on complex data dependent neural networks such as deep residual convolutional network used in [Dinh et al., 2016, Kingma and Dhariwal, 2018, Ho et al., 2019].

First we evaluate the proposed flow for density estimation on tabular datasets, considering two UCI datasets (POWR, GAS) and the natural image patches dataset (BSDS300) used in [Papamakarios et al., 2017]. Description of these datasets and the preprocessing procedure applied can be found therein. We also perform unconditional density estimation on two image datasets; MNIST, consisting of handwritten digits [Y. LeCun, 1998] and CIFAR-10, consisting of natural images [Krizhevsky, 2009]. In BSDS300, the value of

bottom-right pixel is replaced with the average of its immediate neighbors resulting in monochrome patches of size 8×8. For image data, the 2D invertible convolution is used as the flow. All datasets are dequantized by adding uniform distributed noise to each dimension, and then they are scaled to $[0, 1]$ values. Variational dequantization has been proposed as an alternative method offering better variational lower bound on the log-likelihood [Ho et al., 2019], but we here limit to uniform dequantization in order to focus merely on the evaluation of the proposed flows.

We compare the density estimation performance of CONF to the affine coupling flow models real-NVP [Dinh et al., 2016] and Glow [Kingma and Dhariwal, 2018], and the recent continuous-time invertible generative model FFJORD [Grathwohl et al., 2019]. These reversible models admit efficient sampling with a single pass of the generative model. We also compare the density estimation capacity of the proposed model against the autoregressive based methods, MADE [Germain et al., 2015], MAF [Papamakarios et al., 2017]. These family of autoregressive normalizing flows require $\mathcal{O}(D)$ evaluations of the generative function to sample from the model, making them prohibitively expensive for high dimensional applications. The results, summarized in Table 2.1, highlight that the circular convolution-based (FFT-based) CONF (C-CONF) and symmetric convolution-based (DCT-based) CONF (S-CONF) offer significant performance gains over the other models. The performance improvement of the symmetric convolution for CIFAR dataset over the circular convolution is expected and can be explained by the fact that circular padding causes discontinuity at the edges of its images while the symmetric extension is designed to prevent this effect. On the other hand, such behavior is not observed for MNIST as its images have black pixels all around their borders. Since S-CONF outperforms C-CONF in most of the experiments, we use it as the main convolutional flow in the next experiments, simply referring to it as CONF. The significant performance improvement of CONF on image datasets suggest that the feedforward conditioning NN were able to capture 2D local structures.

To make a fair comparison, we used a feedforward neural network archi-



(a)

(b)

Figure 2.7: Samples generated from the CONF model using general purpose fully connected NN as conditioning network that is trained on (a) the MNIST dataset and (b) the CIFAR-10 dataset.

tecture similar to the one used for MAF [Papamakarios et al., 2017] except that we simplified the architecture by using a single network for all parameters

Table 2.3: Average validation negative log-likelihood (in nats) of the ablations on GAS dataset at 5600 epochs.

	CONF	ablation: linear gates	ablation: no convolution
GAS	-10.89 ± .13	-10.12 ± .29	-10.74 ± .06

of a flow layer, while MAF used separate networks for the scaling and shift parameters. Each coupling flow is composed of a maximum of $M = 2$ iterates of the combined convolution flow. The parameters of the network and number of layers are selected to be comparable to those used in [Papamakarios et al., 2017]. Details of model architecture and experimental setup together with more empirical results are presented in appendix. Additionally, generated samples from the model are depicted in figure 2.7.

2.5.2 Ablations study

The coupling convolution flow (2.8) is composed of two new components compared to the affine coupling flow, 1) the pointwise nonlinear bijector and 2) the data-adaptive convolution. In this ablation study, we asses the contribution of each of these components on the overall performance of the CONF. The results in Table 2.3 highlights the effect of each ablation relative to CONF. These results show that the nonlinear bijector, S-Log, contributes more than the data-adaptive convolution in the performance improvement of CONF, in this case study. It is expected that the performance gain of the data-adaptive convolution is more significant in image datasets.

2.5.3 Density estimation using CNN based conditioning networks

We further assess the performance of CONF when the conditioning networks are based on convolutional neural networks, which are specifically designed for image data. A shallow convolutional NN, similar to the one used in GLOW, is employed to generate the parameters of the flow, except that we use one NN to generate all the parameters of a layer, reducing the number of model

(a)

(b)

Figure 2.8: Samples generated from the CONF model using CNN based conditioning NN that is trained on (a) the MNIST dataset and (b) the CIFAR-10 dataset.

parameters. The results of the experiments on MNIST and CIFAR10 data are presented in Table 2.2. The experimental setup and generated samples from the model can be found in Appendix A.1.1 and figure 2.8, respectively.

Table 2.4: Average test negative log-likelihood (in nats) and negative evidence lower bound (ELBO) on four benchmark datasets (lower is better). Reported error bars correspond to 2 standard deviations calculated over 3 trials. The combination of number of flow steps F and M of each model is reported in the format (F-M).

	MNIST		Omniglot		Caltech Silhouettes		Frey Faces	
	-ELBO	NLL	-ELBO	NLL	-ELBO	NLL	-ELBO	NLL
VAE	86.55 ± .06	82.14 ± .07	104.28 ± .39	97.25 ± .23	110.80 ± .46	99.62 ± .74	4.53 ± .02	4.40 ± .03
IAF	84.20 ± .17	80.79 ± .12	102.41 ± .04	96.08 ± .16	111.58 ± .38	99.92 ± .30	4.47 ± .05	4.38 ± .04
Planar	86.06 ± .31	81.91 ± .22	102.65 ± .42	96.04 ± .28	109.66 ± .42	98.53 ± .68	4.40 ± .06	4.31 ± .06
CONF(16-1)	83.89 ± .03	80.86 ± .05	98.35 ± .27	94.54 ± .12	108.64 ± 1.71	97.29 ± .91	4.43 ± .01	4.34 ± .02
O-SNF(4-8)	84.74	81.04 ± .15	101.41 ± .08	95.25 ± .09	109.37 ± .94	97.78 ± .47	4.50 ± .00	4.39 ± .01
CONF(4-8)	**83.22** ± .05	80.64 ± .06	**97.17** ± .08	94.19 ± .03	**104.09** ± 1.03	**94.56** ± .29	4.41 ± .01	4.31 ± .00
O-SNF(16-32)	83.32 ± .06	**80.22** ± .03	99.00 ± .29	93.82 ± .21	106.08 ± .39	94.61 ± .83	4.51 ± .04	4.39 ± .05
CONF(16-16)	-	-	**96.35** ± .05	**93.66**± .03	**101.10** ± .49	**92.37** ± .40	**4.39** ± .02	**4.29** ± .00

2.5.4 Variational inference

We also evaluate the proposed normalizing flow as a flexible inference network for a variational auto-encoder (VAE) [Rezende and Mohamed, 2015]. Here flows are only conditioned on encoded data points, produced by the encoder, and transform the posterior distribution of the latent variable without a coupling connection, resulting in $\boldsymbol{z}^{(t)} = (f^{(M)}_{\boldsymbol{w},\boldsymbol{s}} \circ ... \circ f^{(1)}_{\boldsymbol{w},\boldsymbol{s}})(\boldsymbol{z}^{(t-1)}; \boldsymbol{x}) + \boldsymbol{t}(\boldsymbol{x})$. We compare the performance of the trained VAE using this convolutional flow against other approaches, including a non flow-based VAE with factorized Gaussian distributions, and flow-based VAE using inverse autoregressive flow (IAF), planar flow [Rezende and Mohamed, 2015, Kingma et al.] and Sylvester normalizing flows (SNF) as the building blocks of the normalizing flows. We used the encoder/decoder architecture of [van den Berg et al., 2018] and the results of the available methods are adopted from that paper. The details of training procedure are summarized in Appendix A.1.2.

The results in Table 2.4 show that CONF outperforms Sylvester flow in most cases, and even smaller CONF models show similar or better capacity than larger SNF. This performance gains is achieved while we noted that the CONF and the SNF show almost similar speed in wall-clock time per training iteration. Also, we observe that CONF with $M = 1$ outperforms planar flow by a wide margin on all datasets, except for FreyFaces which is a challenging dataset and prone to overfitting for large SNF; here large CONF ($F = 16, M = 16$) perform

the best among all methods, so demonstrates less sensitivity to overfitting on the FreyFaces dataset.

Number of parameters: Let the stochastic latent variable be a D-dimensional vector $\boldsymbol{z} \in \mathbb{R}^D$ and the encoder's output be $\boldsymbol{e}(\boldsymbol{x}) \in \mathbb{R}^E$, then each step of CONF requires an additional $E \times (2MD + D) + 2M$ parameters to produce the flow parameters based on $\boldsymbol{e}(\boldsymbol{x})$, which is comparable to the number of parameters related to a step of planar flow if $M = 1$. This is, also, of the same order of the number of parameters of Sylvester flow with a bottleneck of size M, which is $E \times (2MD + 2M^2 + M)$.

Chapter 3

Deep Probabilistic Multi-view Learning

3.1 Introduction

When observations consist of multiple views or modalities of the same underlying source of variation, a learning algorithm should efficiently account for the complementary information to alleviate learning difficulty [Chaudhuri et al., 2009] and improve accuracy. A well-established method for two-view analysis is given by canonical correlation analysis (CCA) [Hotelling, 1992], a classical subspace learning technique that extracts the common information between two multivariate random variables by projecting them onto a subspace. CCA, as a standard model for unsupervised two-view learning, has been used in a broad range of tasks such as dimensionality reduction, visualization and time series analysis [Xia et al., 2014].

The goal of representation learning is to capture the essence of data and extract its natural features. Such features can be categories or cluster memberships. In multi-view data, the relationship between different views should be leveraged by the representation learning algorithms to enhance feature extraction. Learning representations in real-world applications, where the data is typically high-dimensional with complex structure, poses significant

challenges and necessitates flexible and expressive yet scalable models such as deep generative neural networks to be applied.

It has been shown in [Chaudhuri et al., 2009] that by projecting multi-view data onto low-dimensional subspaces using CCA, cluster memberships can be recovered under a weak separation condition thus resulting in easier clustering in the subspace. Nevertheless, CCA exhibits poor generalization when trained on small training sets, therefore [Klami and Kaski, 2007, Klami et al., 2013] adopt a Bayesian approach to solve a probabilistic interpretation of CCA. However, real applications involve nonlinear subspaces where more than two view are available. Recently, deep learning has received renewed interest as a standard approach for describing highly expressive modelsof increasingly complex datasets. In multi-view learning, several deep learning based approaches have been successfully extended [Ngiam et al., 2011, Andrew et al., 2013, Wang et al., 2015a]. A deep variational two-view autoencoder was proposed in [Tang et al., 2017, Wang et al., 2016] that in principle offers a generative two-view model with shared representation or shared plus view-specific factors. Despite the names of the methods, their link to CCA are only at a conceptual level without drawing theoretical connections between the proposed two-view models with the probabilistic CCA interpretation in [Bach and Jordan, 2005], while the black box variational inference was adopted.

Clustering Subspace clustering is a family of unsupervised learning methods that divide the high dimensional data points that are drawn from a multiple of low dimensional subspace into clusters of similar points. It has recently become a famous tool especially in the field of computer vision [Ji et al., 2017]. As stated in [Vidal, 2011] and [Ji et al., 2017], estimating the affinity matrix, that collects the similarity between all the pairs, from data points is the a key step in the subspace clustering, based on which the cluster memberships can be extracted.

Two main categories of subspace clustering methods for estimating the affinity matrix are [Ji et al., 2017]: 1) *factorization methods*: that estimate the affinities by factorizing the data into low-dimensional subspaces [Costeira

and Kanade, 1998, Ji et al., 2015, Mo and Draper, 2012, Vidal et al., 2008], and 2) *self-expressiveness based methods*: that represent a data points as a linear combination of other data points in the same subspace. Compared to factorization techniques, several variants of self-expressiveness based methods are tailored to be more robust to noise and outliers by choosing different forms of regularization objective functions terms [Elhamifar and Vidal, 2009, 2013, Feng et al., 2014, Li and Vidal, 2015, Liu et al., 2010, Wang et al., 2013, You et al., 2016]. Recently, a deep subspace clustering (DSC) method was proposed by applying the self expressive layer on a non-linear mapping of the data using a deep auto-encoder resulting in significant improvement in clustering performance for datasets lying in non-linear subspaces [Ji et al., 2017]. Abavisani and Patel [2018a] have recently extended the idea of DSC methods to multi-modal datasets by applying the self expressive layers that share a single common self-representation coefficients matrix for all modalities. However, the main drawback of such methods is that they rely on a self-representation coefficient matrix of size $N \times N$ where N is the number of data points, as the key component which makes these methods prohibitively expensive for large datasets. Hence, in this work we are interested in designing a deep representation learning method that performs clustering without relying on this expensive structure. In this respect, the proposed method in this work lies in the category of factorization based subspace clustering methods.

Main contributions: In this chapter, a modified formulation of probabilistic CCA is presented, then this linear probabilistic layer is extended to an interpretable deep generative multi-view network. The proposed model captures the variations of the views by a *shared latent representation*, describing the common underlying sources of variation, i.e. the essence of multi-view data, and a set of *view-specific latent factors*. Importantly, the model can be naturally generalized to an arbitrary number of views. We design the learning algorithm using a variational inference principle, which is known to be a powerful tool for scaling probabilistic models to complex problems and large datasets [Rezende et al., 2014]. The proposed variational inference is customized to incorporate the probabilistic CCA formulation, which also offers a flexible data fusion

method in the latent space that is appropriate for the general multi-modal setting. The proposed model is particularly suitable for subspace clustering whereby the shared representation can be utilized to extract the underlying cluster memberships. Empirical studies confirm that the proposed deep generative multi-view model can efficiently integrate the relationship between multiple views to alleviate learning difficulty in different downstream tasks, which is known to be the goal of multi-view learning approaches [Chaudhuri et al., 2009] [1].

Notation and Definitions Throughout this chapter, bold lowercase variables denote vectors (*e.g.* $\boldsymbol{x}$) or vector-valued random variables (*e.g.* $\mathbf{x}$), bold uppercase are used for matrices (*e.g.* $\boldsymbol{X}$) or matrix-valued random variables (*e.g.* $\mathbf{X}$) and unbold lowercase are scalars (*e.g.* x) or random variables (*e.g.* x). The transpose of a matrix is denoted as $\boldsymbol{A}^\top$ and $\boldsymbol{e}^{(i)} = [0, \ldots, 0, 1, 0, \ldots, 0]$ is the standard basis vector with a 1 at ith position. There are M views in total and subscripts are intended to identify the view-specific variable, (*e.g.* $\mathbf{x}_m, \boldsymbol{\Sigma}_{mm}$), which is different from an element of a vector that is specified by subscript (*e.g.* x_{mi}). The difference should be clear from context.

In this work the terms *view* and *modality* are used interchangeably to refer to the same concept. Likewise, in order to comply with many works in this field, both *multi-view* and *multi-modal* refer to the general case that all the views are available. The exception is in the experiments where it is specified clearly that only one of the views, the primary view, is available at the test time which will be referred as *"multi-view setting"* in contrast to *"multi-modal setting"* where all the views are available at the train and test time.

3.2 Probabilistic PCA

Principal component analysis (PCA) is a classical subspace learning method to learn a low dimensional representation with uncorrelated dimensions from

[1] Updated version of this work will be available on arXive [Karami and Schuurmans, 2020b]

the observations. Let $\mathbf{z} \in \mathbb{R}^d$ be a random vector of observation, PCA can also be defined as linear transformation of $\mathbf{z}$ into the subspace $\mathbb{R}^{d_0}$ as $\mathbf{r} = \boldsymbol{U}^\top \mathbf{z}$, where $\boldsymbol{U} \in \mathbb{R}^{d \times d_0}$ and $0 < d_0 \leq d$, such that all the dimensions of the projections have maximal variance. Let $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denote the mean and covariance matrices of $\mathbf{z}$, the solution of PCA is equal to $\boldsymbol{U} = \boldsymbol{\Lambda}_{d_0}^{-1/2} \boldsymbol{Q}_{d_0}$ where $\boldsymbol{\Lambda}_{d_0}$ is a diagonal matrix of the d_0 largest eigenvalues of $\boldsymbol{\Sigma}$ and $\boldsymbol{Q}_{d_0}$ is formed by their corresponding eigenvectors (d_0 principal eigenvectors). A probabilistic generative interpretation of PCA is presented in [Tipping and Bishop, 1999] that is extended in the following.

Theorem 3.1 *Assume a probabilistic model of the form*

$$\begin{aligned} \boldsymbol{\phi} &\sim \mathcal{N}(\boldsymbol{\mu}_0, \mathbf{I}_{d_0}), \quad 0 < d_0 \leq d \\ \mathbf{z}|\boldsymbol{\phi} &\sim \mathcal{N}(\boldsymbol{W}\boldsymbol{\phi} + \boldsymbol{\mu}_\epsilon, \sigma^2 \mathbf{I}_d), \quad \boldsymbol{W} \in \mathbb{R}^{d \times d_0}. \end{aligned} \tag{3.1}$$

The maximum likelihood solution of this model in then

$$\begin{aligned} \boldsymbol{W} &= \boldsymbol{U}_d (\boldsymbol{\Lambda}_{d_0} - \sigma^2 \mathbf{I})^{1/2} \boldsymbol{R}, \\ \sigma^2 &= \frac{1}{d - d_0} \sum_{i=d_0+1}^{d} \lambda_i, \quad \boldsymbol{\mu}_\epsilon = \boldsymbol{\mu} - \boldsymbol{W}\boldsymbol{\mu}_0 \end{aligned} \tag{3.2}$$

that are a function of principle eigenvectors and their eigenvalues, i.e. *they are related to solution of PCA problem and* $\boldsymbol{R}$ *is an arbitrary rotation matrix.*

In comparison to the results in [Tipping and Bishop, 1999], an extra degree of freedom $\boldsymbol{\mu}_0$, the mean of latent factor $\boldsymbol{\phi}$, is introduced in (3.1) . This parameter will be estimated when optimizing a lower bound of the maximum likelihood of a combined model in the section 3.3.2.

3.3 Probabilistic CCA

Canonical correlation analysis (CCA) [Hotelling, 1992] is a classical subspace learning method that extracts information from the cross-correlation between

two variables. Let $\mathbf{z}_1 \in \mathbb{R}^{d_1}$ and $\mathbf{z}_2 \in \mathbb{R}^{d_2}$ be a pair of random vectors corresponding to two different views. CCA linearly projects these onto the subspace $\mathbb{R}^{d_0}$ as $\mathbf{r}_1 = \boldsymbol{U}_1^\top \mathbf{z}_1$ and $\mathbf{r}_2 = \boldsymbol{U}_2^\top \mathbf{z}_2$, where $\boldsymbol{U}_1 \in \mathbb{R}^{d_1 \times d_0}$ and $\boldsymbol{U}_2 \in \mathbb{R}^{d_2 \times d_0}$ and $0 < d_0 \leq \min\{d_1, d_2\}$, such that each pair of components $(\mathbf{r}_1(i), \mathbf{r}_2(j))$ are maximally correlated if $i = j$ and uncorrelated otherwise. Let $(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11})$ and $(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22})$ be the mean and covariance matrices of $\mathbf{z}_1$ and $\mathbf{z}_2$, respectively, and $\boldsymbol{\Sigma}_{12}$ is their cross-covariance. Then CCA can be formulated as the optimization problem

$$\begin{aligned} &\max_{\boldsymbol{U}_1, \boldsymbol{U}_2} \operatorname{tr}[\boldsymbol{U}_1^\top \boldsymbol{\Sigma}_{12} \boldsymbol{U}_2] \\ &\boldsymbol{U}_1^\top \boldsymbol{\Sigma}_{11} \boldsymbol{U}_1 = \boldsymbol{U}_2^\top \boldsymbol{\Sigma}_{22} \boldsymbol{U}_2 = \mathbf{I}_{d_0} \end{aligned} \tag{3.3}$$

Given $(\boldsymbol{v}_{1i}, \boldsymbol{v}_{2i}),\ i \in [1, ..., d_0]$ as the pairs of left and right singular vectors corresponding to d_0 largest singular values, $p_i\ i \in [1, ..., d_0]$, of the correlation matrix $\boldsymbol{\Sigma}_{11}^{-1/2} \boldsymbol{\Sigma}_{12} \boldsymbol{\Sigma}_{22}^{-1/2}$, the solution to the CCA problem is given by $(\boldsymbol{u}_{1i}, \boldsymbol{u}_{2i}) = (\boldsymbol{\Sigma}_{11}^{-1/2} \boldsymbol{v}_{1i}, \boldsymbol{\Sigma}_{22}^{-1/2} \boldsymbol{v}_{2i}),\ i \in [1, ..., d_0]$ where $(\boldsymbol{u}_{1i}, \boldsymbol{u}_{2i})$, also called *canonical pairs of directions*, form the columns of $(\boldsymbol{U}_1, \boldsymbol{U}_2)$ and $\boldsymbol{P}_{d_0} = \operatorname{diag}([p_0, ..., p_{d_0}])$ is the diagonal *matrix of canonical correlations*.

Bach and Jordan [2005] and Browne [1979] proposed a probabilistic generative interpretation to the classical CCA problem that reveals the shared latent representation explicitly. An extension of their results to a more flexible model can be expressed as follows.

Theorem 3.2 *Assume the probabilistic generative model for the graphical model in Figure 3.1(b) as:*

$$\begin{aligned} \boldsymbol{\phi} &\sim \mathcal{N}(\boldsymbol{\mu}_0, \mathbf{I}_{d_0}), \quad 0 < d_0 \leq \min\{d_1, d_2\} \\ \mathbf{z}_1 | \boldsymbol{\phi} &\sim \mathcal{N}(\boldsymbol{W}_1 \boldsymbol{\phi} + \boldsymbol{\mu}_{\epsilon_1}, \boldsymbol{\Psi}_1), \quad \boldsymbol{W}_1 \in \mathbb{R}^{d_1 \times d_0}, \boldsymbol{\Psi}_1 \succcurlyeq 0 \\ \mathbf{z}_2 | \boldsymbol{\phi} &\sim \mathcal{N}(\boldsymbol{W}_2 \boldsymbol{\phi} + \boldsymbol{\mu}_{\epsilon_2}, \boldsymbol{\Psi}_2), \quad \boldsymbol{W}_2 \in \mathbb{R}^{d_2 \times d_0}, \boldsymbol{\Psi}_2 \succcurlyeq 0 \end{aligned} \tag{3.4}$$

where $\boldsymbol{\phi}$ is the shared latent representation. The maximum likelihood estimate of the parameters of this model can be expressed in terms of the canonical

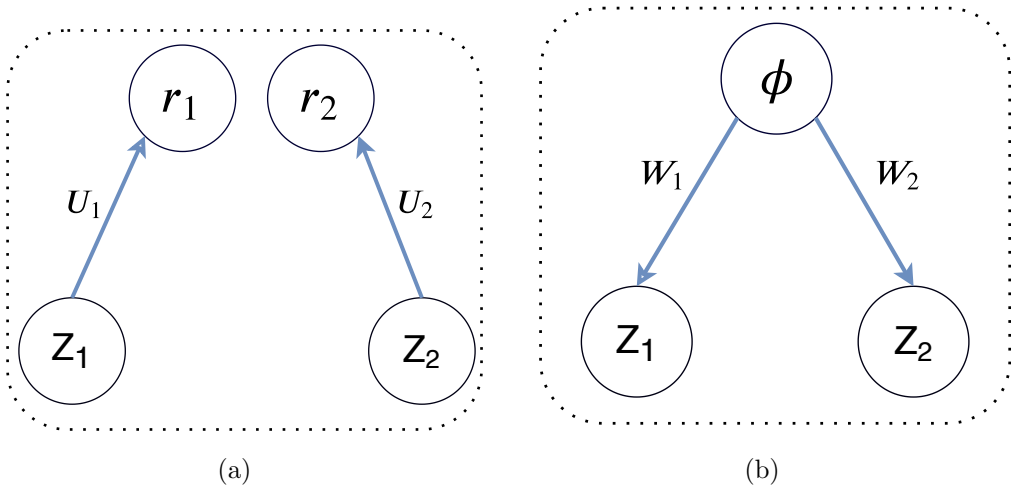


Figure 3.1: (a) Graphical representation of the CCA model, and (b) graphical representation of the probabilistic CCA model.

correlation directions as

$$\hat{\boldsymbol{W}}_1 = \boldsymbol{\Sigma}_{11}\boldsymbol{U}_1\boldsymbol{M} \tag{3.5}$$

$$\hat{\boldsymbol{W}}_2 = \boldsymbol{\Sigma}_{22}\boldsymbol{U}_2\boldsymbol{M}$$

$$\hat{\boldsymbol{\Psi}}_1 = \boldsymbol{\Sigma}_{11} - \hat{\boldsymbol{W}}_1\hat{\boldsymbol{W}}_1^\top$$

$$\hat{\boldsymbol{\Psi}}_2 = \boldsymbol{\Sigma}_{22} - \hat{\boldsymbol{W}}_2\hat{\boldsymbol{W}}_2^\top$$

$$\hat{\boldsymbol{\mu}}_{\epsilon_1} = \boldsymbol{\mu}_1 - \hat{\boldsymbol{W}}_1\boldsymbol{\mu}_0$$

$$\hat{\boldsymbol{\mu}}_{\epsilon_2} = \boldsymbol{\mu}_2 - \hat{\boldsymbol{W}}_2\boldsymbol{\mu}_0$$

where $\boldsymbol{M} = \boldsymbol{P}_{d_0}^{1/2}\boldsymbol{R}$ is the square root of matrix $\boldsymbol{P}_{d_0}$ and $\boldsymbol{R}$ is an arbitrary rotation matrix and the residual errors terms can be defined as $\boldsymbol{\epsilon}_1 := \mathbf{z}_1 - \boldsymbol{W}_1\boldsymbol{\phi}$ and $\boldsymbol{\epsilon}_2 := \mathbf{z}_2 - \boldsymbol{W}_2\boldsymbol{\phi}$. This probabilistic graphical model induces conditional independence of $\mathbf{z}_1$ and $\mathbf{z}_2$ given $\boldsymbol{\phi}$. The parameter $\boldsymbol{\mu}_0$ is not identifiable by maximum likelihood.

Proof: See Appendix B.1. ■

In contrast to the results in [Bach and Jordan, 2005] where $\boldsymbol{\mu}_0 = \mathbf{0}$, here we introduce $\boldsymbol{\mu}_0$ as an extra degree of freedom. We will see that this parameter plays an important role in optimizing the upper bound on the likelihood, and

also in the inference of the shared representation of deep probabilistic CCA. We will also derive an analytical form to identify it based on the parameters of the probabilistic multi-view layer. It can be easily verified that $E(\boldsymbol{\phi}|\mathbf{z}_1) - \boldsymbol{\mu}_0$ and $E(\boldsymbol{\phi}|\mathbf{z}_2) - \boldsymbol{\mu}_0$ lie in the same subspace spanned by the CCA projections, $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively.

3.3.1 Generalization to arbitrary number of views

As an extension to an arbitrary number of views for probabilistic CCA, [Archambeau and Bach, 2009] proposed a general probabilistic model as follows:

$$\begin{aligned} \mathbf{z}_m &= \boldsymbol{W}_m\boldsymbol{\phi}_0 + \boldsymbol{T}_m\boldsymbol{\phi}_m + \boldsymbol{\mu}_m + \nu_m, \\ \nu_m &\sim \mathcal{N}(\mathbf{0}, \tau_m^{-1}\mathbf{I}_{d_m}), \\ \boldsymbol{W}_m &\in \mathbb{R}^{d_m \times d_0}, \boldsymbol{T}_m \in \mathbb{R}^{d_m \times q_m}, \forall m \in \{1, ..., M\} \end{aligned} \tag{3.6}$$

where $\{\boldsymbol{\mu}_m\}_{m=1}^M$ and $\{\nu_m\}_{m=1}^M$ are the view specific offsets and residual errors, respectively. This model can also be viewed as a multibattery factor analysis (MBFA) [Klami et al., 2014, Browne, 1980] in the statistics literature, which describes the statistical dependence between all the views by a single shared latent vector, $\boldsymbol{\phi}_0$, and the factor loading matrices $\boldsymbol{W}_m$, and also explains away the view-specific variations by factors that are private to each view, $\boldsymbol{\phi}_m$ with factor loading $\boldsymbol{T}_m$. Restricting to a single view, this model includes the probabilistic factor analysis as a special case if the prior on the view-specific factor is multivariate independent Gaussian, and reduces to probabilistic PCA if the prior is also isotropic. Archambeau and Bach [2009] followed a Bayesian approach to the linear generative model (3.6) and proposed a variational Expectation-Maximization algorithm to estimate the model parameters.

Explaining the view-specific variations by the variance matrices for each

view, we can represent the probabilistic multi-view model as

$$\begin{aligned} \boldsymbol{\phi} &\sim \mathcal{N}(\boldsymbol{\mu}_0, \mathbf{I}_{d_0}), \\ \mathbf{z}_m|\boldsymbol{\phi} &\sim \mathcal{N}(\boldsymbol{W}_m\boldsymbol{\phi} + \boldsymbol{\mu}_{\epsilon_m}, \boldsymbol{\Psi}_m), \\ \boldsymbol{W}_m &\in \mathbb{R}^{d_m \times d_0}, \boldsymbol{\Psi}_m \succcurlyeq 0, \ \forall m \in \{1, ..., M\} \end{aligned} \tag{3.7}$$

Where the latent factor $\boldsymbol{\phi}$ captures the common variations between all the views. This results in an appropriate candidate for subspace clustering where the extracted common underlying representation can be deployed to obtain the cluster memberships. Let $(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_{mm})$ be the mean and covariance matrices of $\mathbf{z}_m$. Inspired by the maximum likelihood solution of probabilistic CCA in Theorem 3.2 and [Bach and Jordan, 2005], that reformulate the parameter estimate for the probabilistic model based on the classical CCA solutions, we can propose the following system of equation for the parameters of the probabilistic multi-view model

$$\begin{aligned} \boldsymbol{W}_m &= \boldsymbol{\Sigma}_{mm}^{1/2}\boldsymbol{V}_m\boldsymbol{P}_{d_0}^{1/2}\boldsymbol{R} \\ \boldsymbol{\Psi}_m &= \boldsymbol{\Sigma}_{mm} - \boldsymbol{W}_m\boldsymbol{W}_m^\top \\ \boldsymbol{\mu}_{\epsilon_m} &= \boldsymbol{\mu}_m - \boldsymbol{W}_m\boldsymbol{\mu}_0 \end{aligned} \tag{3.8}$$

Where $\boldsymbol{P}_{d_0} = \text{diag}([p_0, ..., p_{d_0}])$ with diagonal entries $p_j \in [0., 1.]$ and $\boldsymbol{V}_m$ are composed of orthonormal vectors $\{\boldsymbol{v}_{mi}\}$. To simplify the model, we assume $\boldsymbol{V}_m = \boldsymbol{V}, \forall m \in \{1, ..., M\}$. The equations in (3.8) reduces to maximum likelihood estimate of PCCA for $m = 2$ views, hence can be viewed as an extension of PCCA for multi-view with more than two views. Defining the correlation matrix as $\boldsymbol{C}_{lm} := \boldsymbol{\Sigma}_{ll}^{-1/2}\boldsymbol{\Sigma}_{lm}\boldsymbol{\Sigma}_{mm}^{-1/2}$, equations in (3.8) imply that $\boldsymbol{P}_{d_0}$ and $\boldsymbol{V}$ are formed by the singular value and singular vectors of the correlation matrix, respectively, *i.e.* $\boldsymbol{C}_{lm} = \boldsymbol{V}\boldsymbol{P}_{d_0}\boldsymbol{V}^\top$. Therefore, analogous to the ML solution of PCCA, $\boldsymbol{P}_{d_0}$ and $\boldsymbol{\Sigma}_{mm}^{-1/2}\boldsymbol{V}_m$ can be interpreted as matrices of *canonical correlations* and *canonical directions*. This also implies that all the pairs of the views have similar correlation matrix.

In the following section, an analytical form is presented to recover $\{\boldsymbol{\mu}_0, \boldsymbol{\mu}_{\epsilon_m}\}_{m=1}^{M}$ based on the moments of the views. We will also provide a simple treatments for obtaining $\boldsymbol{V}$ and $\boldsymbol{R}$. As a consequence, given the first and second order moments of the views together with the diagonal matrix of canonical correlations $\boldsymbol{P}_{d_0}$, one can infer the rest of the parameters for the multi-view generative model in (3.7). This, in fact, simplifies the variational inference network to learn a compact set of parameters.

It is worth noting that, although the deep generative model is built upon a single shared latent factor (and also a single correlation matrix to specify the relationship between all the views), it can be seen that the contribution of the shared factor in mth view is controlled by the factor loading $\boldsymbol{W}_m$ that is, in turn, a function of $\boldsymbol{P}_{d_0}$ and the view specific parameter $\boldsymbol{\Sigma}_{mm}$. Thus, the shared factor does not equally influence the views but instead its effect on each view varies by the strength of its projection, $\boldsymbol{W}_m\boldsymbol{\phi}$, which results in dissimilar cross-covariances $\boldsymbol{\Sigma}_{ml}$ for each pair $m \neq l$. This property, in fact, offers flexibility to model uneven dependencies between different subsets of views which is crucial for expressive multi-view modeling when $M > 2$.

Remark To specify all the pairwise correlations, $M(M-1)/2$ factors are required, on top of the view specific factors; training such a complex model is prohibitively expensive and seems not necessary for some tasks such as subspace clustering. To deal with this problem, group factor analysis (GFA) [Klami et al., 2014] extends the MBFA by applying structured sparsity regularizer on the factor loadings, hence offering a more flexible and interpretable model than MBFA by describing the relationship between subsets of views by subsets of factors. In contrast to GFA, here we are interested in a latent representation that best explains the common underlying variation between all of the views such as cluster membership in multi-modal dataset.

Although constraining the observation models to the classical linear model (3.4) offers closed form inference for the latent variable(s), as well as efficient training algorithms, the resulting expressiveness is very limited for modeling complex data distributions. On the other hand, the generative descriptions of

the probabilistic models in general, and the probabilistic multi-view models (3.4) and (3.6) in particular, can be extended naturally as the building blocks of more complex hierarchical models [Klami et al., 2013]. We can therefore append deep generative networks, known to be powerful techniques for increasing modeling capacity and improving its expressiveness, on top of the linear probabilistic model to obtain a combined model, which we denote as *deep probabilistic CCA* or a *deep probabilistic multi-view* network. A graphical representation of this model is depicted in Figure 3.2. Let $\mathbf{x} := \{\mathbf{x}_m \in \mathbb{R}^{d'_m}\}_{m=1}^{M}$ denote the collection of observations of all views and $\mathbf{z} := \{\boldsymbol{\phi} \in \mathbb{R}^{d_0}\} \cup \{\mathbf{z}_m \in \mathbb{R}^{d_m}\}_{m=1}^{M}$ be the collection of the shared latent representation and latent variables corresponding to each view. The *latent linear probabilistic CCA layer* of the formulation presented in (3.4) (or the *latent linear probabilistic multi-modal layer* in (3.7)) models the linear cross-correlation between all latent variables $\{\mathbf{z}_m\}_{m=1}^{M}$ in the latent space, while the nonlinear generative *observation networks*, also called the *decoders* in the context of variational auto-encoders, are responsible for expressing the complex variations of each view. The observation models are described by deep neural networks $p_{\theta_m}(\mathbf{x}_m|\mathbf{z}_m) = g_m(\mathbf{z}_m; \theta_m)$ with the set of model parameters $\theta = \{\theta_m\}_{m=1}^{M}$. In the following, an approximate variational inference approach is presented for training such a deep generative multi-view model.

3.3.2 Variational inference

To obtain the maximum likelihood estimate of the model parameters, it is desirable to maximize the marginal data log-likelihood averaged on the dataset $\mathcal{D} = \{x^{(i)}\},\ i = 1, .., N$, that can be expressed as

$$\log p_\theta(\mathbf{X}) = \frac{1}{N} \sum_{i=1}^{N} \log p_\theta(\boldsymbol{x}^{(i)}) \simeq \mathbb{E}_{\mathbf{x} \sim \hat{P}_{data}}[\log p_\theta(\mathbf{x})]$$

This objective requires marginalization over all latent variables which entails computing the expectation of the likelihood function $p_\theta(\mathbf{x}|\mathbf{z})$ over the prior

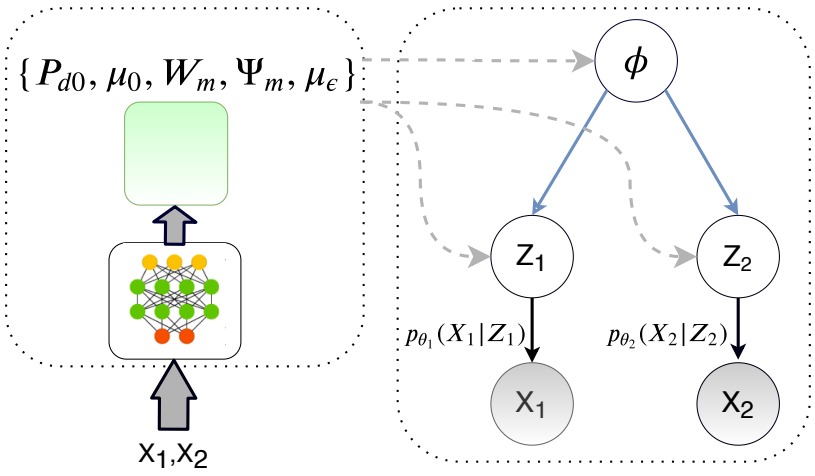


Figure 3.2: Graphical representation of the deep probabilistic CCA model, where the blue edges belong to latent linear probabilistic CCA model and the black edges represent the deep nonlinear observation networks (decoders) $p_{\theta_m}(\mathbf{x}_m|\mathbf{z}_m) = g_m(\mathbf{z}_m; \theta_m)$. Shaded nodes denotes observed views and dashed line represent the stochastic samples drawn from the approximate posteriors.

distribution on the set of latent variables, $p(\mathbf{z})$. The marginalization is typically intractable for complex models. One work around is to follow the variational inference principle [Jordan et al., 1999], by introducing an approximate posterior distribution $q_\eta(\mathbf{z}|\mathbf{x})$ — also known as *variational inference network* in the context of *amortized variational inference* and is often modeled by deep NNs with model parameters η — then maximize the resulting variational lower bound on the marginal log-likelihood

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\eta}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{\mathrm{KL}}[q_\eta(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})] \tag{3.9}$$

This approach has recently attained renewed interest and studied extensively, and, due to its success in training deep generative models, is considered a default, flexible statistical inference method [Rezende et al., 2014, Kingma and Welling, 2013]. This bound, also known as the *evidence lower bound (ELBO)*, can be decomposed into two main terms: the first, the expectation of the log-likelihood function $\log p_\theta(\mathbf{x}|\mathbf{z})$, is known as the *negative reconstruction error*. The conditional independence structure of the deep generative multimodal model implies that the likelihood function can be factored, allowing the negative

reconstruction error to be expressed as

$$\mathbb{E}_{q_\eta}[\log p_\theta(\mathbf{x}|\mathbf{z})] = \sum_{m=1}^{M} \mathbb{E}_{q_\eta}[\log p_{\theta_m}(\mathbf{x}_m|\mathbf{z}_m)].$$

Although the expectations above do not typically provide a closed analytical form, they can be approximated using Monte Carlo estimation by drawing L random samples from the approximate posterior $q_\eta(\mathbf{z}|\mathbf{x})$ for each data point $\mathbf{x} = \boldsymbol{x}^{(i)}$. [2]

The second term in the ELBO is the *KL divergence* between the approximate posterior and the prior distribution of the latent variables, which acts as a regularizer that injects prior knowledge about the latent variable into the learning algorithm. Considering the conditional independence of the latent variables $\{\mathbf{z}_m|\boldsymbol{\phi}\}$ induced by the probabilistic graphical model of latent linear layer (3.4), the approximate posterior of the set of latent variables can be factorized as $q_\eta(\mathbf{z}|\mathbf{x}) = q_\eta(\boldsymbol{\phi}|\mathbf{x}) \prod_{m=1}^{M} q_\eta(\mathbf{z}_m|\boldsymbol{\phi}, \mathbf{x})$ therefore, the KL divergence term can be decomposed into

$$\begin{aligned} D_{\mathrm{KL}}[q_\eta(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})] =& D_{\mathrm{KL}}[q_\eta(\boldsymbol{\phi}|\mathbf{x})\|p(\boldsymbol{\phi})] + \\ & \sum_{m=1}^{M} D_{\mathrm{KL}}[q_\eta(\boldsymbol{\epsilon}_m|\mathbf{x})\|p(\boldsymbol{\epsilon}_m)] \end{aligned} \tag{3.10}$$

The detailed derivation can be found in appendix B.2.

We model the variational approximate posteriors by joint multivariate Gaussian distributions with marginal densities $q_\eta(\mathbf{z}_m|\mathbf{x}_m) = \mathcal{N}(\mathbf{z}_m; \boldsymbol{\mu}_m(\mathbf{x}_m), \boldsymbol{\Sigma}_{mm}(\mathbf{x}_m))$, which are assumed for simplicity to be elementwise independent per each view, so having diagonal covariance matrices $\boldsymbol{\Sigma}_{mm} = \mathrm{diag}(\boldsymbol{\sigma}_m^2(\mathbf{x}_m))$, $\boldsymbol{\sigma}_m \in \mathbb{R}^{d_m}$. The cross correlation specified by canonical correlation matrix $\boldsymbol{P}_{d_0} = \mathrm{diag}(\boldsymbol{p}(\mathbf{x})), \boldsymbol{p} \in \mathbb{R}^{d_0}$. The parameters of these variational posteriors are specified by separate deep neural networks, also called *encoders*. In this model, a set of encoders are used to output the view-specific moments $\{(\boldsymbol{\mu}_m, \boldsymbol{\sigma}_m^2) = f_m(\mathbf{x}_m; \eta_m)\}_{m=1}^{M}$, and

[2] This, indeed, leads to the Monte Carlo approximation of the gradient of the expected log-likelihood, required for stochastic gradient descent training [Rezende et al., 2014]

an encoder network describes the cross correlation $\boldsymbol{p} = f_0(\mathbf{x}^*; \eta_0)$. Depending on the application, $\mathbf{x}^*$ can be either one (or a subset) of the views, when only one (or a subset) of the views are available at the test time (*e.g.* in the multi-view setting where $\mathbf{x}^* = \mathbf{x}_1$), or a concatenation of all the views (*e.g.* in the multi-modal setting). Combined, the inference model is parameterized by $\eta = \{\eta_0\} \cup \{\eta_m\}_{m=1}^M$. Having obtained the moments of approximate posteriors, we can obtain the canonical directions and subsequently the parameters of the probabilistic CCA model, according to the results presented in Theorem 3.2. It is worth noting that the diagonal choices for covariance matrices $\{\boldsymbol{\Sigma}_{mm}\}_{m=1}^M$ simplify the algebraic operations significantly, resulting in a trivial SVD computation and matrix inversion required for CCA solution used in Theorem 3.2.[3] Consequently, one can also easily verify that the canonical pairs of directions will be $(\boldsymbol{u}_{1i}, \boldsymbol{u}_{2i}) = (\sigma_{1i}^{-1/2}\boldsymbol{e}^{(i)}, \sigma_{2i}^{-1/2}\boldsymbol{e}^{(i)})$ where $\boldsymbol{e}^{(i)}$ is the standard basis vector $[0, \ldots, 0, 1, 0, \ldots, 0]$ with a 1 at ith position. The same argument can be extended for the generalized probabilistic multi-view model and its parametrization equations in (3.8).

We assume isotropic multivariate Gaussian priors on the latent variables as $\boldsymbol{\phi} \sim \mathcal{N}(\mathbf{0}, \lambda_0^{-1}\mathbf{I})$, $\boldsymbol{\epsilon}_m \sim \mathcal{N}(\mathbf{0}, \lambda_m^{-1}\mathbf{I})$ and specify the approximate posteriors by Gaussian distributed vectors with diagonal covariances, as explained above, that result in closed form solutions for the KL divergence terms [Kingma and Welling, 2013] as

$$D_{\mathrm{KL}}[q_\eta(\boldsymbol{\phi}|\mathbf{x})\|p(\boldsymbol{\phi})] = \frac{1}{2}\lambda_0\|\boldsymbol{\mu}_0\|^2 + \frac{1}{2}\sum_{i=1}^{d_0}(\lambda_0 - \log\lambda_0 - 1)$$

$$D_{\mathrm{KL}}[q_\eta(\boldsymbol{\epsilon}_m|\mathbf{x})\|p(\boldsymbol{\epsilon}_m)] = \frac{1}{2}\lambda_m\|\boldsymbol{\mu}_{\epsilon_m}\|^2 + \frac{1}{2}\sum_{i=1}^{d_m}(\lambda_m\sigma_{mi}^2 - \log\lambda_m\sigma_{mi}^2 - 1)$$

Based on the above equations, in the following, we provide an analytical approach to optimally identify the mean of shared latent variable, $\boldsymbol{\mu}_0$, from the

[3] These types of simplifying assumption on the approximate posteriors have also been used in various deep variational inference models [Rezende et al., 2014, Kingma and Welling, 2013]. Although the representation power of such linear latent model is limited but using flexible enough deep generative models, that can explain away the complex nonlinear structures among the data, can justify these choices.

parameters of the model, which is not identifiable by likelihood maximization in theorem 3.2.

Lemma 3.1 *I) Rewriting the KL divergences with respect to the terms depending on the mean of latent factors give rise to the following optimization problem*

$$\min_{\boldsymbol{\mu}_0} \frac{1}{2}\lambda_0\|\boldsymbol{\mu}_0\|^2 + \frac{1}{2}\sum_{m=1}^{M}\lambda_m\|\boldsymbol{\mu}_{\epsilon_m}\|^2 + \mathcal{K} \tag{3.11}$$
$$s.t.\ \ \boldsymbol{\mu}_{\epsilon_m} = \boldsymbol{\mu}_m - \boldsymbol{W}_m\boldsymbol{\mu}_0, \quad \forall m \in \{1, ..., M\}$$

where $\mathcal{K}$ is sum of the terms not depending on the means. Solving this optimization problem results the optimal minimizer

$$\boldsymbol{\mu}_0^* = (\lambda_0\mathbf{I} + \sum_{m=1}^{M}\lambda_m\boldsymbol{W}_m^\top\boldsymbol{W}_m)^{-1}(\sum_{m=1}^{M}\lambda_m\boldsymbol{W}_m^\top\boldsymbol{\mu}_m). \tag{3.12}$$

Having obtained the optimal $\boldsymbol{\mu}_0^*$, one can compute the means of the view-specific factors, $\{\boldsymbol{\mu}_{\epsilon_m}\}_{m=1}^M$, subsequently.

Proof: See Appendix B.2 for the proof. ■

According to the inference network, the optimal $\boldsymbol{\mu}_0$ obtained via (3.12) is a function of all the views, which can be viewed as a type of data fusion in the latent space customized for the variational inference learning of our model. This makes it an appropriate choice for the multi-modal setting. On the other hand, in the multi-view setting we are interested in a solution that depends only on the primary view available at test time. To deal with this, we can solve a revised version of the optimization problem (3.11) by ignoring the terms that depend on the non-primary views, leading to the minimizer

$$\hat{\boldsymbol{\mu}}_0 = (\lambda_0\mathbf{I} + \lambda_1\boldsymbol{W}_1^\top\boldsymbol{W}_1)^{-1}\lambda_1\boldsymbol{W}_1^\top\boldsymbol{\mu}_1. \tag{3.13}$$

Remark As an alternative approach in the multi-view setting, one can train the model using the optimal inference based on both views in equation (3.12) while using the primary view-based estimate $\hat{\boldsymbol{\mu}}_0$ in (3.13) at the test time, but

our empirical studies showed that using the same inference as in (3.13) for both training and test time offers richer shared representation variable resulting in slightly better performance in the downstream tasks. This can be explained by the fact that the train and test samples are drawn from the same data distribution so the model learned by the training samples better describes the test samples.

Remark Another possible approach is to treat $\boldsymbol{\mu}_0$ as an extra parameter that is directly inferred by a deep NN, but this needs more NN layers to train and in practice we found this approach less efficient than the proposed optimal procedure.

We further assume that the rotation matrix $\boldsymbol{R}$ is identity in the solution to the probabilistic linear models (3.5), ((3.2), or (3.8)), while leaving it to the deep generative network to approximate the rotation. Specifically, in our neural network architecture, we select a fully connected first layer of the decoder to exactly mimic the rotation matrix.

In summary, the encoders, together with the parameterization of the model in (3.5), and/or (3.8) provide a variational inference network for the parameters of the *latent probabilistic multi-view model*, $\{\boldsymbol{P}_{d_0}(\mathbf{x}_1), \boldsymbol{\mu}_0, \boldsymbol{W}_m(\mathbf{x}_m), \boldsymbol{\Psi}_m(\mathbf{x}_m), \boldsymbol{\mu}_{\epsilon_m}\}_{m=1}^{M}$, as non-linear functions of the observations.

Drawing samples from the latent variables: Given the variational parameters of the latent probabilistic CCA model, one can draw samples of the latent factors $\{\boldsymbol{\phi}, \boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2\}$ from the approximate posteriors $\{q_\eta(\boldsymbol{\phi}), q_\eta(\boldsymbol{\epsilon}_1), q_\eta(\boldsymbol{\epsilon}_2)\}$, using a differentiable transformation based on the reparameterization trick [Kingma et al., 2019], and generate latent representations as $\mathbf{z}_1 = \boldsymbol{W}_1\boldsymbol{\phi} + \boldsymbol{\epsilon}_1$ and $\mathbf{z}_2 = \boldsymbol{W}_2\boldsymbol{\phi} + \boldsymbol{\epsilon}_2$, which are fed into the decoders to generate samples $\{\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2\}$ at the observation space. The conditional independence property of the probabilistic CCA implies that the produced latent samples are conditionally independent given the shared latent variable $\boldsymbol{\phi}$ while their cross correlation is captured by $\boldsymbol{\phi}$ and specified by the variational canonical correlation $\boldsymbol{P}_{d_0}$.

Therefore, the negative reconstruction error term can be stated as

$$\begin{aligned}
&\mathbb{E}_{q_\eta}[\log p_\theta(\mathbf{x}|\mathbf{z})] = \\
&\quad \mathbb{E}_{q_\eta(\boldsymbol{\phi}),q_\eta(\boldsymbol{\epsilon}_1)}[\log p_{\theta_1}(\mathbf{x}_1|\mathbf{z}_1 = \boldsymbol{W}_1\boldsymbol{\phi} + \boldsymbol{\epsilon}_1)] + \\
&\quad \mathbb{E}_{q_\eta(\boldsymbol{\phi}),q_\eta(\boldsymbol{\epsilon}_2)}[\log p_{\theta_2}(\mathbf{x}_2|\mathbf{z}_2 = \boldsymbol{W}_2\boldsymbol{\phi} + \boldsymbol{\epsilon}_2)].
\end{aligned}$$

3.3.3 Related work

To capture nonlinearity in multi-view data, several kernel-based methods have been proposed [Hardoon et al., 2004, Bach and Jordan, 2003]. Such methods, in general, require a large memory to store a massive amount of training data for the test phase. Kernel-CCA in particular requires an $N \times N$ eigenvalue decomposition which is computationally expensive for large datasets. To overcome this issue, some kernel approximation techniques based on random sampling of training data are proposed in [Williams and Seeger, 2001] and [Lopez-Paz et al., 2014]. Probabilistic non-linear multi-view learning has been considered in [Shon et al., 2006, Damianou et al., 2012]. As an alternative, deep neural networks (DNNs) offer powerful parametric models that can be trained for large pools of data using the recent advances in stochastic optimization algorithms. In the multi-view setting, a deep auto-encoder model, called (SplitAE), was proposed in [Ngiam et al., 2011] in which an encoder maps the primary view to a latent representation and two decoders are trained so that the reconstruction error of both views is minimized.

The classical CCA was extended to deep CCA (DCCA) in [Andrew et al., 2013] by replacing the linear transformations of both views with two deep nonlinear NNs, then learning the model parameters by maximizing the cross correlation between the nonlinear projections. DCCA is then extended to deep CCA autoencoder (DCCAE) in [Wang et al., 2015a] where autoencoders are leveraged to additionally reconstruct the inputs, hence introducing extra reconstruction error terms to the objective function. While DCCAE can improve representation learning over DCCA, empirical studies have shown that it tends to ignore the added reconstruction error terms, which results in poorly

reconstructed views [Wang et al., 2015a]. Training algorithms for these classical CCA-based methods require sufficiently large training batches to approximate the covariance matrices and the gradients. Moreover, they do not naturally provide an inference model to estimate the shared latent factor, nor do they enable generative sampling from the model in the input space, while also being restricted to the two-view setting. In contrast, the reconstruction error terms appear naturally in the objective for variational inference, the ELBO, and therefore play a fundamental role in training of the decoder hence, richer decoder and reconstruction are expected using the proposed variational autoencoders. Furthermore, the stochastic backpropagation method with small mini-batches has proven to be a standard and scalable technique for training deep variational autoencoders [Rezende et al., 2014]. Finally, the probabilistic multi-view model enables enforcing desired structures such as sparsity [Archambeau and Bach, 2009] by adopting a broader range of exponential family distributions for priors and approximate posteriors on the latent factors to capture while this property is not immediately apparent in the classical CCA-based variants.

As explained in section 3.3.2, in our proposed deep generative model, the effect of the shared factor on each view varies by the strength of its projection, $\boldsymbol{W}_m\boldsymbol{\phi}$, which results in more flexibility in modeling uneven dependencies between different subsets of views that is of significant importance for modeling more general multi-view cases with arbitrary number pf views. On the other hand, in the variational two-view autoencoders in [Tang et al., 2017, Wang et al., 2016], the shared latent representation equally contributes in both views, so these variational two-view methods can be viewed as special cases of the more generic model proposed here when the posterior factor loading $\{\boldsymbol{W}_m\}_{m=1}^2$ are substituted with identity matrix, hence, they are expected to offer lower flexibility. This can explain why they offer less expressive representation than DCCAE in some experimental studies.

More recently, different VAE based multi-modal deep generative models has been proposed that model the variational posterior of the shared latent variable given all modalities as the product of unimodal posteriors, namely product of experts (PoE) [Wu and Goodman, 2018] or as a weighted summation of

unimodal posteriors, namely mixture of experts (MoE) [Shi et al., 2019]. Refer to chapter 5 for possible future direction on applying these ideas to enhance the flexibility of the proposed variational PCCA.

3.4 Experiments

We empirically evaluate the representation learning performance of the proposed method and compare against well established baselines in two scenarios: I) when all views are available at training time but only a single view (the primary view) is available at test time, namely the multi-view setting, and II) all views are available at training and testing time, namely the multi-modal setting.

3.4.1 Multi-view experiments

Experimental design: For the experimental study, we used the two-view noisy MNIST datasets of [Wang et al., 2015a] and [Wang et al., 2016] created based on MNIST handwritten digits that consists of grayscale images of size 28×28 pixels with pixel values scaled to range $[0, 1]$. The first view of the dataset was synthesized by rotating each image at angles randomly sampled from uniform distribution $\mathcal{U}(-\pi/4, \pi/4)$ while the image of the second view was randomly sampled from the images with similar identity to the first view but not necessary the same image, then was corrupted by random uniform noise while the final value was truncated to remain in range $[0, 1]$. As a result of this procedure, both views are just sharing the same identity (label) of the digit but not the style of the handwriting as they are from arbitrary images in the same class. The training set was divided into training/validation subsets of length $50K/10K$ and the performance was measured on the $10K$ images in the test set.

To provide a fair comparison, we used neural network architectures with the same capacity as those used in [Wang et al., 2015a] and [Wang et al., 2016]. Accordingly, for the deep network models, all inference and decoding networks were composed of 3 fully connected nonlinear hidden layers of 1024 units, with

(a) (b)

Figure 3.3: (a) Sample images from two-view noisy MNIST dataset

`ReLU` gates used as the nonlinearity for all hidden units. The first and the second encoder specify $(\boldsymbol{\mu}_1, \text{diag}(\sigma_1^2)) = f_1(\mathbf{x}_1; \theta_1)$, $(\boldsymbol{\mu}_2, \text{diag}(\sigma_2^2)) = f_2(\mathbf{x}_2; \theta_2)$ with the variances specified by a `softplus` function, and an extra encoder modeling the canonical correlations $\text{diag}(p_i)$ using the `sigmoid` function as the output gate. Independent Bernoulli distributions and independent Gaussian distributions were selected to specify the likelihood functions of the first and the second view, respectively, with the parameters of each view being specified by its own decoder network; `sigmoid` functions were applied on outputs used to estimate the means of both views while the variances of the Gaussian variables were specified by `softplus` functions. To prevent over-fitting, stochastic drop-out [Srivastava et al., 2014] was applied to all the layers as a regularization technique. The ADAM optimizer [Kingma and Ba, 2014] was adopted for training the parameters of the deep neural networks. The details of the experimental setup and training procedure can be found in Appendix B.4.

To evaluate the learned representation, the discriminative and clustering tasks were examined on the shared latent variable. For the discriminative goal, a one-versus-one linear SVM classification algorithm was applied on the shared representation $\boldsymbol{\phi}$. The parameters of the SVM algorithm were tuned using the validation set and the classification error was measured on the test set.

Method	Error (%)	NMI (%)	ACC (%)
Linear CCA	19.6	56.0	72.9
SpliAE	11.9	69.0	64.0
KCCA	5.1	87.3	94.7
DCCA	2.9	92.0	97.0
DCCAE	2.2	93.4	97.5
VCCA	3.0	-	-
VCCA-private	2.4	-	-
VPCCA	**1.9**	**94.8**	**98.1**

Table 3.1: Performance of the downstream tasks for different multi-view learning algorithms on the noisy two-view MNIST digit images. Performance measures are classification error rate (the lower the better), normalized mutual information (NMI) and accuracy (ACC) of clustering (the higher the better) [Cai et al., 2005]. ***VPCCA****: multi-view setting,* i.e. *only primary view is available at the test time so* $\boldsymbol{\mu}_0$ *of equation (3.13) is used. The results of variational PCCA method are averaged over 3 trials where the results of the baseline methods are from [Wang et al., 2015a, 2016]. The baseline methods are* ***Linear CCA****: linear single layer CCA,* ***DCCA****: deep CCA [Andrew et al., 2013],* ***Randomized KCCA****: randomized kernel CCA approximation with Gaussian RBF kernels and random Fourier features [Lopez-Paz et al., 2014],* ***DCCAE****: deep CCA-Auto encoder [Wang et al., 2015a],* ***VCCA****: multi-view variational auto-encoder [Wang et al., 2016]* ***VCCA-private****: shared-private multi-view variational auto-encoder [Wang et al., 2016].*

We also performed spectral clustering [Von Luxburg, 2007] on the k-nearest-neighbor graph constructed from the shared representation. To comply with the experiments in [Wang et al., 2015a] the degree (number of neighbors) of the nodes was tuned in the set $\{5, 10, 20, 30, 50\}$ using the validation set, and k-means was used as the last step to construct a final partitioning into 10 clusters in the embedding space. The proposed deep probabilistic CCA is compared against available multi-view methods in terms of performance at the downstream tasks, reported in Table C.1, where the results highlight that the proposed variational model significantly improves representation learning from multi-view datasets.

Repeating the experiments in the multi-modal setting (*i.e.* both views available at test time) and using (3.12) to recover the mean of the shared latent variable significantly improves the performance of downstream tasks resulting in classification error=0.4% and clustering NMI=98.3% or ACC=99.4%. These findings support the merit of the proposed algorithm for successfully integrating

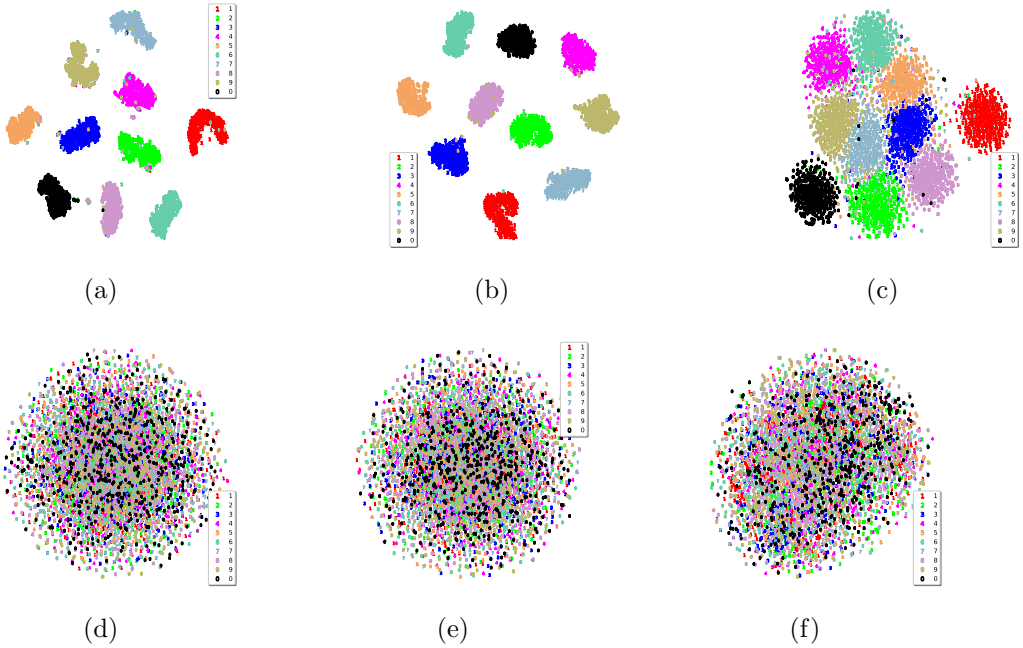


Figure 3.4: 2D t-SNE embedding of samples of the shared representation $\boldsymbol{\phi}$ are from models (a) VPCCA when only 1st view is available at test time, i.e. $\boldsymbol{\phi} \sim q_\eta(\boldsymbol{\phi}|\mathbf{x}_1)$*, (b) VPCCA-2v when both views are available at test time* i.e. $\boldsymbol{\phi} \sim q_\eta(\boldsymbol{\phi}|\mathbf{x}_1, \mathbf{x}_2)$*, (c) VCCA-private when only 1st view is available at test time,* i.e. $\boldsymbol{\phi} \sim q_\eta(\boldsymbol{\phi}|\mathbf{x}_1)$*. Moreover, 2D t-SNE embedding of samples of the residual error (view specific factor) of the first view $\boldsymbol{\epsilon}_1 \sim q_\eta(\boldsymbol{\epsilon}_1|\mathbf{x}_1)$ are from models (d) VPCCA (e) VPCCA-2v and (f) VCCA-private (the 1st private view).*

information from different modalities.

Figures 3.4 depict the 2D t-SNE embeddings of the shared latent representations and private factor of the 1st view for multi-view setting (VPCCA), multi-modal setting (VPCCA-2v when both views are available at test time) and VCCA-private [Wang et al., 2016]. They verify that the representation of the images of different classes are well separated in the shared latent space while VPCCA can separate the classes better; among them, VPCCA-2v results in the cleanest 2D embedding.

Reconstruction of the second view based on the primary view In some applications we are interested in estimation some of the views based on the available views at test time. In order to design such network, one can modify the variational inference of $\mathbf{z}_2$ so that its variance is specified as a function of the first view, *i.e.* $\boldsymbol{\Sigma}_{22} = \text{diag}(\sigma_2^2(\mathbf{x}_1))$, results that the factor loading of the second view, $\boldsymbol{W}_2$ depends only on the first view, hence we can

perform the reconstruction of the absent view at the test time.

3.4.2 Multi-modal clustering

An important and interesting application of the proposed deep generative model is in clustering multi-modal datasets which we evaluate in this set of experiments. Recently, a deep multi-modal subspace clustering method [Abavisani and Patel, 2018b] has successfully extended the idea of deep subspace clustering (DSC) [Ji et al., 2017] to multiple modalities. A key component of such approaches is applying a self-expressive layer on a non-linear mapping of the data obtained by deep auto-encoders, which represents the projection of data points as a linear combination of other data point projections. Although offering significant improvement in clustering performance for data lying in non-linear subspaces, such methods require a self-representation coefficient matrix of size $N \times N$ where N is the number of data points, making this approach prohibitively expensive for large datasets.

Datesets

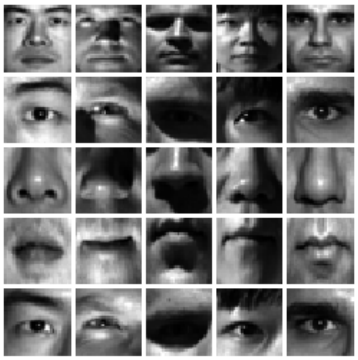
The clustering performance of the proposed method is evaluated on the following standard datasets. Samples from all modalities of these datasets are depicted in Figure 3.5.

Handwritten Digits: We chose two famous handwritten digits datasets MNIST [Y. LeCun, 1998] and USPS [Hull, 1994] that consist of grayscale digit images of size 28×28 pixels and 16×16 pixels, respectively. To make multi-modal dataset, each digit image in the MNIST dataset is paired with an arbitrary sample of the same digit identity but from USPS dataset. This process guarantees that the images of both modalities are just sharing the same identity (label) of the digit but not the style of the handwriting. The handwritten digits datasets were used for single-modal training and also for multi-modal, with $M = 2$, subspace clustering.

Multi-modal Facial Components: We also evaluated the proposed method on the multi-modal facial dataset used in [Abavisani and Patel, 2018a],

(a)

(b)



(c)

Figure 3.5: (a) Sample images form MNIST dataset, (b) and their corresponding samples from the second modality, drawn from USPS datasets. (c) Sample images from faces and face components from Extended Yale-B dataset; modalities are showed in different rows.

where the Extended Yale-B dataset [Lee et al., 2005] was used as the base and 4 facial components were extracted, by cropping eyes, nose and mouth, and formed 5 different modalities, including the whole face image. All modalities were resized to images of size 28×28 pixels. This dataset is composed of 64 frontal images of 38 individuals under different illuminations and is a standard dataset in subspace clustering studies. For this multi-modal data, we train the general deep probabilistic multi-view model (equation (3.7)) that extends the deep probabilistic CCA to arbitrary number of views.

Experimental design: To provide a fair comparison, the encoders and decoders in this set of experiments were defined by neural networks with similar architectures as those used in [Abavisani and Patel, 2018a], except that our model does not require the self-expressive layer, a linear fully connected layer with parameter matrix of size $N \times N$ coefficients where N is the number data points, which limits the total training size of the family of self-expressiveness based methods. This is a key advantage of the proposed model that significantly reduces the total number of parameters, especially for large input sizes. Thus, the proposed architecture is sufficiently scalable to take advantage of all the training samples.

Accordingly, the encoders (inference networks) of all modalities were com-

posed of convolutional NN (CNN) layers while the decoders (observation networks) were built of transposed convolution layers. `ReLU` gate was used as the nonlinearity for all the hidden units of the deep networks. The encoders specified $(\boldsymbol{\mu}_m, \text{diag}(\sigma_m^2)) = f_m(\mathbf{x}_m; \theta_m)$, where the variances were modeled by a `softplus` function. An extra encoder network modeled the canonical correlations, $\text{diag}(p_i)$, using the `sigmoid` function as the output gate. The observation likelihood functions of all the views, $p_{\theta_m}(\mathbf{x}_m|\mathbf{z}_m)$, were modeled by independent Bernoulli distributions with the mean parameter being specified by decoder networks, $g_m(\mathbf{z}_m; \theta_m)$; with `sigmoid` functions applied to estimate valid means for the distributions. To train the parameters of deep generative model, we used ADAM optimization [Kingma and Ba, 2014] with learning rate of .0002 and default hyper-parameters and minibatch size of 200 data points. Details of the model architecture and experimental setup together with more empirical results are presented in appendix B.4.

We observed that an optimal choice of the ratio of prior noise precision, λ_0/λ_i, can significantly improve the learned representation for the proposed model. This may be explained by the fact that adjusting the priors of the latent linear probabilistic layer can control the view specific factors to be flexible enough to capture the variations private to each view but restricted enough so as not to describe the relationships between the views. A related idea was elaborated in the formulation of group factor analysis [Klami et al., 2014]. In contrast, VCCA-private Wang et al. [2016] did not exhibit such behavior in our experiments and required less parameter tuning.

To estimate shared latent features, VPCCA used the optimal data fusion of (3.11) in the latent space while, in VCCA-private, we applied a dense linear layer on the outputs of the encoders, $\{f_m(\mathbf{x}_m; \theta_m)\}_{m=1}^M$, to estimate $\boldsymbol{\mu}_0$.

Clustering is, then, performed on the shared latent factor $\boldsymbol{\phi}$ using spectral clustering [Von Luxburg, 2007] on the k-nearest-neighbor graph, with the number of neighbors set to $k = 5$. As the last step, spectral clustering is used to discretize the real-valued representation in the embedding space to extract the final partitioning. Clustering performance are measured using clustering Accuracy rate (ACC), Normalized Mutual Information (NMI) [Cai et al., 2005]

(a) multi-modal clustering

	Digits			Extended Yale-B		
	ACC	NMI	ARI	ACC	NMI	ARI
CMVFC	47.6	73.56	38.12	66.84	72.03	40
TM-MSC	80.65	83.44	75.67	63.12	67.06	38.37
MSSC	81.65	85.33	77.36	80.3	82.78	50.18
MLRR	80.6	84.13	76.53	67.62	73.36	40.85
KMSSC	84.4	89.45	79.61	87.65	81.5	63.83
KMLRR	86.85	80.34	82.76	82.45	85.43	59.71
DMSC	95.15	92.09	90.22	99.22	98.89	98.38
VCCA-private	90.02	92.43	85.09	97.52	98.09	96.07
VPCCA	**98.78**	**96.72**	**97.35**	**99.72**	**99.56**	**99.22**

(b) unimodal clustering

	MNIST			USPS		
	ACC	NMI	ARI	ACC	NMI	ARI
SSC	67.5	71.64	57.03	37.5	36.61	28.4
LRR	67.4	66.51	58.33	44.35	35.18	32.11
DSC	**92.05**	**87.07**	**84.6**	72.15	74.73	65.47
DVFA	79.81	83.50	71.55	**90.09**	**88.80**	**83.94**

Table 3.2: Performance for different multi-modal clustering algorithms on single-modal, two-modal handwritten digits made from MNIST and USPS and multi-modal facial components extracted from Yale-B dataset. Performance metrics are clustering Accuracy rate (ACC), Normalized Mutual Information (NMI) [Cai et al., 2005] and Adjusted Rand Index (ARI) [Rand, 1971]; all measures are in percent and the higher means the better.
In Table (a), a multi-modal setting is considered and we assume that all modalities are available at test time so ***VPCCA*** *uses $\boldsymbol{\mu}_0$ of equation (3.12). In Table (b), a unimodal case is considered and the variational PCCA of (3.8) with $m = 1$ is applied which is called deep variational factor analysis (DVFA). The results of the variational PCCA method are averaged over 3 trials. The baseline subspace clustering methods are: SSC [Elhamifar and Vidal, 2013], LRR [Liu et al., 2010] and DSC [Ji et al., 2017] for single-modal datasets, and TM-MSC [Zhang et al., 2015], CMVFC [Cao et al., 2015], MSSC, MLRR , KMSSC, KMLRR [Abavisani and Patel, 2018b] and DMSC [Abavisani and Patel, 2018a] for multi-modal dataset. The results of the baseline methods are from [Abavisani and Patel, 2018a].*

and Adjusted Rand Index (ARI) [Rand, 1971] as performance metrics.

The clustering performance of the proposed method is compared against the well established subspace clustering methods SSC [Elhamifar and Vidal, 2013], LRR [Liu et al., 2010] and DSC [Ji et al., 2017] for single-modal datasets,

while TM-MSC [Zhang et al., 2015], CMVFC [Cao et al., 2015], MSSC, MLRR, KMSSC, KMLRR [Abavisani and Patel, 2018b] and DMSC [Abavisani and Patel, 2018a] that are used as the baselines methods for the multi-modal setting. The results summarized in Table 3.2 show that the proposed deep generative models offer superior clustering performance compared to the reference methods for most test cases. Specifically in the multi-modal setting, the proposed deep generative model sets new state-of-the-art which, subsequently, highlights that the proposed method can efficiently leverage the extra modalities and extract the common underlying information among the modalities, that is the cluster memberships. Extra experiments with multi-modal facial datasets when subset of modalities are missing at test time are presented in appendix B.3.

Chapter 4

Linear Dynamical System Identification

4.1 Introduction

Linear dynamical systems (LDS) provide a fundamental model for estimation and forecasting in discrete-time multi-variate time series. In an LDS, each observation is associated with a latent state; these unobserved states evolve as a Gauss-Markov process where each state is a linear function of the previous state plus noise. Such a model of a partially observed dynamical system has been widely adopted, particularly due to its efficiency for prediction of future observations using Kalman filtering.

Estimating the parameters of an LDS—sometimes referred to as system identification—is a difficult problem, particularly if the goal is to obtain the maximum likelihood estimate of parameters. Consequently, spectral methods from the subspace identification literature, based on moment-matching rather than maximum likelihood, have become popular. These methods provide closed form solutions, often involving a singular value decomposition of a matrix constructed from the empirical moments of observations [Moonen and Ramos, 1993, Van Overschee and De Moor, 1994, Viberg, 1995, Katayama, 2006, Song et al., 2010, Boots and Gordon, 2012]. The most widely used

such algorithms for parameter estimation in LDSs are the family of N4SID algorithms [Van Overschee and De Moor, 1994], which are computationally efficient and asymptotically consistent [Andersson, 2009, Hsu et al., 2012]. Recent evidence, however, suggests that these moment-matching approaches may suffer from weak statistical efficiency, performing particularly poorly with small sample sizes [Foster et al., 2012, Zhao and Poupart, 2014].

Maximum likelihood for LDS estimation, on the other hand, has several advantages. For example, it is asymptotically efficient under general conditions [Cramér, 1946, Ch.33], and this property often translates to near-minimax finite-sample performance. Further, maximum likelihood is amenable to coping with missing data. Another benefit is that, since the likelihood for exponential families and corresponding convex losses (Bregman divergences) are well understood [Banerjee et al., 2005], maximum likelihood approaches can generalize to a broad range of distributions over the observations. Similarly, other common machine learning techniques, such as regularization, can be naturally incorporated in a maximum likelihood framework, interpretable as maximum a posteriori estimation.

Unfortunately, unlike spectral methods, there is no known efficient algorithm for recovering parameters that maximize the marginal likelihood of observed data in an LDS. Standard iterative approaches are based on EM [Ghahramani and Hinton, 1996, Roweis and Ghahramani, 1999], which are quite slow [Roweis and Ghahramani, 1999] and have been observed to produce locally optimal solutions that yield poor results [Katayama, 2006]. A classical system identification method, called the prediction error method (PEM), is based on minimization of prediction error and can be interpreted as maximum likelihood estimation under certain distributional assumptions (e.g., Ch. 7.4 of Ljung 1999, Åström 1980). PEM, however, is prone to local minima and requires selection of a canonical parameterization, which can be difficult in practice and can result in ill-conditioned problems [Katayama, 2006].

In this chapter, we propose an alternative approach to LDS parameter estimation under exponential family observation noise. In particular, we reformulate the LDS as a two-view generative model, which allows us to

approximate the estimation task as a form of matrix factorization, and apply recent global optimization techniques for such models [Zhang et al., 2012, Yu et al., 2014]. To extend these previous algorithms to this setting, we provide a novel proximal update for the two-view approach that significantly simplifies the algorithm. Finally, for forecasting on synthetic and real data, we demonstrate that the proposed algorithm matches or outperforms N4SID, while scaling better with increasing sample size and data dimension.

4.2 Linear dynamical systems

We address discrete-time, time-invariant linear dynamical systems, specified as

$$\begin{aligned} \boldsymbol{\phi}_{t+1} &= \mathbf{A}\boldsymbol{\phi}_t + \eta_t \\ \mathbf{x}_t &= \mathbf{C}\boldsymbol{\phi}_t + \epsilon_t \end{aligned} \tag{4.1}$$

where $\boldsymbol{\phi}_t \in \mathbb{R}^k$ is the hidden state at time t; $\mathbf{x}_t \in \mathbb{R}^d$ is the observation vector at time t; $\mathbf{A} \in \mathbb{R}^{k \times k}$ is the dynamics matrix; $\mathbf{C} \in \mathbb{R}^{d \times k}$ is the observation matrix; η is the state evolution noise; and ϵ is the observation noise. The noise terms are assumed to be independent. As is common, we assume that the state evolution noise is Gaussian: $\eta \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\eta)$. We additionally allow for general observation noise to be generated from an exponential family distribution (e.g., Poisson). The graphical representation for this LDS is shown in Figure 4.1.

An LDS encodes the intuition that a latent state is driving the dynamics, which can significantly simplify estimation and forecasting. The observations typically contain only partial information about the environment (such as in the form of limited sensors), and further may contain noisy or even irrelevant observations. Learning transition models for such observations can be complex, particularly if the observations are high-dimensional. For example, in spatiotemporal processes, the data is typically extremely high-dimensional, composed of structured grid data; however, it is possible to extract a low-rank state-space that significantly simplifies analysis [Gelfand et al., 2010, Chapter 8]. Further, for forecasting, iterating transitions for such a low-rank state-space

can provide longer range predictions with less error accumulation than iterating with the observations themselves.

The estimation problem for an LDS involves extracting the unknown parameters, given a time series of observations $\mathbf{x}_1, \ldots, \mathbf{x}_T$. Unfortunately, jointly estimating the parameters $\mathbf{A}, \mathbf{C}$ and $\boldsymbol{\phi}_t$ is difficult because the multiplication of these variables typically results in a nonconvex optimization. Given the latent states $\boldsymbol{\phi}_t$, estimation of $\mathbf{A}$ and $\mathbf{C}$ is more straightforward, though there are still some issues with maintaining stability [Siddiqi et al., 2007]. There are some recent advances improving estimation in time series models using matrix factorization. White et al. [2015] provide a convex formulation for auto-regressive moving average models—although related to state-space models, these do not permit a straightforward conversion between the parameters of one to the other. Yu et al. [2015] factorize the observation into a hidden state and dictionary, using a temporal regularizer on the extracted hidden state—the resulting algorithm, however, is not guaranteed to provide an optimal solution due to the non-convexity of its objective function.

4.3 Two-view Formulation of LDS

In this section, we reformulate the LDS as a generative two-view model with a shared latent factor. In the following section, we demonstrate how to estimate the parameters of this reformulation optimally, from which parameter estimates of the original LDS can be recovered.

To obtain a two-view formulation, we re-express the two equations for the LDS as two equations for pairs of sequential observations. To do so, we multiply the state evolution equation in (4.1) by $\mathbf{C}$ and add ϵ_{t+1} to obtain $\mathbf{C}\boldsymbol{\phi}_{t+1} + \epsilon_{t+1} = \mathbf{CA}\boldsymbol{\phi}_t + \mathbf{C}\eta_t + \epsilon_{t+1}$; representing the LDS model as

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{E}\boldsymbol{\phi}_t + \epsilon'_{t+1} \\ \mathbf{x}_t &= \mathbf{C}\boldsymbol{\phi}_t + \epsilon_t \end{aligned} \tag{4.2}$$

where we refer to $\mathbf{E} := \mathbf{CA}$ as the factor loading matrix and $\epsilon'_{t+1} := \mathbf{C}\eta_t + \epsilon_{t+1}$

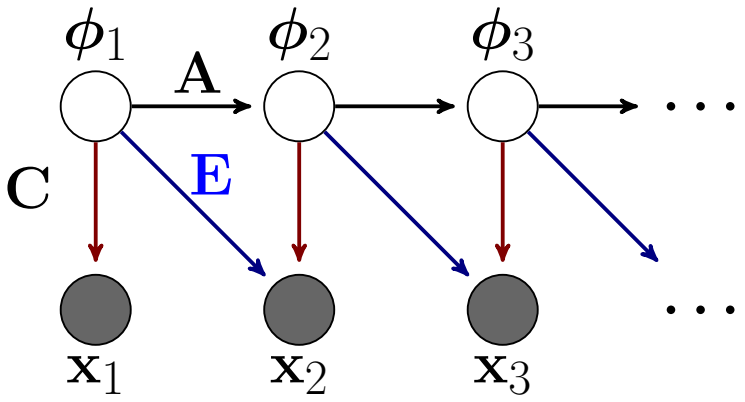


Figure 4.1: Graphical representation for the standard LDS formulation and the corresponding two-view model. The two-view formulation is obtained by a linear transformation of the LDS model. The LDS model includes only parameters $\mathbf{C}$ *and* $\mathbf{A}$ *and the two-view model includes parameters* $\mathbf{C}$ *and* $\mathbf{E} = \mathbf{CA}$*, where* $\mathbf{A}$ *can be extracted from* $\mathbf{E}$ *after* $\mathbf{C}$ *and* $\mathbf{E}$ *are estimated.*

as the noise of the second view. We then have a two-view problem where we need to estimate parameters $\mathbf{E}$ and $\mathbf{C}$. Since the noise components ϵ_t and ϵ'_{t+1} are independent, the two views $\mathbf{x}_t$ and $\mathbf{x}_{t+1}$ are conditionally independent given the shared latent state $\boldsymbol{\phi}_t$. To obtain the maximum likelihood estimate of the LDS, it is desirable to maximize the marginal data log-likelihood that can be factorized according to the chain rule of probability as

$$\log p(\mathbf{x}_{1:T}|C, E) = \sum_{t=1}^{T} \log p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, C, E) \tag{4.3}$$

To deal with the intractability of the marginalization over the latent state, one can assume a Dirac delta posterior distribution on $\boldsymbol{\phi}_{1:T}$ – that was adopted from *hard EM* or *Viterbi EM* method [Brown et al., 1993] – therefore resulting in a simpler objective for the maximum likelihood problem[1] for the two-view formulation:

$$\max_{C,E,\Phi} \log p(\mathbf{x}_{1:T}|\boldsymbol{\phi}_{0:T}, C, E) = \max_{C,E,\Phi} \sum_{t=1}^{T} \log p(\mathbf{x}_t|\boldsymbol{\phi}_{t-1}, \boldsymbol{\phi}_t, C, E) \tag{4.4}$$

where we used the fact that, given the hidden states, the observations are

[1]This is, indeed, maximizing a lower bound on the marginal log-likelihood instead. See [White et al., 2015] for the detailed derivation.

conditionally independent. The log-likelihood equation 4.4 is equivalent to the original LDS, but is expressed in terms of the distribution $p(\mathbf{x}_t|\boldsymbol{\phi}_{t-1}, \boldsymbol{\phi}_t, C, E)$, where the probability of an observation increases if it has high probability under both $\boldsymbol{\phi}_{t-1}$ and $\boldsymbol{\phi}_t$. The graphical depiction of the LDS and its implied two-view model is illustrated in Figure 4.1.

4.3.1 Relaxation

To tackle the estimation problem, we reformulate the estimation problem for this equivalent two-view model of the LDS. Note that according to the two-view model (4.2), the conditional distribution (4.4) can be expressed as $p(\mathbf{x}_t|\boldsymbol{\phi}_{t-1}, \boldsymbol{\phi}_t, C, E) = p(\mathbf{x}_t|\mathbf{E}\boldsymbol{\phi}_{t-1}) = p(\mathbf{x}_t|\mathbf{C}\boldsymbol{\phi}_t)$. Substituting each of these in the summation of equation 4.4 would result in a factor loading model that ignores the temporal correlation among data; therefore, to take the system dynamics into account we consider a balanced averaging of both as [2]

$$\arg\max_{C,E,\Phi} \sum_{t=1}^{T} \log p(\mathbf{x}_t|\boldsymbol{\phi}_{t-1}, \boldsymbol{\phi}_t, C, E) = \tag{4.5}$$

$$\arg\max_{C,E,\Phi} \sum_{t=1}^{T} \log p(\mathbf{x}_t|\mathbf{E}\boldsymbol{\phi}_{t-1}) + \log p(\mathbf{x}_t|\mathbf{C}\boldsymbol{\phi}_t) \tag{4.6}$$

This is equivalent to a natural choice on the likelihood as $p(\mathbf{x}_t|\boldsymbol{\phi}_{t-1}, \boldsymbol{\phi}_t, C, E) \propto p(\mathbf{x}_t|\mathbf{E}\boldsymbol{\phi}_{t-1})p(\mathbf{x}_t|\mathbf{C}\boldsymbol{\phi}_t)$, *i.e.* the likelihood of an observation increases if it has high conditional likelihood given both $\boldsymbol{\phi}_{t-1}$ and $\boldsymbol{\phi}_t$.

Bregman divergences and exponential family distributions: We can model the generalized linear observation model by exponential family distributions parameterized by θ, defined as $p_F(\mathbf{x}|\theta) = \exp(\mathbf{x}^\top\theta - F(\theta))p_0(\mathbf{x})$, that is specified by the *potential function* $F : \mathbb{R}^d \to \mathbb{R}$. The exponential families encompass many well-know distributions such as the Gaussian, Bernoulli, Poisson,

[2]The balanced averaging can be generalized to a convex combination of the log-likelihood which adds a flexibility to the problem that can be tuned to improve performance. However, we found that the simple balanced combination renders the best experimental performance in most cases.

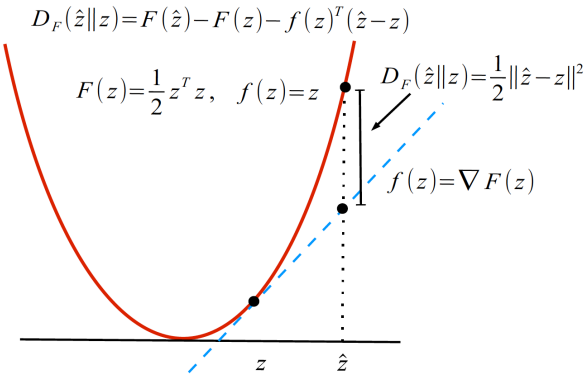


Figure 4.2: An illustration of the Bregman divergence of the Gaussian distribution defined as the difference between its corresponding potential function $F(\hat{\mathbf{z}}) = \frac{1}{2}\mathbf{z}^\top\mathbf{z}$ and its linear approximation by the first order Taylor expansion. The figure is from [White, 2009].

gamma, beta and Weibull while can also be used to approximate a broad range of distributions. Now, let's review a general family of convex losses known as Bregman divergences. For any strictly convex differentiable potential function F, the *Bregman divergence* $D_F(\hat{\mathbf{z}}\|\mathbf{z})$ is defined as the difference between $F(\hat{\mathbf{z}})$ and its linear approximation by the first order Taylor expansion around $\mathbf{z}$ evaluated at $\hat{\mathbf{z}}$, so it can be formally written as $D_F(\hat{\mathbf{z}}\|\mathbf{z}) := F(\hat{\mathbf{z}}) - F(\mathbf{z}) - \mathbf{f}(\mathbf{z})^\top(\hat{\mathbf{z}}-\mathbf{z})$ where $\mathbf{f} = \nabla F$ is called the *transfer function* associated with F [Banerjee et al., 2005]. It is clear from the definition that the Bregman divergence is a convex function in the first argument. It can be shown that maximizing the log-likelihood of the natural exponential family distributions with respect to their parameters θ reduces to the Bregman divergences[3]. As an example, a Gaussian distribution, that can be expressed by an exponential family defined by the potential function $F(\mathbf{z}) = \frac{1}{2}\mathbf{z}^\top\mathbf{z} = \frac{1}{2}\|\mathbf{z}\|_2^2$ with $f = \mathbf{z}$, results in the Euclidean loss $D_F(\hat{\mathbf{z}}\|\mathbf{z}) = \frac{1}{2}||\hat{\mathbf{z}} - \mathbf{z}||_2^2$. See figure 4.2 for the illustration of this loss as a Bregman divergence. Consequently, the log-likelihood in (4.5) can be expressed as

$$\arg\min_{C,E,\Phi} \sum_{t=1}^{T} D_F(\mathbf{E}\boldsymbol{\phi}_{t-1}||f^{-1}(\mathbf{x}_t)) + D_F(\mathbf{C}\boldsymbol{\phi}_t||f^{-1}(\mathbf{x}_t))$$

[3]This class includes many well-known losses, for example the Euclidean loss and Mahalanobis distance correspond to Gaussian distribution (with $F(\mathbf{z}) = \frac{1}{2}\mathbf{z}^\top\boldsymbol{\Sigma}^{-1}\mathbf{z}$ and $f(\mathbf{z}) = \boldsymbol{\Sigma}^{-.5}\mathbf{z}$), logistic loss or cross entropy corresponds to Bernoulli distribution (with sigmoid transfer $f(\mathbf{z}) = (1+\exp(\mathbf{z}))^{-1}$) and relative entropy corresponds to multinoulli (categorical) distributions (with softmax transfer $f(\mathbf{z}) = \exp(\mathbf{z})(\exp(\mathbf{1}^\top\mathbf{z}))^{-1}$). Consult [Banerjee et al., 2005, White, 2009] for a complete overview of this correspondence.

Each Bregman divergence term can be interpreted as the fitness measure for each view. The above derivation could be extended to different variance terms for ϵ and ϵ', which would result in different weights on the two Bregman divergences above. Further, we could also allow different exponential families (hence different Bregman divergences) for the two distributions; however, there is no clear reason why this would be beneficial over simply selecting the same exponential family, since both describe $\mathbf{x}_t$. In this work, therefore, we will explore a balanced loss, with the same exponential family for each view.

In order to obtain a low rank solution, one can relax the hard rank constraint and employ the block norm $\|\mathbf{\Phi}\|_{2,1} = \sum_{j=1}^{k} \|\mathbf{\Phi}_{j:}\|_2$ as the rank-reducing regularizer on the latent state.[4] This regularizer offers an adaptive rank reducing scheme that zeros out many of the rows of the latent states and hence results in a low rank solution without knowing the rank *a priori*. For the reconstruction models $\mathbf{C}$ and $\mathbf{E}$, we need to specify a prior that respects the conditional independence of the views $\mathbf{x}_t$ and $\mathbf{x}_{t+1}$ given $\boldsymbol{\phi}_t$. This goal can be achieved if $\mathbf{C}$ and $\mathbf{E}$ are constrained individually so that they do not compete against each other to reconstruct their respective views [White et al., 2012]. Incorporating the regularizer and constraints, the resulting optimization problem has the form

$$\begin{aligned} &\arg\min_{C,E,\Phi} \sum_{t=1}^{T} \mathcal{L}_1(\mathbf{E}\boldsymbol{\phi}_{t-1}; \mathbf{x}_t) + \mathcal{L}_2(\mathbf{C}\boldsymbol{\phi}_t; \mathbf{x}_t) + \lambda \sum_{j=1}^{k} \|\mathbf{\Phi}_{j:}\|_2 \qquad (4.7) \\ &\text{s.t.} \|\mathbf{C}_{:j}\|_2 \leq \gamma_1, \|\mathbf{E}_{:j}\|_2 \leq \gamma_2 \ \forall j \in (1, k). \end{aligned}$$

Where $\mathcal{L}_1$ and $\mathcal{L}_2$ are convex losses, in general, corresponding to the first and the second view, respectively, that will become the Bregman divergences in the case of generalized linear observation.

The above constrained optimization problem is convex in each of the factor loading matrices $\{\mathbf{C}, \mathbf{E}\}$ and the state matrix $\mathbf{\Phi}$, but not jointly convex in terms of all these variables. Nevertheless, the following lemma show that

[4] Throughout this chapter, $\mathbf{X}_{i:}$ ($\mathbf{X}_{:i}$) is used to denote the ith row (ith column) of matrix $\mathbf{X}$ and also $[\mathbf{X}; \mathbf{Y}]$ ($[\mathbf{x}; \mathbf{y}]$) denotes the matrix (vector) concatenation operator which is equal to $[\mathbf{X}^\top, \mathbf{Y}^\top]^\top$ ($[\mathbf{x}^\top, \mathbf{y}^\top]^\top$).

equation 4.7 admits a convex reformulation by change of variable.

Lemma 4.1 *Let* $\hat{\mathbf{Z}}^{(1)} := \mathbf{C}\mathbf{\Phi}$ *and* $\hat{\mathbf{Z}}^{(2)} := \mathbf{E}\mathbf{\Phi}$ *with their concatenated matrix* $\hat{\mathbf{Z}} := \begin{bmatrix} \hat{\mathbf{Z}}^{(1)} \\ \hat{\mathbf{Z}}^{(2)} \end{bmatrix}$ *and* $\mathbf{Z}^{(1)} := [\mathbf{x}_{1:T-1}]$, $\mathbf{Z}^{(2)} := [\mathbf{x}_{2:T}]$. *In addition, let's define* $\mathbf{I}^{(1)} := diag(\begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix})$, $\mathbf{I}^{(2)} := diag(\begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix})$, *then the multi-view optimization problem (4.7) can be reformulated in the following convex form*

$$\min_{\substack{\|\mathbf{C}_{:j}\|_2 \le \gamma_1 \\ \|\mathbf{E}_{:j}\|_2 \le \gamma_2}} \min_{\mathbf{\Phi}: \begin{bmatrix} \mathbf{C} \\ \mathbf{E} \end{bmatrix} \mathbf{\Phi} = \hat{\mathbf{Z}}} L_1(\mathbf{C}\mathbf{\Phi}; \mathbf{Z}^{(1)}) + L_2(\mathbf{E}\mathbf{\Phi}; \mathbf{Z}^{(2)}) + \lambda \|\mathbf{\Phi}\|_{2,1}$$

$$= \min_{\hat{\mathbf{Z}}} \; L_1(\hat{\mathbf{Z}}^{(1)}; \mathbf{Z}^{(1)}) + L_2(\hat{\mathbf{Z}}^{(2)}; \mathbf{Z}^{(2)}) + \lambda \max_{0 \le \eta \le 1} \|\mathbf{U}_\eta^{-1} \hat{\mathbf{Z}}\|_{tr}$$

where $\mathbf{U}_\eta = \frac{\gamma_1}{\sqrt{\eta}} \mathbf{I}^{(1)} + \frac{\gamma_2}{\sqrt{1-\eta}} \mathbf{I}^{(2)}$ *and* $L_i(\mathbf{Y}; \hat{\mathbf{Y}}) = \sum_{t=1}^{T} \mathcal{L}_i(\mathbf{y}_t; \hat{\mathbf{y}}_t)$. *Moreover, we can show that the regularizer term* $\|\mathbf{U}_\eta^{-1} \hat{\mathbf{Z}}\|_{tr}$ *is concave in* η. *The trace norm induces a low rank result.*

Proof: The proof can be readily derived from the results of White et al. [2012]. ■

In the next section, we demonstrate how to obtain globally optimal estimates of $\mathbf{E}, \mathbf{C}$ and $\mathbf{\Phi}$.

Remark 1: This maximum likelihood formulation demonstrates how the distributional assumptions on the observations $\mathbf{x}_t$ can be generalized to any exponential family. Once expressed as the above optimization problem, one can further consider other losses and regularizers that may not immediately have a distributional interpretation, but result in improved prediction performance. This generalized formulation of maximum likelihood for LDS, therefore, has the additional benefit that it can flexibly incorporate optimization improvements, such as robust losses.[5] Also a regularizer can be designed to control overfitting to noisy observation, which is an issue in LDS that can result in an unstable

[5]Thus, we used $\mathcal{L}_1$ and $\mathcal{L}_2$ in equation 4.7 to generally refer to any loss function that is convex in its first argument.

latent dynamics estimate [Buesing et al., 2012a]. Therefore, by controlling undesired overfitting to noisy samples one can also prevent unintended unstable model identification.

Remark 2: We can generalize the optimization further to learn an *LDS with exogenous input*: a control vector $\mathbf{u}_t \in \mathbb{R}^d$ that impacts both the hidden state and observations. This entails adding some new variables to the general LDS model that can be expressed as

$$\begin{aligned} \boldsymbol{\phi}_{t+1} &= \mathbf{A}\boldsymbol{\phi}_t + \mathbf{B}\mathbf{u}_t + \eta_t \\ \mathbf{x}_t &= \mathbf{C}\boldsymbol{\phi}_t + \mathbf{D}\mathbf{u}_t + \epsilon_t \end{aligned}$$

with additional matrices $\mathbf{B} \in \mathbb{R}^{k\times d}$ and $\mathbf{D} \in \mathbb{R}^{d\times d}$. Again, by multiplying the state evolution equation by matrix $\mathbf{C}$ the resulting equations are

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{E}\boldsymbol{\phi}_t + \mathbf{F}\mathbf{u}_t + \mathbf{D}\mathbf{u}_{t+1} + \epsilon'_{t+1} \\ \mathbf{x}_t &= \mathbf{C}\boldsymbol{\phi}_t + \mathbf{D}\mathbf{u}_t + \epsilon_t \end{aligned}$$

where $\mathbf{F} := \mathbf{CB}$. Therefore, the loss can be generally expressed as

$$\mathcal{L}_1(\mathbf{E}\boldsymbol{\phi}_{t-1} + \mathbf{F}\mathbf{u}_{t-1} + \mathbf{D}\mathbf{u}_t; \mathbf{x}_t) + \mathcal{L}_2(\mathbf{C}\boldsymbol{\phi}_t + \mathbf{D}\mathbf{u}_t; \mathbf{x}_t).$$

The optimization would now be over the variables $\mathbf{C}, \mathbf{E}, \boldsymbol{\Phi}, \mathbf{D}, \mathbf{F}$, where the optimization could additionally include regularizers on $\mathbf{D}$ and $\mathbf{F}$ to control overfitting. Importantly, the addition of these variables $\mathbf{D}, \mathbf{F}$ does not modify the convexity properties of the loss, and the treatment for estimating $\mathbf{E}, \mathbf{C}$ and $\boldsymbol{\Phi}$ in section 4.4 directly applies. The optimization problem is jointly convex in $\mathbf{D}, \mathbf{F}$ and any one of $\mathbf{E}$, $\mathbf{C}$ or $\boldsymbol{\Phi}$ and jointly convex in $\mathbf{D}$ and $\mathbf{F}$. Therefore, an outer minimization over $\mathbf{D}$ and $\mathbf{F}$ can be added to Algorithm 1 and we will still obtain a globally optimal solution.

4.4 LDS Estimation Algorithm

To learn the optimal parameters for the reformulated two-view model, we adopt the generalized conditional gradient (GCG) algorithm developed by Yu et al. [2014]. GCG is designed for optimization problems of the form $l(x) + f(x)$ where $l(x)$ is convex and continuously differentiable with Lipschitz continuous gradient and $f(x)$ is a (possibly non-differentiable) convex function. The algorithm is computationally efficient, as well providing a reasonably fast $O(1/t)$ rate of convergence to the global minimizer. Though we have a nonconvex optimization problem, we can use the convex reformulation for two-view low-rank matrix factorization and resulting algorithm in [Yu et al., 2014, Section 4]. This algorithm includes a generic local improvement step, which significantly accelerates the convergence of the algorithm to a global optimum in practice. We provide a novel local improvement update, which both speeds learning and enforces a sparser structure on $\mathbf{\Phi}$, while maintaining the same theoretical convergence properties of GCG.

In our experiments, we specifically address the setting when the observations are assumed to be Gaussian, giving an Euclidean (ℓ_2) loss. We also prefer the unconstrained objective function that can be efficiently minimized by fast unconstrained optimization algorithms. Therefore, using the well-established equivalent form of the regularizer [Bach et al., 2008], the objective (4.7) can be equivalently cast for the Gaussian distributed time series $\mathbf{x}_t$ as

$$\min_{C,E,\Phi} \sum_{t=1}^{T} \|\mathbf{E}\boldsymbol{\phi}_{t-1} - \mathbf{x}_t\|_2^2 + \|\mathbf{C}\boldsymbol{\phi}_t - \mathbf{x}_t\|_2^2 + \lambda \sum_{j=1}^{k} \|\mathbf{\Phi}_{j:}\|_2 \ \max(\tfrac{1}{\gamma_1}\|\mathbf{C}_{:j}\|_2, \tfrac{1}{\gamma_2}\|\mathbf{E}_{:j}\|_2). \tag{4.8}$$

This product form of the regularizer is also preferred over the square form used in [Yu et al., 2014], since it induces row-wise sparsity on $\mathbf{\Phi}$. Though the square form $\|\mathbf{\Phi}\|_F^2$ admits efficient optimizers due to its smoothness, it does not prefer to zero out rows of $\mathbf{\Phi}$ while with the regularizer of the form (4.8), the learned hidden state will be appropriately projected down to a lower-dimensional space where many dimensions could be dropped from $\mathbf{\Phi}$, $\mathbf{C}$ and $\mathbf{E}$ giving a low rank

solution. In practice, we found that enforcing this sparsity property on $\mathbf{\Phi}$ significantly improved stability.[6] Consequently, we need optimization routines that are appropriate for the non smooth regularizer terms.

The local improvement step involves alternating block coordinate descent between $\mathbf{C}, \mathbf{E}$ and $\mathbf{\Phi}$, with an accelerated proximal gradient algorithm (FISTA) [Beck and Teboulle, 2009] for each descent step. To use the FISTA algorithm we need to provide a proximal operator for the non-smooth regularizer in equation 4.8.

Let the proximal operator of a convex and possibly non-differentiable function $\lambda f(\mathbf{y})$ be defined as

$$\text{prox}_{\lambda f}(\mathbf{x}) = \arg\min_{\mathbf{y}} \lambda f(\mathbf{y}) + \tfrac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2.$$

FISTA is an accelerated version of ISTA (Iterative Shrinkage-Thresholding Algorithm) that iteratively performs a gradient descent update with the smooth component of the objective, and then applies the proximal operator as a projection step. Each iteration updates the variable $\mathbf{x}$ as $\mathbf{x}^{k+1} = \text{prox}_{\lambda_k f}\left(\mathbf{x}^k - \lambda_k \nabla l(\mathbf{x}^k)\right)$, which converges to a fixed point. If there is no known form for the proximal operator, as is the case for our non-differentiable regularizer, a common strategy is to numerically calculate the proximal update. This approach, however, can be prohibitively expensive, and an analytic (closed) form is clearly preferable. We derive such a closed form for equation 4.8 in Theorem 4.1.

Theorem 4.1 *For a vector* $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}$ *composed of two subvectors* $\mathbf{v}_1, \mathbf{v}_2$*, define* $f(\mathbf{v}) = \lambda\|\mathbf{v}\|_{2v} := \lambda \max(\|\mathbf{v}_1\|_2, \|\mathbf{v}_2\|_2)$*. The proximal operator for this function*

[6] This was likely due to a reduction in the size of the transition parameters, resulting in improved re-estimation of $\mathbf{A}$ and a corresponding reduction in error accumulation when using the model for forecasting.

is

$$prox_f(\mathbf{v}) = \begin{cases} \begin{bmatrix} \mathbf{v}_1 \max\{1 - \frac{\alpha}{\|\mathbf{v}_1\|}, 0\} \\ \mathbf{v}_2 \max\{1 - \frac{\lambda-\alpha}{\|\mathbf{v}_2\|}, 0\} \end{bmatrix} & \text{if } \|\mathbf{v}_1\| \leq \|\mathbf{v}_2\| \\ \begin{bmatrix} \mathbf{v}_1 \max\{1 - \frac{\lambda-\beta}{\|\mathbf{v}_1\|}, 0\} \\ \mathbf{v}_2 \max\{1 - \frac{\beta}{\|\mathbf{v}_2\|}, 0\} \end{bmatrix} & \text{if } \|\mathbf{v}_2\| \leq \|\mathbf{v}_1\| \end{cases}$$

where $\alpha := \max\{.5(\|\mathbf{v}_1\| - \|\mathbf{v}_2\| + \lambda), 0\}$ *and* $\beta := \max\{.5(\|\mathbf{v}_2\| - \|\mathbf{v}_1\| + \lambda), 0\}$.

Proof: See Appendix D.1. ■

This result can be further generalized to enable additional regularization components on $\mathbf{C}$ and $\mathbf{E}$, such as including an ℓ_1 norm on each column to further enforce sparsity (such as in the elastic net). There is no closed form for the proximal operator of the sum of two functions in general. We prove, however, that for special case of a linear combination of the two-view norm with any norms on the columns of $\mathbf{C}$ and $\mathbf{E}$, the proximal mapping reduces to a simple composition rule.

Theorem 4.2 *For norms* $R_1(\mathbf{v}_1)$ *and* $R_2(\mathbf{v}_2)$*, the proximal operator of the linear combination* $R_c(\mathbf{v}) = \lambda\|\mathbf{v}\|_{2v} + \nu_1 R_1(\mathbf{v}_1) + \nu_2 R_2(\mathbf{v}_2)$ *for* $\nu_1, \nu_2 \geq 0$ *admits the simple composition* $prox_{R_c}(\mathbf{v}) = prox_{\lambda\|.\|_{2v}}\left(\begin{bmatrix} prox_{\nu_1 R_1}(\mathbf{v}_1) \\ prox_{\nu_2 R_2}(\mathbf{v}_2) \end{bmatrix}\right)$.

Proof: See Appendix D.1. ■

4.4.1 Recovery of the LDS model parameters

The above reformulation provides a tractable learning approach to obtain the optimal parameters for the two-view reformulation of LDS; given this optimal solution, we can then estimate the parameters to the original LDS. The first step is to estimate the transition matrix $\mathbf{A}$. A natural approach is to use equation 4.2, and set $\hat{\mathbf{A}} = \hat{\mathbf{C}}^\dagger \hat{\mathbf{E}}$ for pseudoinverse $\hat{\mathbf{C}}^\dagger$. This $\hat{\mathbf{A}}$, however, might be sensitive to inaccurate estimation of the (effective) hidden state dimension

Algorithm 1 LDS-DV

Input: training sequence $\{x_t, t \in [1, T]\}$
Output: $\mathbf{C}, \mathbf{A}, \boldsymbol{\phi}_t, \boldsymbol{\Sigma}_\eta, \boldsymbol{\Sigma}_\epsilon$
Initialize $\mathbf{C}_0, \mathbf{E}_0, \boldsymbol{\Phi}_0$
$\mathbf{U}_1 \leftarrow [\mathbf{C}_0^\top;\ \mathbf{E}_0^\top]^\top, \quad \mathbf{V}_1 \leftarrow \boldsymbol{\Phi}_0^\top$
for $i = 1, \ldots$ **do**
  $(\mathbf{u}_i, \mathbf{v}_i) \leftarrow \arg\min_{\mathbf{u}\mathbf{v}^\top \in \mathcal{A}} \left\langle \nabla \ell(\mathbf{U}_i, \mathbf{V}_i), \mathbf{u}\mathbf{v}^\top \right\rangle$ // compute polar
  $(\eta_i, \theta_i) \leftarrow \arg \min_{0 \le \eta \le 1, \theta \ge 0} \ell((1-\eta)\mathbf{U}_i\mathbf{V}_i^\top + \theta \mathbf{u}_i\mathbf{v}_i^\top) + \lambda((1-\eta)\rho_i + \theta)$ // partially corrective update (PCU)
  $\mathbf{U}_{init} \leftarrow [\sqrt{1-\eta_i}\mathbf{U}_i, \sqrt{\theta_i}\mathbf{u}_i]$, $\mathbf{V}_{init} \leftarrow [\sqrt{1-\eta_i}\mathbf{V}_i, \sqrt{\theta_i}\mathbf{v}_i]$
  $(\mathbf{U}_{i+1}, \mathbf{V}_{i+1}) \leftarrow \text{FISTA}(\mathbf{U}_{init}\mathbf{V}_{init})$
  $\rho_i = \frac{1}{2}\sum_{j=1}^{i+1}(\|(\mathbf{U}_{i+1})_{:i}\|_{2v}^2 + \|(\mathbf{V}_{i+1})_{:i}\|_2^2)$
end for
$(\mathbf{C};\ \mathbf{E}) \leftarrow \mathbf{U}_{i+1}, \quad \boldsymbol{\Phi} \leftarrow \mathbf{V}_{i+1}^\top$
$\mathbf{A} \leftarrow \Phi_{2:T} * \Phi_{1:T-1}^\dagger$
estimate $\boldsymbol{\Sigma}_\eta,\ \boldsymbol{\Sigma}_\epsilon$ by sample covariances

k. We found in practice that modifications from the optimal choice of k might result in unstable solutions and produce unreliable forecasts. Instead, a more stable $\hat{\mathbf{A}}$ can be learned from the hidden states themselves. This approach also focuses estimation of $\mathbf{A}$ on the forecasting task, which is our ultimate aim.

Given the sequence of hidden states, $\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_T$, there are several strategies that could be used to estimate $\mathbf{A}$, including simple autoregressive models to more sophisticated strategies [Siddiqi et al., 2007]. We opt for a simple linear regression solution $\hat{\mathbf{A}} = \arg\min_{\mathbf{A}} \sum_{t=1}^{T-1} \|\boldsymbol{\phi}_{t+1} - \mathbf{A}\boldsymbol{\phi}_t\|_2^2$ which we found produced stable $\hat{\mathbf{A}}$.

To estimate the noise parameters $\boldsymbol{\Sigma}_\eta, \boldsymbol{\Sigma}_\epsilon$, recall $\eta_t = \boldsymbol{\phi}_{t+1} - \hat{\mathbf{A}}\boldsymbol{\phi}_t,\ \epsilon_t = \mathbf{x}_t - \mathbf{C}\boldsymbol{\phi}_t$. Having obtained $\hat{\mathbf{A}}$, therefore, we can estimate the noise covariance matrices by computing their sample covariances as $\hat{\boldsymbol{\Sigma}}_\eta = \frac{1}{T-1}\sum_{t=1}^{T} \eta_t\eta_t^\top$, $\hat{\boldsymbol{\Sigma}}_\epsilon = \frac{1}{T-1}\sum_{t=1}^{T} \epsilon_t\epsilon_t^\top$. The final LDS learning procedure is outlined in Algorithm 1. For more details about polar computation and partially corrective subroutine see [Yu et al., 2014, Section 4].

4.5 Experimental results

We evaluate the proposed algorithm by comparing one step prediction performance and computation speed with alternative methods for real and synthetic

time series. We report the normalized mean square error (NMSE) defined as $\text{NMSE} = \frac{\sum_{t=1}^{T_{test}} \|y_t - \hat{y}_t\|^2}{\sum_{t=1}^{T_{test}} \|y_t - \mu_y\|^2}$ where $\mu_y = \frac{1}{T_{test}} \sum_{t=1}^{T_{test}} y_t$.

Algorithms: We compared the proposed algorithm to a well-established method-of moment-based algorithm, N4SID [Van Overschee and De Moor, 1994], Hilbert space embeddings of hidden Markov models (HSE-HMM) [Song et al., 2010], expectation-maximization for estimating the parameters of a Kalman filter (EM) [Roweis and Ghahramani, 1999] and PEM [Ljung, 1999]. These are standard baseline algorithms that are used regularly for LDS identification. The estimated parameters by N4SID were used as the initialization point for EM and PEM algorithms in our experiments. We used the built-in functions, `n4sid` and `pem`, in Matlab, with the order selected by the function, for the subspace identification method and PEM, respectively. For our algorithm, we select the regularization parameter λ using cross-validation. For the time series, the training data is split by performing the learning on first 80% of the training data and evaluating the prediction performance on the remaining 20%.

Real datasets: For experiments on real datasets we select the climate time series from IRI data library that recorded the surface temperature on the monthly basis for tropical Atlantic ocean (ATL) and tropical Pacific ocean (CAC). In CAC we selected first 30×30 grids out of the total 84×30 locations with 399 monthly samples, while in ATL the first 9×9 grids out of the total 38×25 locations are selected each with timeseries of length 564. We partitioned each area to smaller areas of size 3×3 and arrange them to vectors of size 9, then seasonality component of the time series are removed and data is centered to have zero mean. We ran two experiments for each dataset. For the first, the whole sequence is sliced into 70% training and 30% test. For the second, a short training set of 70 samples is selected, with a test sequence of size 50.

Synthetic datasets: In the synthetic experiments, the datasets are generated by an LDS model (4.1) of different system orders, k, and observation sizes, d. For each test case, 100 data sequences of length 200 samples are generated and sliced to 70%, 30% ratios for training set and test set, respectively. The dynamics matrix $\mathbf{A}$ is selected to produce a stable system: $\{|\sigma_i(\mathbf{A})| = s : s \leq 1, \ \forall i \in (1, k)\}$ where $\sigma_i(\mathbf{A})$ is the ith eigen value of

Table 1: Real time series

	ATL(Long)		**ATL(Short)**		**CAC(Long)**		**CAC(Short)**	
	NMSE	Time	NMSE	Time	NMSE	Time	NMSE	Time
LDS-MV	**0.45**$\pm$**0.03**	0.26	**0.54**$\pm$**0.05**	0.22	**0.58**$\pm$**0.02**	0.28	**0.63**$\pm$**0.03**	0.14
N4SID	**0.52**$\pm$**0.04**	2.34	**0.59**$\pm$**0.05**	0.95	**0.61**$\pm$**0.02**	1.23	0.84$\pm$0.07	1.08
EM	0.64$\pm$0.04	7.87	0.88$\pm$0.07	3.92	0.81$\pm$0.02	5.70	1.02$\pm$0.08	4.12
HSE-HMM	675.87$\pm$629.46	0.79	0.97$\pm$0.01	0.16	11.24$\pm$8.23	0.39	2.82$\pm$1.60	0.17
PEM-SSID	0.71$\pm$0.08	20.00	1.52$\pm$0.66	16.38	1.38$\pm$0.15	19.67	2.68$\pm$0.78	20.58

Table 2: Synthetic time series

	(S1) d=5 , k=3		**(S2) d=5 , k=3**		**(S1) d=8 , k=6**		**(S2) d=8 , k=6**		**(S1) d=16 , k=9**		**(S2) d=16 , k=9**	
	NMSE	Time	NMSE	Time	NMSE	Time	NMSE	Time	NMSE	Time	NMSE	Time
LDS-MV	**0.12**$\pm$**0.01**	0.49	**0.17**$\pm$**0.02**	0.36	**0.08**$\pm$**0.00**	0.66	**0.04**$\pm$**0.00**	0.52	**0.07**$\pm$**0.00**	1.01	**0.03**$\pm$**0.00**	1.72
N4SID	**0.12**$\pm$**0.01**	0.81	0.42$\pm$0.04	0.76	0.11$\pm$0.00	1.45	0.39$\pm$0.04	1.38	0.10$\pm$0.00	4.29	0.42$\pm$0.04	4.40
EM	0.18$\pm$0.01	4.99	**0.15**$\pm$**0.02**	4.62	0.14$\pm$0.01	6.01	**0.04**$\pm$**0.00**	5.03	0.13$\pm$0.00	19.21	0.03$\pm$0.00	19.83
HSE-HMM	2.4e+4$\pm$1.7e+4	0.48	2.2e+7$\pm$2.2e+7	0.50	7.8e+03$\pm$7.7e+03	0.49	0.65$\pm$0.02	0.55	22.92$\pm$21.83	0.53	0.71$\pm$0.01	0.61
PEM-SSID	0.14$\pm$0.01	10.72	0.25$\pm$0.03	9.08	0.12$\pm$0.01	15.22	0.08$\pm$0.01	13.97	0.09$\pm$0.01	38.39	0.06$\pm$0.02	41.10

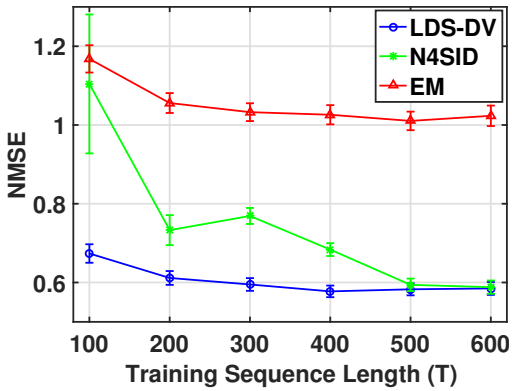
Table 4.1: Results for real and synthetic datasets are listed in Table 1 and Table 2, respectively. The first column of each dataset is the average normalized MSE with standard error and the second column is the algorithm runtime in CPU seconds. The best NMSE according to pairwise t-test with significance level of 5% is highlighted.

matrix $\mathbf{A}$. The noise components are drawn from Gaussian distributions and scaled so that $p_\eta := E\{\eta^\top \eta\}/m$ and $p_\epsilon := E\{\epsilon^\top \epsilon\}/n$. Each test is repeated with the following settings: $\{$**S1:** $s = 0.970, p_\eta = 0.50$ and $p_\epsilon = 0.1\}$, $\{$**S2:** $s = 0.999, p_\eta = 0.01$ and $p_\epsilon = 0.1\}$.

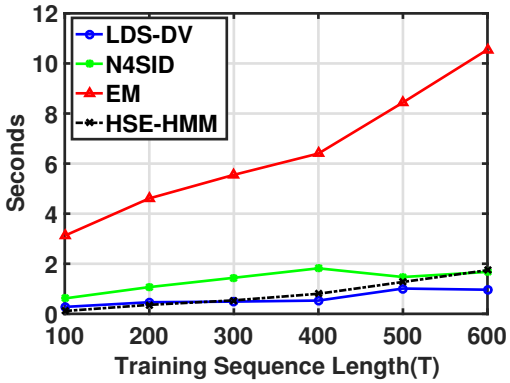
Results: The NMSE and run-time results obtained on real and synthetic datasets are shown in Table 1 and Table 2, respectively. In terms of NMSE, LDS-DV outperforms and matches the alternative methods. In terms of algorithm speed, the LDS-DV learns the model much faster than the competitors and scales well to larger dimension models. The speed improvement is more significant for larger datasets and observations with higher dimensions.

For test cases with $|\sigma_i(\mathbf{A})| \simeq 1$, designed to evaluate the prediction performance of the methods for marginally stable systems, LDS-DV still can learn a stable model while the other algorithms might not learn a stable model. The proposed LDS-DV method does not explicitly impose stability, but the regularization favors $\mathbf{A}$ that is stable. The regularizer on latent state encourages smooth dynamics and controls overfitting: overfitting to noisy observations can lead to unstable estimate of the model [Buesing et al., 2012a], and a smooth latent trajectory is a favorable property in most real-world applications.

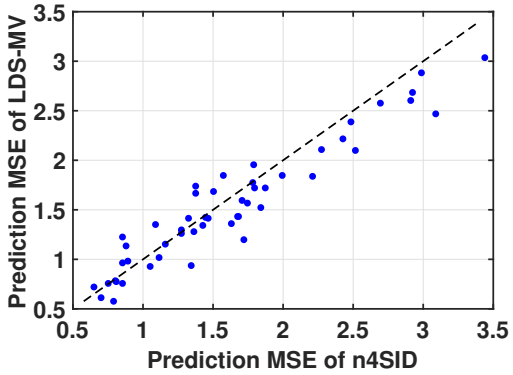
Figure 4.3(c) shows the MSE of LDS-DV versus N4SID, for all the CAC



(a) NMSE



(b) Runtime



(c) Scatter plot of MSE

Figure 4.3: a) NMSE of the LDS-DV for increasing length of training sequence. The difference between LDS-DV and N4SID is more significant in shorter training length, while both converge to the same accuracy in large T. HSE-HMM is omitted due to its high error. b) Runtime in CPU seconds for increasing length of training sequence. LDS-DV scales well with large sample length. c) MSE of the LDS-DV versus MSE of N4SID. In higher values of MSE, the points are below identity function line and LDS-DV is more likely to win.

time-series. This figure illustrates that for easier problems, LDS-DV and N4SID are more comparable. However, as the difficulty increase, and MSE increases, LDS-DV begins to consistently outperform N4SID.

Figures 4.3(a) and 4.3(b) illustrate the accuracy and runtime respectively of the algorithms versus training length. We used the synthetic LDS model under condition S1 with $n = 8,\ m = 6$. Values are averaged over 20 runs with a test length of 50 samples. LDS-DV has better early performance, for smaller sample sizes. At larger sample sizes, they reach approximately the same error level.

Chapter 5

Summary and Discussion

In this thesis, some new ideas in probabilistic generative modeling have been developed that play important roles in expanding these models to tackle the growing demand of complex and fast machine learning algorithms.

The main focus of chapters 2 and 3 was on recent advances in deep generative models.

Chapter 2 showed that circular and symmetric convolutions can be used as invertible transformations with fast and efficient inversion, deconvolution, and Jacobian determinant evaluation. These features make the approach well suited for designing flexible normalizing flows. Using these invertible convolutions, we introduced a family of data adaptive coupling layers, which consist of convolutions, where the kernel of the convolutions are themselves a function of the coupling layer input. We also analytically derived invertible pointwise nonlinearities that implicitly induce specific regularizers on intermediate activations in deep flow models. The results also help better understand the role of nonlinear gates through the lens of their contribution to latent variables' distributions. Using these new architectural components, we achieved state of the art performance on several datasets for invertible normalizing flows with fast sampling.

In chapter 3, deep probabilistic generative modeling for multi-view data was studied. We developed a simple yet powerful tool for multi-view learning based on the probabilistic interpretation of CCA. It has been shown that,

following the theoretical formulation of the linear probabilistic CCA model in conjunction with variational inference principles for deep generative networks, we can obtain a scalable end-to-end learning algorithm for two-view data. Moreover, the deep probabilistic model is generalized to problems with an arbitrary number of views. Experimental results have shown that the proposed model is able to efficiently integrate the relationship between multiple views to obtain a more powerful representation, achieving state-of-the-art performance on several downstream tasks. An important application of the proposed deep generative model is in multi-modal clustering, where the proposed model could efficiently leverage the extra modalities to uncover the cluster memberships, the common underlying information among modalities. These, indeed, confirm that the proposed method is an efficient way to extend variational inference to deep probabilistic multi-view learning.

Lastly, we provided an algorithm for optimal estimation of the parameters for a time-invariant, discrete-time linear dynamical system in chapter 4. More precisely, we provided a reformulation of the model as a two-view objective, which allowed recent advances for optimal estimation for two-view models to be applied. The resulting algorithm is simple to use and flexibly allows different losses and regularizers to be incorporated. Despite this simplicity, significant improvements were observed over a widely accepted method for subspace identification (N4SID), both in terms of accuracy for forecasting and runtime.

The goal of this chapter was optimal estimation of the hidden states and transition matrices as they are essential for forecasting purpose; however, in some settings, estimation of noise parameters for LDS models is also desired. An unresolved issue is joint optimal estimation of these noise parameters. Though we do explicitly estimate the noise parameters, we do so only from the residuals after obtaining the optimal hidden states and transition and observation matrices. Moreover, consistency of the learned parameters by the proposed procedure of this chapter is still an open problem and will be an interesting future work.

The proposed optimization approach for LDSs should be useful for applica-

tions where alternative noise assumptions are desired. A Laplace assumption on the observations, for example, provides a more robust ℓ_1 loss. A Poisson distribution has been advocated for count data, such as for neural activity, where the time series is a vector of small integers [Buesing et al., 2012b]. The proposed formulation of estimation for LDSs easily enables extension to such distributions. An important next step is to investigate the applicability to a wider range of time series data.

5.1 Future Work

The methods presented in this work can be expanded in several ways. In the following some interesting future directions are presented.

Combining LDS and invertible flows: Normalizing flows offer an exact and efficient inference for latent variables using the invertible nature of the generative network, therefore resulting in tractable exact maximum likelihood estimation. This, in fact, makes NFs promising candidates to generalize multivariate time series models such as linear dynamical systems. Substituting the observation model of LDS with a deep NF, one can achieve an invertible generative model associating a Gauss-Markov model as the latent space dynamics of the model that sounds a promising avenue for future research in time series analysis. In [Kumar et al., 2019], a flow-based sequence modeling was proposed for multi-frame video prediction, called VideoFlow, which applied autoregressive probabilistic priors on the hierarchical latent space of the NF. VideoFlow produced results comparable to the state-of-the-art generative models in video prediction.

LDA as CCA: Fisher linear discriminant analysis can be viewed as CCA when the first view is the vector of features and the second view is the indicter vector (one-hot encoding) of labels. A possible extension of variational probabilistic CCA is to modify it to a deep variational LDA for classification. Here, we suggest approximating the posterior of latent variables corresponding to

the labels as $q_\eta(\mathbf{z}_2|\mathbf{x}_2) = \mathcal{N}(\mathbf{z}_2; \boldsymbol{\mu}_2 = \mathbf{x}_2, \boldsymbol{\Sigma}_{22} = f(\mathbf{x}_1; \theta))$, where the difference with the deep probabilistic CCA is that both variances vectors are functions of the features (1st view), hence a conditional model to estimate the labels given the features can be approximated.

More flexible flow-based VCCA: Although simple Gaussian approximate posteriors are adopted for the latent linear probabilistic model of the deep generative multi-view models, arbitrarily complex approximate posteriors can be obtained by applying rich normalizing flows on these simple base distribution, as introduced in chapter 2. By reducing the gap between the true posterior and its approximation, this technique is expected to provide a more expressive generative models for complex multi-view applications hence serving as a potential candidate for future studies.

Using the mixture of experts (MoE) or product of experts (PoE) to design more flexible variational multi-modal architectures: In chapter 3, a simple variational approach is proposed to model the approximate posteriors of a set of cross-correlated multivariate Gaussian latent variables $\{\mathbf{z}_m\}_{m=1}^M$ whereas their cross correlation are specified by a linear probabilistic CCA layer. On the other hand, the ideas of product of experts (PoE) and mixture of experts (MoE) [Wu and Goodman, 2018, Shi et al., 2019] — which model the variational posterior of the latent variable of all modalities as a composition of a set of unimodal posteriors — can be applied in combination to the variational probabilistic CCA model to achieve more flexible posteriors in multi-modal settings. To this end, simple unimodal posteriors $\{q(\boldsymbol{\phi}, \boldsymbol{\epsilon}_1 ... \boldsymbol{\epsilon}_M|\mathbf{x}_m)\}_{m=1}^M$, as defined in chapter 3, which model the cross-correlated latent factors by the linear probabilistic CCA based on one of the modalities, can be combined in MoE or PoE form to construct a multi-modal posterior $q(\boldsymbol{\phi}, \boldsymbol{\epsilon}_1 ... \boldsymbol{\epsilon}_M|\mathbf{x}_1, ..., \mathbf{x}_M)$. This idea proposes a future direction for multi-modal learning.

Deep variational CCA as an extension of LDS: CCA is fundamental tool in LDS identification [Katayama, 2006]. Also, in [Karami et al., 2017] a

two-view reformulation of LDS model for time series analysis was established. Therefore, another possible extension of our work is to modify the probabilistic generative multi-view model and derive a deep LDS model.

References

Mahdi Karami, Dale Schuurmans, Jascha Sohl-Dickstein, Laurent Dinh, and Daniel Duckworth. Invertible convolutional flow. In *Advances in Neural Information Processing Systems*, pages 5636–5646, 2019a.

Mahdi Karami, Dale Schuurmans, Jascha Sohl-Dickstein, Laurent Dinh, and Daniel Duckworth. Symmetric convolutional flow. In *Workshop on Invertible Neural Nets and Normalizing Flows(INNF), ICML 2019*, 2019b.

Mahdi Karami, Laurent Dinh, Daniel Duckworth, Jascha Sohl-Dickstein, and Dale Schuurmans. Generative convolutional flow for density estimation. In *Workshop on Bayesian Deep Learning NeurIPS 2018*, 2018.

Mahdi Karami and Dale Schuurmans. Variational inference for deep probabilistic canonical correlation analysis. *arXiv preprint arXiv:2003.04292*, 2020a.

Mahdi Karami, Martha White, Dale Schuurmans, and Csaba Szepesvári. Multi-view matrix factorization for linear dynamical system estimation. In *Advances in Neural Information Processing Systems*, pages 7092–7101, 2017.

Mahdi Karami. Deep generative multi-view learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 465–477. Springer, 2019.

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

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 689–696, 2011.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016a.

Johannes Ballé, Valero Laparra, and Eero P Simoncelli. Density modeling of images using a generalized normalization transformation. *arXiv preprint arXiv:1511.06281*, 2015.

Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016b.

Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5485–5493, 2017.

Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

Volodymyr Kuleshov, S Zayd Enam, and Stefano Ermon. Audio super resolution using neural networks. *arXiv preprint arXiv:1708.00853*, 2017.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Karol Gregor, Frederic Besse, Danilo Jimenez Rezende, Ivo Danihelka, and Daan Wierstra. Towards conceptual compression. In *Advances In Neural Information Processing Systems*, pages 3549–3557, 2016.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018.

Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017.

Michael Tschannen, Eirikur Agustsson, and Mario Lucic. Deep generative models for distribution-preserving lossy compression. In *Advances in Neural Information Processing Systems*, pages 5929–5940, 2018.

James Townsend, Tom Bird, and David Barber. Practical lossless compression with latent variables using bits back coding. *arXiv preprint arXiv:1901.04866*, 2019.

Yonghong Luo, Xiangrui Cai, Ying Zhang, Jun Xu, et al. Multivariate time series imputation with generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 1596–1607, 2018.

Manoj Kumar, Mohammad Babaeizadeh, Dumitru Erhan, Chelsea Finn, Sergey Levine, Laurent Dinh, and Durk Kingma. Videoflow: A flow-based generative model for video. *arXiv preprint arXiv:1903.01434*, 2019.

Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H Campbell, and Sergey Levine. Stochastic variational video prediction. *arXiv preprint arXiv:1710.11252*, 2017.

Rob Fergus, David W Hogg, Rebecca Oppenheimer, Douglas Brenner, and Laurent Pueyo. S4: A spatial-spectral model for speckle suppression. *The Astrophysical Journal*, 794(2):161, 2014.

Jeffrey Regier, Andrew Miller, Jon McAuliffe, Ryan Adams, Matt Hoffman, Dustin Lang, David Schlegel, and Mr Prabhat. Celeste: Variational inference for a generative model of astronomical images. In *International Conference on Machine Learning*, pages 2095–2103, 2015.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Alexander B Wiltschko, Matthew J Johnson, Giuliano Iurilli, Ralph E Peterson, Jesse M Katon, Stan L Pashkovski, Victoria E Abraira, Ryan P Adams, and Sandeep Robert Datta. Mapping sub-second structure in mouse behavior. *Neuron*, 88(6):1121–1135, 2015.

Scott W Linderman, Matthew J Johnson, Matthew A Wilson, and Zhe Chen. A bayesian nonparametric approach for uncovering rat hippocampal population codes during spatial navigation. *Journal of neuroscience methods*, 263:36–47, 2016.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, pages 881–889, 2015.

Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems*, pages 4790–4798, 2016c.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Maciej Wiatrak and Stefano V Albrecht. Stabilizing generative adversarial network training: A survey. *arXiv preprint arXiv:1910.00927*, 2019.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.

George Papamakarios, Iain Murray, and Theo Pavlakou. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, pages 2338–2347, 2017.

Mahdi Karami and Dale Schuurmans. Variational inference for deep probabilistic canonical correlation analysis. *arXiv preprint arXiv:2003.04292*, 2020b.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 1530–1538, 2015.

Rianne van den Berg, Leonard Hasenclever, Jakub M Tomczak, and Max Welling. Sylvester normalizing flows for variational inference. *arXiv preprint arXiv:1803.05649*, 2018.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow.

Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, and David Duvenaud. FFJORD: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations*, 2019.

Guoqing Zheng, Yiming Yang, and Jaime Carbonell. Convolutional normalizing flows. *arXiv preprint arXiv:1711.02255*, 2017.

Emiel Hoogeboom, Rianne van den Berg, and Max Welling. Emerging convolutions for generative normalizing flows. *arXiv preprint arXiv:1901.11137*, 2019.

John F Monahan. *Numerical methods of statistics*. Cambridge University Press, 2011.

Z. Cinkir. A fast elementary algorithm for computing the determinant of Toeplitz matrices. *ArXiv e-prints*, January 2011.

Per-Gunnar Martinsson, Vladimir Rokhlin, and Mark Tygert. A fast algorithm for the inversion of general toeplitz matrices. *Computers & Mathematics with Applications*, 50(5-6):741–752, 2005.

Robert M Gray et al. Toeplitz and circulant matrices: A review. *Foundations and Trends® in Communications and Information Theory*, 2(3):155–239, 2006.

Rafael C Gonzalez and Richard E Woods. Digital image processing, 1992.

Stephen A Martucci. Symmetric convolution and the discrete sine and cosine transforms. *IEEE Transactions on Signal Processing*, 42(5):1038–1051, 1994.

Yu Cheng, Felix X Yu, Rogerio S Feris, Sanjiv Kumar, Alok Choudhary, and Shi-Fu Chang. An exploration of parameter redundancy in deep networks with circulant projections. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2857–2865, 2015.

Marcin Moczulski, Misha Denil, Jeremy Appleyard, and Nando de Freitas. Acdc: A structured efficient linear layer, 2015.

Thomas M Foltz and BM Welsh. Image reconstruction using symmetric convolution and discrete trigonometric transforms. *JOSA A*, 15(11):2827–2840, 1998.

Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in neural information processing systems*, pages 901–909, 2016.

Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. Neural autoregressive flows. In *International Conference on Machine Learning*, pages 2083–2092, 2018.

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design, 2019.

C. Cortes Y. LeCun. The mnist database of handwritten digit. 1998.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

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

Harold Hotelling. Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. Springer, 1992.

Rongkai Xia, Yan Pan, Lei Du, and Jian Yin. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

Arto Klami and Samuel Kaski. Local dependent components. In *Proceedings of the 24th international conference on Machine learning*, pages 425–432. ACM, 2007.

Arto Klami, Seppo Virtanen, and Samuel Kaski. Bayesian canonical correlation analysis. *Journal of Machine Learning Research*, 14(Apr):965–1003, 2013.

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *International Conference on Machine Learning*, pages 1247–1255, 2013.

Weiran Wang, Karen Livescu, and Jeff Bilmes. On Deep Multi-View Representation Learning. *Icml*, 37, 2015a.

Qingming Tang, Weiran Wang, and Karen Livescu. Acoustic feature learning via deep variational canonical correlation analysis. *arXiv preprint arXiv:1708.04673*, 2017.

Weiran Wang, Honglak Lee, and Karen Livescu. Deep variational canonical correlation analysis. *arXiv preprint arXiv:1610.03454*, 2016.

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

Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep subspace clustering networks. In *Advances in Neural Information Processing Systems*, pages 24–33, 2017.

René Vidal. Subspace clustering. *IEEE Signal Processing Magazine*, 28(2): 52–68, 2011.

João Paulo Costeira and Takeo Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, 1998.

Pan Ji, Mathieu Salzmann, and Hongdong Li. Shape interaction matrix revisited and robustified: Efficient subspace clustering with corrupted and incomplete data. In *Proceedings of the IEEE International Conference on computer Vision*, pages 4687–4695, 2015.

Quanyi Mo and Bruce A Draper. Semi-nonnegative matrix factorization for motion segmentation with missing data. In *European Conference on Computer Vision*, pages 402–415. Springer, 2012.

René Vidal, Roberto Tron, and Richard Hartley. Multiframe motion segmentation with missing data using powerfactorization and gpca. *International Journal of Computer Vision*, 79(1):85–105, 2008.

Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2797. IEEE, 2009.

Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2765–2781, 2013.

Jiashi Feng, Zhouchen Lin, Huan Xu, and Shuicheng Yan. Robust subspace segmentation with block-diagonal prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3818–3825, 2014.

Chun-Guang Li and Rene Vidal. Structured sparse subspace clustering: A unified optimization framework. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 277–286, 2015.

Guangcan Liu, Zhouchen Lin, and Yong Yu. Robust subspace segmentation by low-rank representation. In *ICML*, volume 1, page 8, 2010.

Yu-Xiang Wang, Huan Xu, and Chenlei Leng. Provable subspace clustering: When lrr meets ssc. In *Advances in Neural Information Processing Systems*, pages 64–72, 2013.

Chong You, Chun-Guang Li, Daniel P Robinson, and René Vidal. Oracle based active set algorithm for scalable elastic net subspace clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3928–3937, 2016.

Mahdi Abavisani and Vishal M Patel. Deep multimodal subspace clustering networks. *IEEE Journal of Selected Topics in Signal Processing*, 12(6): 1601–1614, 2018a.

Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

Michael W Browne. The maximum-likelihood solution in inter-battery factor analysis. *British Journal of Mathematical and Statistical Psychology*, 32(1): 75–86, 1979.

Cédric Archambeau and Francis R Bach. Sparse probabilistic projections. In *Advances in neural information processing systems*, pages 73–80, 2009.

Arto Klami, Seppo Virtanen, Eemeli Leppäaho, and Samuel Kaski. Group factor analysis. *IEEE transactions on neural networks and learning systems*, 26(9):2136–2147, 2014.

Michael W Browne. Factor analysis of multiple batteries by maximum likelihood. *British Journal of Mathematical and Statistical Psychology*, 33(2):184–199, 1980.

Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

David R. Hardoon, Sandor R. Szedmak, and John R. Shawe-taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Comput.*, 16(12):2639–2664, December 2004. ISSN 0899-7667.

Francis R. Bach and Michael I. Jordan. Kernel independent component analysis. *J. Mach. Learn. Res.*, 3:1–48, March 2003. ISSN 1532-4435.

Christopher KI Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.

David Lopez-Paz, Suvrit Sra, Alex Smola, Zoubin Ghahramani, and Bernhard Schölkopf. Randomized nonlinear component analysis. In *International Conference on Machine Learning*, pages 1359–1367, 2014.

Aaron Shon, Keith Grochow, Aaron Hertzmann, and Rajesh PN Rao. Learning shared latent structure for image synthesis and robotic imitation. In *Advances in neural information processing systems*, pages 1233–1240, 2006.

Andreas Damianou, Carl Ek, Michalis Titsias, and Neil Lawrence. Manifold relevance determination. *arXiv preprint arXiv:1206.4610*, 2012.

Mike Wu and Noah Goodman. Multimodal generative models for scalable weakly-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5575–5585, 2018.

Yuge Shi, N Siddharth, Brooks Paige, and Philip Torr. Variational mixture-of-experts autoencoders for multi-modal deep generative models. In *Advances in Neural Information Processing Systems*, pages 15692–15703, 2019.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Deng Cai, Xiaofei He, and Jiawei Han. Document clustering using locality preserving indexing. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1624–1637, 2005.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Mahdi Abavisani and Vishal M Patel. Multimodal sparse and low-rank subspace clustering. *Information Fusion*, 39:168–177, 2018b.

Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

Kuang-Chih Lee, Jeffrey Ho, and David J Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (5):684–698, 2005.

William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

Changqing Zhang, Huazhu Fu, Si Liu, Guangcan Liu, and Xiaochun Cao. Low-rank tensor constrained multiview subspace clustering. In *Proceedings of the IEEE international conference on computer vision*, pages 1582–1590, 2015.

Xiaochun Cao, Changqing Zhang, Chengju Zhou, Huazhu Fu, and Hassan Foroosh. Constrained multi-view video face clustering. *IEEE Transactions on Image Processing*, 24(11):4381–4393, 2015.

M Moonen and J Ramos. A subspace algorithm for balanced state space system identification. *IEEE Transactions on Automatic Control*, 1993.

P Van Overschee and B De Moor. N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 1994.

M Viberg. Subspace-based methods for the identification of linear time-invariant systems. *Automatica*, 1995.

T Katayama. *Subspace Methods for System Identification*. Springer, 2006.

L Song, B Boots, S Siddiqi, G Gordon, and A Smola. Hilbert space embeddings of hidden Markov models. In *International Conference on Machine Learning*, 2010.

B Boots and G Gordon. Two-manifold problems with applications to nonlinear system identification. In *International Conference on Machine Learning*, 2012.

S Andersson. Subspace estimation and prediction methods for hidden Markov models. *The Annals of Statistics*, 2009.

D Hsu, S Kakade, and T Zhang. A spectral algorithm for learning Hidden Markov Models. *Journal of Computer and System Sciences*, 2012.

D Foster, J Rodu, and L Ungar. Spectral dimensionality reduction for HMMs. arXiv:1203.6130v1, 2012.

H Zhao and P Poupart. A sober look at spectral learning. arXiv:1406.4631, 2014.

H Cramér. *Mathematical Methods of Statistics*. Princeton University Press, 1946.

A Banerjee, S Merugu, I Dhillon, and J Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 2005.

Z Ghahramani and G.E. Hinton. Parameter estimation for linear dynamical systems. Technical report, 1996.

S Roweis and Z Ghahramani. A unifying review of linear Gaussian models. *Neural Computation*, 1999.

L Ljung. *System Identification (2Nd Ed.): Theory for the User*. Prentice Hall PTR, 1999.

K Åström. Maximum likelihood and prediction error methods. *Automatica*, 16 (5):551–574, 1980.

X Zhang, Y Yu, and D Schuurmans. Accelerated training for matrix-norm regularization: A boosting approach. In *Advances in Neural Information Processing Systems*, 2012.

Y Yu, X Zhang, and D Schuurmans. Generalized Conditional Gradient for Sparse Estimation, 2014. arXiv:1410.4828.

A Gelfand, P Diggle, P Guttorp, and M Fuentes. *Handbook of Spatial Statistics*. CRC Press, 2010.

S Siddiqi, B Boots, and G Gordon. A Constraint Generation Approach to Learning Stable Linear Dynamical Systems. In *Advances in Neural Information Processing Systems*, 2007.

M White, J Wen, M Bowling, and D Schuurmans. Optimal estimation of multivariate ARMA models. In *AAAI Conference on Artificial Intelligence*, 2015.

HF Yu, N Rao, and I Dhillon. High-dimensional Time Series Prediction with Missing Values. arXiv:1509.08333, 2015.

Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 1993.

Martha White. Regularized factor models, 2009.

M White, Y Yu, X Zhang, and D Schuurmans. Convex multi-view subspace learning. In *Advances in Neural Information Processing Systems*, 2012.

L Buesing, J Macke, and M Sahani. Learning stable, regularised latent models of neural population dynamics. *Network: Computation in Neural Systems*, 2012a.

F Bach, J Mairal, and J Ponce. Convex sparse matrix factorizations. arXiv:0812.1869v1, 2008.

A Beck and M Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2, 2009.

L Buesing, J Macke, and M Sahani. Spectral learning of linear dynamics from generalised-linear observations with application to neural population data. In *Advances in Neural Information Processing Systems*, 2012b.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Advances in neural information processing systems*, pages 3738–3746, 2016.

Luigi Antelmi, Nicholas Ayache, Philippe Robert, and Marco Lorenzi. Sparse multi-channel variational autoencoder for the joint analysis of heterogeneous data. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 302–311, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

Dean P Foster, Sham M Kakade, and Tong Zhang. Multi-view dimensionality reduction via canonical correlation analysis. 2008.

Weiran Wang, Raman Arora, Karen Livescu, and Jeff A Bilmes. Unsupervised learning of acoustic features via deep canonical correlation analysis. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4590–4594. IEEE, 2015b.

Yuhong Guo. Convex subspace representation learning from multi-view data. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

S Boyd and L Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

N Parikh and S Boyd. *Proximal Algorithms*. Foundations and Trends in Optimization. Now Publishers, 2013.

B Haeffele, E Young, and R Vidal. Structured Low-Rank Matrix Factorization: Optimality, Algorithm, and Applications to Image Processing. In *International Conference on Machine Learning*, 2014.

J Macke, L Buesing, and M Sahani. Estimating State and Model Parameters in State-Space Models of Spike Trains. *Advanced State Space Methods for Neural and Clinical Data*, 2015.

Appendix A

Invertible Convolutional Flow

A.1 Model architecture and training procedure

A.1.1 Density estimation

To train the model, we used the Adam optimizer [Kingma and Ba, 2014] with initial learning rate of .001 which was decayed slowly to 0.0001 with exponentially decaying of rate .97. We apply `sigm()` to the output of conditioning network to obtain the scaling filters, $\boldsymbol{s}$ and the convolution kernels at the frequency domain, $\boldsymbol{w}_f$. Actnorm [Kingma and Dhariwal, 2018] is employed as normalization bijector in the chain of flow and as a layer in the NN. An $l2$ regularizer with coefficient of 5e-5 is applied on all the weights. Also to control overfitting, we use dropout layer with $p_{drop} = .2$ for MNIST. To transform MNIST data from a bounded to an unbounded domain, a logit mapping of the form $y = \text{logit}(\alpha + (1-\alpha)\frac{x}{256})$ is applied with $\alpha = 10^{-6}$. All datasets are dequantized by adding uniform distributed noise to each dimension, and then they are scaled to $[0, 1]$ values.

The aforementioned setting is used for both density estimation experiments in Table 2.1 and Table 2.2.

Normalizing flow architecture, NN architecture for parameter generation and other hyper parameters of the results reported in Table 2.1 are outlined in Table A.1. Squeezing from space to channel dimension is applied Q times and

followd by K flow step after each squeeze, that is showed in the format $Q \times K$ for MNIST and CIFAR10 in the Table. No factor out (splitting) is used. The squeeze and convolution together can be interpreted as dilated convolution of factor 2. Although, we used 2D invertible convolution flow for these two datasets but the general purpose fully connected feedforward conditioning NN is applied for parameter generation.

Table A.1: Hyper parameters of the results reported in Table 2.1.

	normalizing flow architecture		NN architecture		
Dataset	# flow steps	M (itertes per step)	# layers	# hidden units	Minibatch size
POWER	10	2	2	200	10000
GAS	10	2	2	100	10000
BSDS300	10	1	2	512	10000
MNIST	2×5	1	2	1024	512
CIFAR10	3×4	2	2	1024	512

For the CNN based NN experiments of Table 2.2, the results of realNVP and GLOW on CIFAR10 dataset are adopted from Kingma and Dhariwal [2018]. GLOW uses multiscale architecture with 3 scales each one composed of 32 steps of flow and use different shallow neural networks with 2 hidden layers and 512 channels (width) for each parameter of the flow. Splitting is performed on the channels dimension only. After each scale a factor out with rate 1/2 is applied. We used the same architecture except that we use one NN to generate all parameters of a flow step but we doubled its width to 1024 channels. For MNIST, we again followed similar architecture for the normalizing flow where 2 scales each one composed of 12 steps of flow. The NN of depth 2 hidden layers with width of 512 channels are applied as the conditioning network. The results of realNVP and GLOW on MNIST dataset are adopted from Grathwohl et al. [2019] where they used the following flow structure:

$$2 * (3 * (\text{coupling layers with checkerboard masking}) + \text{squeeze} + 3 * (\text{coupling layers with channel masking})) + 4 * (\text{coupling layers with channel masking})$$

Each CONF is composed of $M = 2$ iterates of convolution-multiplication

on both datasets.

A.1.2 Variational inference

We employed the encoder/decoder architecture of van den Berg et al. [2018] with different optimization setting. We apply `exp()` to the output of encoder to obtain the scaling filters, $\boldsymbol{s}$ and the convolution kernels at the frequency domain, $\boldsymbol{w}_f$. Minibatch size of 500 samples (100 for FreyFaces) is selected and the other hyper parameters are adjusted according to get better training. The Adam optimizer [Kingma and Ba, 2014] is used for training with learning rate decaying from initial value lr_{init} to $.1 \times lr_{init}$ after warmup.

The annealing, a.k.a. warm-up, procedure is used that gradually increase the effect of KL divergence term in the loss function Sønderby et al. [2016], but we found that, on FreyFaces dataset, our model train better without warm-up. The hyper-parameters are summarized in Table A.2.

Table A.2: Hyper parameters of VAE results reported in Table 2.4.

Dataset	Minibatch size	# warmup	lr	ϵ_{Adam}
MNIST	500	100	0.001	0.1
Omniglot	500	100	0.001	0.1
FreyFaces	100	0	0.0005	0.1
Caltech	500	2000	0.001	0.1

A.2 Another symmetric convolution

There exist different extensions, here we define another type that can have straightforward interpretation. Let a base sequence be extended by an even-symmetric operation $\varepsilon\{.\}$ around its last element as

$$\hat{\boldsymbol{x}}(n) = \varepsilon\{\boldsymbol{x}(n)\} := \begin{cases} \boldsymbol{x}(n) & n = 0, 1, ..., N \\ \boldsymbol{x}(2N - n) & n = N + 1, ..., 2N - 1 \end{cases} \tag{A.1}$$

this type of even-symmetric expansion is depicted in Figure A.1. Again, the

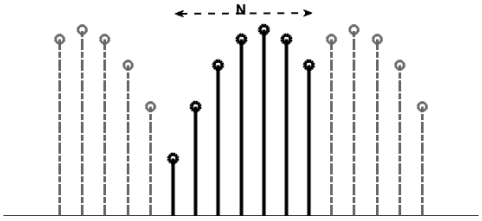


Figure A.1: Even-symmetric expansion around first and last element of the base sequence, where the base sequence specified by dark solid lines.

symmetric convolution of two sequences can be defined in terms of the circular convolution of their corresponding even-symmetric extensions as $\boldsymbol{y} = \boldsymbol{w} *_s \boldsymbol{x} = \mathcal{R}\{\hat{\boldsymbol{x}} \circledast \hat{\boldsymbol{w}}\}$ and also the convolution-multiplication property holds for this type given the discrete cosine transform defined as

$$\boldsymbol{x}_c(k) = \mathcal{F}_{dct}\{\boldsymbol{x}\}_k = \sum_{n=0}^{N} \boldsymbol{x}(n) \times 2\alpha_n \cos\left(\frac{\pi k n}{N}\right) \tag{A.2}$$

$$\text{where } \alpha_n = \begin{cases} 1/2 & n = 0, N \\ 1 & otherwise \end{cases}$$

This is called DCT-I in the literature. It can be shown that the Jacobian matrix of this transform have the following structure

$$\boldsymbol{J}_S = \begin{bmatrix} w_0 & w_1 + w_1 & \dots & w_{N-2} + w_{N-2} & w_{N-1} \\ w_1 & w_0 + w_2 & \dots & w_{N-3} + w_{N-1} & w_{N-2} \\ \vdots & \vdots & & \vdots & \vdots \\ w_{N-2} & w_{N-3} + w_{N-1} & \dots & w_0 + w_2 & w_1 \\ w_{N-1} & w_{N-2} + w_{N-2} & \dots & w_1 + w_1 & w_0 \end{bmatrix}$$

Since scaling a column or row of a square matrix with factor α, multiply its determinant by α, hence the multiplying the first and last column of this matrix

by factor of two give rise to

$$\boldsymbol{J}'_S = \begin{bmatrix} 2w_0 & w_1 + w_1 & \dots & w_{N-2} + w_{N-2} & 2w_{N-1} \\ 2w_1 & w_0 + w_2 & \dots & w_{N-3} + w_{N-1} & 2w_{N-2} \\ \vdots & \vdots & & \vdots & \vdots \\ 2w_{N-2} & w_{N-3} + w_{N-1} & \dots & w_0 + w_2 & 2w_1 \\ 2w_{N-1} & w_{N-2} + w_{N-2} & \dots & w_1 + w_1 & 2w_0 \end{bmatrix}$$

$$= \begin{bmatrix} w_0 & w_1 & \dots & w_{N-2} & w_{N-1} \\ w_1 & w_0 & \ddots & w_{N-3} & w_{N-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ w_{N-2} & w_{N-3} & \ddots & w_0 & w_1 \\ w_{N-1} & w_{N-2} & \dots & w_1 & w_0 \end{bmatrix} + \begin{bmatrix} w_0 & w_1 & \dots & w_{N-2} & w_{N-1} \\ w_1 & w_2 & \iddots & w_{N-1} & w_{N-2} \\ \vdots & \iddots & \iddots & \iddots & \vdots \\ w_{N-2} & w_{N-1} & \iddots & w_2 & w_1 \\ w_{N-1} & w_{N-2} & \dots & w_1 & w_0 \end{bmatrix}$$

where $\det(J'_S) = 4\det(J_S)$. Therefore, this symmetric convolution provides a structured Jacobian matrix that can be specified in terms of a Toeplitz matrix and an upside-down Toeplitz (also called a Hankel) matrix for determinant computation.

Appendix B

Deep Probabilistic Multi-view Learning

B.1 Proof of Theorem 3.2

The marginal mean and covariance matrix of the joint views $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2)$ under the linear probabilistic model (3.4) are $\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{W}_1\boldsymbol{\mu}_0 + \boldsymbol{\mu}_{\epsilon_1} \\ \boldsymbol{W}_2\boldsymbol{\mu}_0 + \boldsymbol{\mu}_{\epsilon_2} \end{pmatrix}$ and $\boldsymbol{\Sigma} = \boldsymbol{W}\boldsymbol{W}^\top + \boldsymbol{\Psi}$ where we define $\boldsymbol{W} = \begin{pmatrix} \boldsymbol{W}_1 \\ \boldsymbol{W}_2 \end{pmatrix}$ and $\boldsymbol{\Psi} = \begin{pmatrix} \boldsymbol{\Psi}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Psi}_2 \end{pmatrix}$, therefore, similar to the proof in [Bach and Jordan, 2005], the negative log-likelihood of the data can be written as

$$
\begin{aligned}
\ell_1 &= \frac{n(d_1+d_2)}{2}\log 2\pi + \frac{n}{2}\log|\boldsymbol{\Sigma}| \\
&\quad + \frac{1}{2}\sum_{j=1}^{n} \operatorname{tr} \boldsymbol{\Sigma}^{-1}(\mathbf{z}_j - \boldsymbol{\mu})(\mathbf{z}_j - \boldsymbol{\mu})^\top \\
&= \frac{n(d_1+d_2)}{2}\log 2\pi + \frac{n}{2}\log|\boldsymbol{\Sigma}| \\
&\quad + \frac{n}{2}\operatorname{tr}\boldsymbol{\Sigma}^{-1}\tilde{\boldsymbol{\Sigma}} + \frac{n}{2}(\tilde{\boldsymbol{\mu}} - \boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\tilde{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top
\end{aligned}
$$

Maximizing ℓ_1 with respect to $\boldsymbol{\mu}$ results in a maximum $\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{W}_1\boldsymbol{\mu}_0 + \boldsymbol{\mu}_{\epsilon_1} \\ \boldsymbol{W}_2\boldsymbol{\mu}_0 + \boldsymbol{\mu}_{\epsilon_2} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}$ and the negative log-likelihood is reduced to

$$\ell_1 = \frac{n(d_1 + d_2)}{2} \log 2\pi + \frac{n}{2} \log |\boldsymbol{\Sigma}| + \frac{n}{2} \operatorname{tr} \boldsymbol{\Sigma}^{-1} \tilde{\boldsymbol{\Sigma}}$$

The rest of the proof follows immediately along the line of proof in Bach and Jordan [2005].

B.2 Some proofs of section 3.3.2

B.2.1 Proof of additive property of KL (3.10)

Conditional independence of the latent variables $\{\mathbf{z}_m|\boldsymbol{\phi}\}$ induced by the probabilistic graphical model of latent linear layer (3.4) implies that the approximate posterior of the set of latent variables can be factorized as

$$q_\eta(\mathbf{z}|\mathbf{x}) = q_\eta(\boldsymbol{\phi}|\mathbf{x}) \prod_{m=1}^{M} q_\eta(\mathbf{z}_m|\boldsymbol{\phi}, \mathbf{x}). \tag{B.1}$$

In addition, assuming independent prior distribution on the latent variables, *i.e.* $p(\mathbf{z}) = p(\boldsymbol{\phi}) \prod_{m=1}^{M} p(\mathbf{z}_m)$ leads to

$$\begin{aligned} D_{\mathrm{KL}}[q_\eta(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})] &= \int q_\eta(\boldsymbol{\phi}|\mathbf{x}) \prod_{m=1}^{M} q_\eta(\mathbf{z}_m|\boldsymbol{\phi}, \mathbf{x}) \times \log \frac{q_\eta(\boldsymbol{\phi}|\mathbf{x}) \prod_{m=1}^{M} q_\eta(\mathbf{z}_m|\boldsymbol{\phi}, \mathbf{x})}{p(\boldsymbol{\phi}) \prod_{m=1}^{M} p(\mathbf{z}_m)} \\ &= \int q_\eta(\boldsymbol{\phi}|\mathbf{x}) \log \frac{q_\eta(\boldsymbol{\phi}|\mathbf{x})}{p(\boldsymbol{\phi})} + \sum_{m=1}^{M} \int q_\eta(\mathbf{z}_m|\boldsymbol{\phi}, \mathbf{x}) \log \frac{q_\eta(\mathbf{z}_m|\boldsymbol{\phi}, \mathbf{x})}{p(\mathbf{z}_m)} \\ &= D_{\mathrm{KL}}[q_\eta(\boldsymbol{\phi}|\mathbf{x})\|p(\boldsymbol{\phi})] + \sum_{m=1}^{M} D_{\mathrm{KL}}[q_\eta(\boldsymbol{\epsilon}_m|\mathbf{x})\|p(\boldsymbol{\epsilon}_m)] \blacksquare \end{aligned}$$

The final equation is derived by linear change of variable $\boldsymbol{\epsilon}_m = \mathbf{z}_m - \boldsymbol{W}_m\boldsymbol{\phi}_m$.

A similar derivation was also presented in [Antelmi et al., 2019] while both proofs, indeed, approve the fact that KL divergence is additive for independent distributions. ■

B.2.2 Proof of Lemma 3.1

Assuming isotropic multivariate Gaussian priors on the latent variables as $\boldsymbol{\phi} \sim \mathcal{N}(\mathbf{0}, \lambda_0^{-1}\mathbf{I})$, $\boldsymbol{\epsilon}_m \sim \mathcal{N}(\mathbf{0}, \lambda_m^{-1}\mathbf{I})$ and specifying the approximate posteriors as Gaussian distributed vectors with diagonal covariances results in closed form solutions for the KL divergence terms [Kingma and Welling, 2013] as

$$\begin{aligned} D_{\mathrm{KL}}[q_\eta(\boldsymbol{\phi}|\mathbf{x})\|p(\boldsymbol{\phi})] &= \frac{1}{2}\lambda_0\|\boldsymbol{\mu}_0\|^2 + \\ &\quad \frac{1}{2}\sum_{i=1}^{d_0}(\lambda_0 - \log\lambda_0 - 1) \\ D_{\mathrm{KL}}[q_\eta(\boldsymbol{\epsilon}_m|\mathbf{x})\|p(\boldsymbol{\epsilon}_m)] &= \frac{1}{2}\lambda_m\|\boldsymbol{\mu}_{\epsilon_m}\|^2 + \\ &\quad \frac{1}{2}\sum_{i=1}^{d_m}(\lambda_m\sigma_{mi}^2 - \log\lambda_m\sigma_{mi}^2 - 1) \end{aligned}$$

Splitting the terms in the KL divergence to those depending on the mean variables and the remaining ones results

$$\begin{aligned} &\min_{\boldsymbol{\mu}_0} \frac{1}{2}\lambda_0\|\boldsymbol{\mu}_0\|^2 + \frac{1}{2}\sum_{m=1}^{M}\lambda_m\|\boldsymbol{\mu}_{\epsilon_m}\|^2 + \mathcal{K} \\ &\text{s.t. } \boldsymbol{\mu}_{\epsilon_m} = \boldsymbol{\mu}_m - \boldsymbol{W}_m\boldsymbol{\mu}_0, \quad \forall m \in \{1, ..., M\} \end{aligned}$$

Now, solving this constraint optimization problem using the method of Lagrange multipliers leads to the optimal minimizer

$$\boldsymbol{\mu}_0^* = (\lambda_0\mathbf{I} + \sum_{m=1}^{M}\lambda_m\boldsymbol{W}_m^\top\boldsymbol{W}_m)^{-1}(\sum_{m=1}^{M}\lambda_m\boldsymbol{W}_m^\top\boldsymbol{\mu}_m).$$

This provides an analytical approach to optimally recover $\boldsymbol{\mu}_0$ from the parameters of the model.

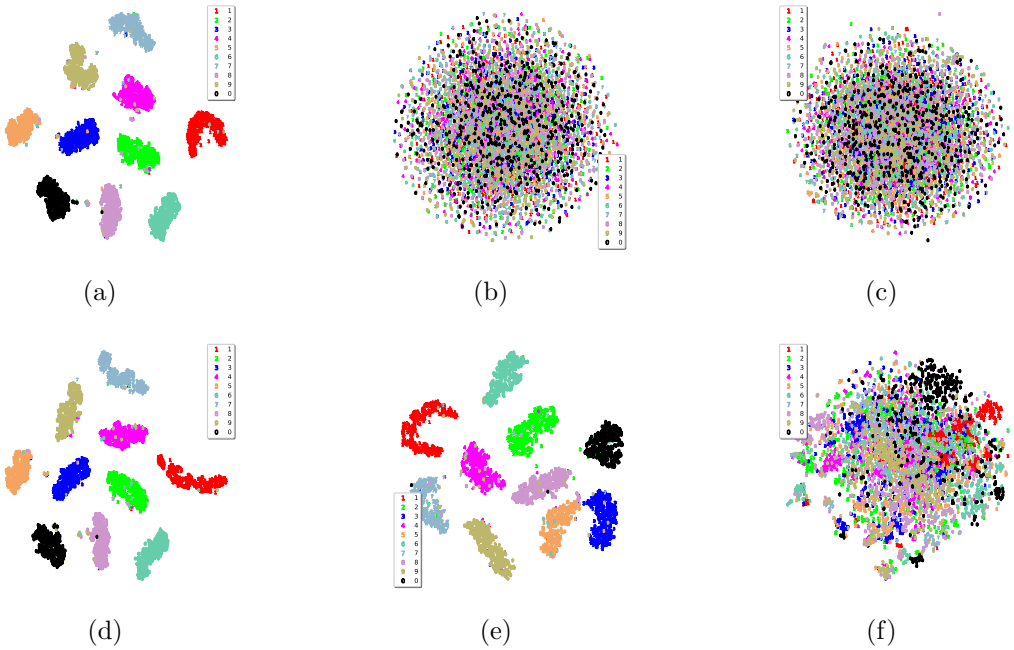
B.3 Extra experiments

B.3.1 Qualitative experiments

2-D embedding

Figures B.1 and B.2 depict the 2D t-SNE embeddings of the shared latent representations for multi-view and multi-modal setting, respectively. They verify that the representation of the images of different classes are well separated in the shared latent space.

From B.1(e), the mean of residual error of the first view still contains the categorical information which suggests that the discriminative performance tasks based on the latent variable can be enhanced by employing some constrains that enforce disentanglement in recovering process of the $\boldsymbol{\mu}_0$.



(a) (b) (c)

(d) (e) (f)

Figure B.1: 2D t-SNE embedding of ***the samples of*** *(a) shared representation* $\boldsymbol{\phi} \sim q_\eta(\boldsymbol{\phi}|\mathbf{x})$*, (b) residual error of the first view* $\boldsymbol{\epsilon}_1 \sim q_\eta(\boldsymbol{\epsilon}_1|\mathbf{x})$ *and (c) residual error of the second view* $\boldsymbol{\epsilon}_2 \sim q_\eta(\boldsymbol{\epsilon}_2|\mathbf{x})$*. Also, 2D t-SNE embedding of* ***the mean of*** *(d) shared representation* $\boldsymbol{\mu}_0$*, (e) residual error of the first view* $\boldsymbol{\mu}_{\epsilon_1}$ *and (f) residual error of the second view* $\boldsymbol{\mu}_{\epsilon_2}$*.* ***Multi-view*** *setting is applied,* i.e. *only the primary view is available at the test time so the equation (3.13) is used to recover* $\boldsymbol{\mu}_0$*.*

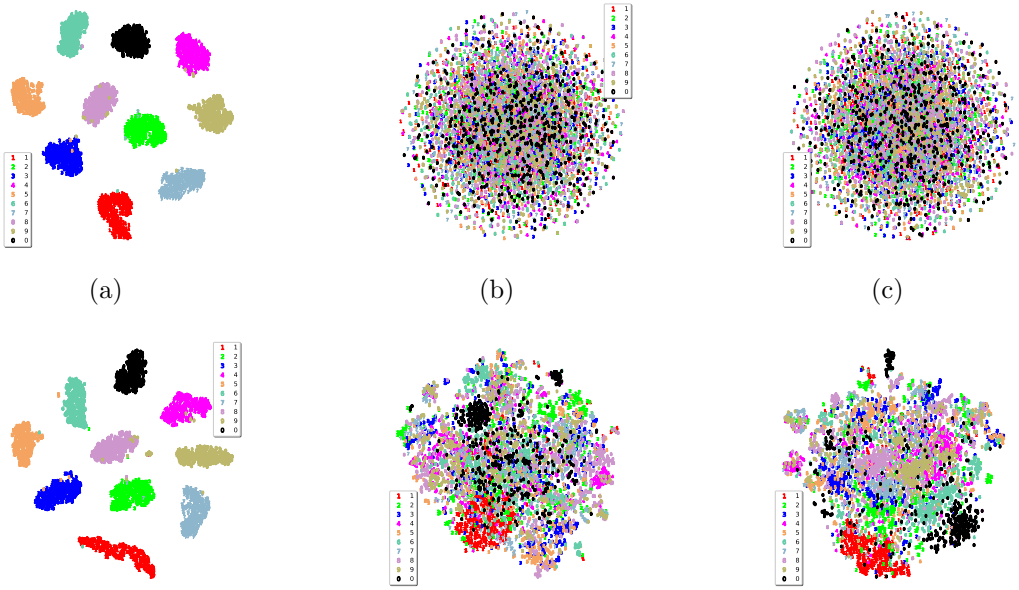


Figure B.2: 2D t-SNE embedding of ***the samples of*** *(a) shared representation* $\boldsymbol{\phi} \sim q_\eta(\boldsymbol{\phi}|\mathbf{x})$*, (b) residual error of the first view* $\boldsymbol{\epsilon}_1 \sim q_\eta(\boldsymbol{\epsilon}_1|\mathbf{x})$ *and (c) residual error of the second view* $\boldsymbol{\epsilon}_2 \sim q_\eta(\boldsymbol{\epsilon}_2|\mathbf{x})$*. Also, 2D t-SNE embedding of* ***the mean of*** *(d) shared representation* $\boldsymbol{\mu}_0$*, (e) residual error of the first view* $\boldsymbol{\mu}_{\epsilon_1}$ *and (f) residual error of the second view* $\boldsymbol{\mu}_{\epsilon_2}$*.* ***Multi-modal*** *setting is applied,* i.e. *both views are available at the test time so the equation (3.12) is used for estimation of* $\boldsymbol{\mu}_0$*.*

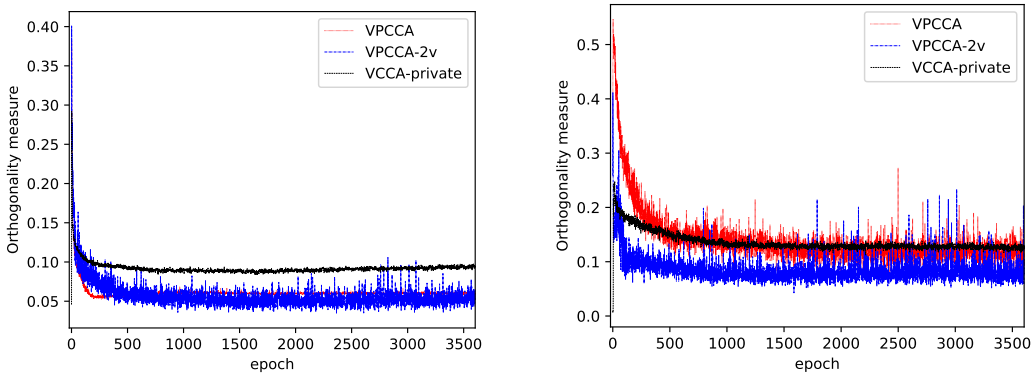
Disengagement of the shared and view specific factors

To quantify the disentanglement, we define the normalized orthogonality measure as

$$\text{orth}(\boldsymbol{\Phi}, \boldsymbol{E}) := \frac{\|\boldsymbol{\Phi}\boldsymbol{E}^\top\|_F}{\sqrt{\|\boldsymbol{E}\boldsymbol{E}^\top\|_F}\sqrt{\|\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\|_F}}$$

where matrices $\{\boldsymbol{\Phi}, \boldsymbol{E}\}$ are composed of the vectors of samples $\boldsymbol{\phi}, \boldsymbol{\epsilon}$.

This orthogonality measure is evaluated over the course of training and plotted in figure B.3 for the first and the second view of the noisy MNIST dataset, which show that VPCCA and VPCCA-2v models achieve more dissimilarity between the shared and view-specific factors.



(a) Orthogonality between $\boldsymbol{\phi}$ and $\boldsymbol{\epsilon}_1$ (b) Orthogonality between $\boldsymbol{\phi}$ and $\boldsymbol{\epsilon}_2$

Figure B.3: Orthogonality measure between (a) shared representation and 1st view-specific factor orth$(\boldsymbol{\Phi}, \boldsymbol{E}_1)$ (b) shared representation and 1st view-specific factor orth$(\boldsymbol{\Phi}, \boldsymbol{E}_2)$ where the columns of $\boldsymbol{\Phi}, \boldsymbol{E}_m$ are drawn from $\boldsymbol{\phi} \sim q_\eta(\boldsymbol{\phi}|\mathbf{x})$ and $\boldsymbol{\epsilon}_m \sim q_\eta(\boldsymbol{\epsilon}_m|\mathbf{x})$, respectively, and posteriors are evaluated on the validation set of noisy MNIST dataset.

B.3.2 Multimodal Yale-B with missing modalities at test time

We repeat the experiments for multi-modal facial datasets when subsets of modalities are missing and partial images are available at test time. These two different setting are evaluated: I) 3 modalities (eye+ nose+ mouth) are available at test time II) only one modality (eye) is available for downstream task. Here, the canonical correlation in VPCCA, $\boldsymbol{P}_{d_0}$, is built as NN function of the available modalities for downstream task and a modification of equation (3.12) is adopted for estimation of $\boldsymbol{\mu}_0$. The clustering results, compared in table B.1, show that the VPCCA works better when more modalities are available at test time while it is outperformed by VCCA-private in the case that most of modalities are missing, which, in this case, demonstrate that the proposed VPCCA can efficiently integrate higher number of modalities in learning richer representation.

	All modalities			**3 modalities (eye+ nose+ mouth)**			**1 modality (eye)**		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
VCCA-private	97.52	98.09	96.07	97.43	98.40	96.17	96.15	97.56	94.64
VPCCA	99.72	99.56	99.22	98.47	98.81	97.07	95.61	97.50	92.33

*Table B.1: Comparison of clustering performance between VPCCA and VCCA for multi-modal facial datasets when subset of modalities are missing. To estimate $\boldsymbol{\mu}_0$, here, the **VPCCA** uses a modified version of equation (3.12) by dropping the terms whose views are missing at test time. The clustering performance are averaged over 3 trails and the higher the better.*

B.4 Model architecture and training procedure

B.4.1 Two-view noisy MNIST experiments

The parameters of each algorithm are tuned through cross validation with grid search over $p_{dropout} \in \{.0, .2\}$ the variance of the shared representation $\boldsymbol{\phi}$ $\lambda_0^{-1} \in \{100., 500., 2000., 5000.\}$ and equal variance for residual errors $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2$ in range $\lambda_0^{-1} \in \{8., 4., 2., 1., .5, .25, .125\}$. Results are averaged over 3 trails.

The dimensionality of the shared representation was $d_0 = 30$ and the dimensionality of the latent factors were $d_1 = d_2 = 60$.
Weight decay of 0.0001 was applied as the regularization for all the parameters of NNs.

B.4.2 Multi-modal clustering experiments

Digits: The dimensionality of the shared representation was $d_0 = 30$ and the dimensionality of the latent factors were $d_1 = d_2 = 60$.

The parameters of each algorithm are tuned through cross validation with grid search over $p_{dropout} \in \{.0, .2\}$ the variance of the shared representation $\boldsymbol{\phi}$ $\lambda_0^{-1} \in \{1., 5., 20., 100., 500., 2000.\}$ and equal variance for residual errors $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2$ in range $\lambda_0^{-1} \in \{8., 4., 2., 1., .5, .25, .125\}$.

Yale-B facial components: The dimensionality of the shared representation was $d_0 = 120$ and the dimensionality of the latent factors were $d_1 = d_2 = 160$.

The parameters of each algorithm are tuned through cross validation with grid search over $p_{dropout} \in \{.0, .2\}$ the variance of the shared representation $\boldsymbol{\phi}$ $\lambda_0^{-1} \in \{.2, 1., 5., 20., 100., 500., 2000.\}$ and equal variance for residual errors $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2$ in range $\lambda_0^{-1} \in \{8., 4., 2., 1., .5, .25, .125\}$.

In both experiments a weight decay of 0.0001 was applied as the regularization for all the parameters of NNs.

Appendix C

Deep Generative Multi-view Learning

C.1 Introduction

The problem of multi-view learning is studied extensively in the literature and its merits has been demonstrated in extracting richer representation from available multiple views at the training time [Chaudhuri et al., 2009] [Hardoon et al., 2004] [Foster et al., 2008]. To capture nonlinearity in the model, one can either use kernel methods or follow the recent growing path of the deep neural network (DNN). Both of these methods have been explored in the literature and researchers proposed some advanced two-view models [Hardoon et al., 2004, Bach and Jordan, 2003] [Andrew et al., 2013]. Kernel based methods, such as KCCA [Hardoon et al., 2004], require large memory to store a massive amount of training data to use at the test time. To overcome this issue and improve the kernel based method in terms of memory and speed, some kernel approximation techniques based on random sampling of training data are proposed in [Williams and Seeger, 2001] and [Lopez-Paz et al., 2014]. On the other hand, the main advantage of the DNN over kernel based method is that, its parametric model can be better trained with larger amount of data using the fast stochastic optimization techniques.

The proposed deep two-view methods can be mainly categorized in two groups. On one hand, there are models inspired by auto-encoder, e.g. split autoencoder (SplitAE) of [Ngiam et al., 2011], in which the deep autoencoders are trained so that the reconstruction error of both views are minimized. In this methods, the encoding network of both view are shared while each view has its own (split) decoder network. On the other hand, another pathway is based on canonical correlation analysis (CCA), such as deep CCA (DCCA) method [Andrew et al., 2013] that extends the linear single layer CCA to deep CCA in which the model parameters are estimated to maximize the cross correlation between the projection of both views.

To combine the benefits of both deep auto-encoder (AE) and CCA for multi-view datasets and hence enhance learned representation, the idea of *deep CCA-Auto encoder (DCCAE)* is proposed in [Wang et al., 2015a]. This method tries to optimize the following objective function that is combination of reconstruction errors of two autoencoders and the canonical correlation between the learned bottleneck features (the output of the deep encoders)

$$\begin{aligned}
\min_{W_f,W_g,W_p,W_q,U,V} \quad & -\tfrac{1}{T}\operatorname{tr}\mathbf{U}^T f(\mathbf{X})g(\mathbf{Y})^T\mathbf{V} \\
& + \tfrac{\lambda}{T}\sum_{i=1}^{T}\left(\|\mathbf{x}_i - p(f(\mathbf{x}_i))\|^2 + \|\mathbf{y}_i - q(g(\mathbf{y}_i))\|^2\right) \\
\text{s.t.} \quad & \tfrac{1}{T}\mathbf{U}^T f(\mathbf{X})f(\mathbf{X})^T\mathbf{U} = \mathbf{I} \\
& \tfrac{1}{T}\mathbf{V}^T g(\mathbf{Y})g(\mathbf{Y})^T\mathbf{V} = \mathbf{I} \\
& u_i^T f(\mathbf{X})g(\mathbf{Y})^T v_j = 0 \qquad \text{for } i \neq j
\end{aligned} \tag{C.1}$$

Here, the functions $\{f, g, p, q\}$ are flexible nonlinear mappings modeled by neural networks that are parameterized by the set of learnable parameters $\{W_f, W_g, W_p, W_q\}$. $\lambda > 0$ is a trade-off parameter that controls the reconstruction error and canonical correlation between the projected views in the objective function (C.1). In this equation, CCA term tries to maximize the mutual information between the projected views, $f(\mathbf{x}_i)$ and $g(\mathbf{y}_i)$, and AE loss tries to minimize the reconstruction error between views and their projections.

This approach was shown to outperform DCCA and SplitAE for classification and clustering tasks in two-view application [Wang et al., 2015a].

On the other hand, DCCAE has some drawbacks that limits its applications. Its main drawbacks are two folds. First, the objective function and the constraints couples all the training samples through the (cross-)covariance terms, this will block the stochastic optimization method (e.g. SGD) to be applied here in its standard form. Nevertheless, it was shown in [Wang et al., 2015b] that if the mini-batch size is large enough the stochastic gradient can approximate the true gradient but still this requires very large mini-batch sizes which imposes heavy computational complexity on the training algorithm. Second, it does not estimate the hidden state and a model that can generate the second view based on the observation from the primary (first) view. In addition, the empirical studies showed that the canonical term of the objective function (C.1) dominates in practice and hence the objective is less sensitive to the reconstruction error; this in turn result in the trained autoencoders that don't reconstruct the views very well while mainly trying to learn projected mapping $\mathbf{U}^T f(\mathbf{X}),\ \mathbf{V}^T f(\mathbf{Y})$ that are maximally correlated.

Wang et al. [2015a] also proposed a modification of their DCCAE method, in which the constraints are relaxed so that the feature dimensions are no longer required to be uncorrelated, the objective of this method, also called as *correlated autoencoder (CorrAE)*, is formulated as

$$\begin{aligned} \min_{W_f, W_g, W_p, W_q, U, V} \quad & -\tfrac{1}{T} \operatorname{tr} \mathbf{U}^T f(\mathbf{X}) g(\mathbf{Y})^T \mathbf{V} \\ & + \tfrac{\lambda}{T} \sum_{i=1}^{T} \left(\|\mathbf{x}_i - p(f(\mathbf{x}_i))\|^2 + \|\mathbf{y}_i - q(g(\mathbf{y}_i))\|^2 \right) \\ \text{s.t.} \quad & \tfrac{1}{T} \mathbf{u}_i^T f(\mathbf{X}) f(\mathbf{X})^T \mathbf{u}_i = \tfrac{1}{T} \mathbf{v}_i^T g(\mathbf{Y}) g(\mathbf{Y})^T \mathbf{v}_i = 1. \end{aligned} \tag{C.2}$$

This variation of the deep multi-view model is designed to examine the importance of the correlation among the learned feature dimensions by comparing its performance with that of the original DCCAE method in some learning tasks.

Deep Generative Multi-view (DGMV) model: On the other hand, it was shown by White et al. [2012] and Yu et al. [2014] that simple linear CCA can be expressed as a linear generative two-view form where the views are generated as perturbed linear model of the latent representation $\boldsymbol{\phi}_i$ as

$$\begin{cases} \mathbf{x}_i = \mathbf{C}\boldsymbol{\phi}_i + \epsilon_i, \\ \mathbf{y}_i = \mathbf{E}\boldsymbol{\phi}_i + \nu_i \end{cases} \tag{C.3}$$

where the perturbation terms are Gaussian independent and identically distributed (i.i.d.) vectors $\epsilon \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\epsilon)$ and $\nu \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\nu)$. This model makes the latent representation explicit and its joint model parameter estimation and latent variable inference can be expressed as a regularized loss objective function that can be reformulated as a convex optimization problem. We can generalize (C.3) to nonlinear model resulting in the deep nonlinear generative multi-view model

$$\begin{cases} \mathbf{x}_i = p(\boldsymbol{\phi}_i) + \epsilon_i, \\ \mathbf{y}_i = q(\boldsymbol{\phi}_i) + \nu_i \end{cases} \tag{C.4}$$

where the generative mappings $p(\boldsymbol{\phi}_i),\ q(\boldsymbol{\phi}_i)$ can be modeled by deep neural networks parameterized by W_p, W_q. Therefore, given the shared latent representation $\boldsymbol{\phi}_i$, two views can be generated by a non-linear mapping plus independent Gaussian noises hence one can formulate the following regularized loss objective function

$$\min_{W_p, W_q, \Phi} \quad \frac{1}{T}\sum_{i=1}^{T} \left(\|x_i - p(\phi_i)\|^2 + \|y_i - q(\phi_i)\|^2 \right) + \mathcal{R}(\Phi), \tag{C.5}$$

In this work, we tackle this deep multi-view subspace learning problem by introducing auto-encoders as inference model.

C.2 Problem Definition

As explained in the previous section, we prefer a deep multi-view network that offers a model to explicitly infer the shared latent source that generates both views and can predict the second view based on the available primary view at the test time. To this end, we introduce two auto-encoder networks with encoder (recognition) networks $f(),\ g()$ that provide latent projected views, $f_{\mathbf{x}_i} = f(\mathbf{x}_i)$ and $g_{\mathbf{y}_i} = g(\mathbf{y}_i)$, and the decoder (reconstruction) networks $p(\phi_i),\ q(\phi_i)$ that reconstruct each view based on the latent representation. The encoders and decoders can be modeled by deep neural networks with learnable parameter matrices $\{\mathbf{W}_f, \mathbf{W}_g, \mathbf{W}_p, \mathbf{W}_q\}$ that correspond to each deep model function. Inspired by the generative interpretation of linear CCA (C.3), we add a generative linear two-view layer, on top of auto-encoder in the latent space, in order to obtain a shared latent representation $\boldsymbol{\phi}_i$ for the pair of encoded projected $\{f_{\mathbf{x}_i},\ g_{\mathbf{y}_i}\}$. Since the auto-encoders reconstruct each individual view, the latent variable $\boldsymbol{\phi}_i$ indeed provides a shared underlying representation of both views in a deep nonlinear form. In the other words, the deep generative two-view network (DGMV) can be expressed mathematically as the following pairs of models

$$\begin{cases} \mathbf{x}_i = p(f_{\mathbf{x}_i}) + \epsilon_i, \\ \mathbf{y}_i = q(g_{\mathbf{y}_i}) + \nu_i \end{cases}, \qquad \begin{cases} f_{\mathbf{x}_i} = \mathbf{C}\boldsymbol{\phi}_i + \epsilon'_i, \\ g_{\mathbf{y}_i} = \mathbf{E}\boldsymbol{\phi}_i + \nu'_i \end{cases} \tag{C.6}$$

where $\mathbf{C}, \mathbf{E}$ are the factor loading matrices (matrices of basis) for each view and the latent representations vectors $\boldsymbol{\phi}_i$ are stacked in the matrix Φ. Figure C.1 depicts the graphical representation of this model. Consequently, the deep multi-view subspace learning problem can be formulated by the the following

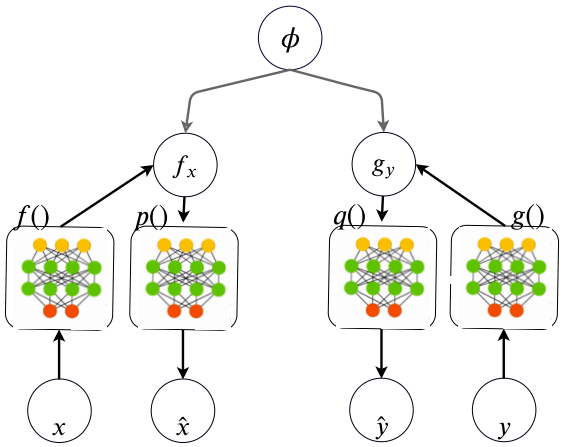


Figure C.1: Graphical representation of the deep generative two-view model.

combined regularized objective function

$$\min_{\mathbf{W}_f,\mathbf{W}_g,\mathbf{W}_p,\mathbf{W}_q,\mathbf{C},\mathbf{E},\Phi} \underbrace{\frac{\lambda}{T}\sum_{i=1}^{T}\|\mathbf{x}_i - p(f(\mathbf{x}_i))\|^2 + \|\mathbf{y}_i - q(g(\mathbf{y}_i))\|^2}_{\text{autoencodr objective terms}}$$
$$\underbrace{+\frac{1}{T}\sum_{i=1}^{T}\mathcal{L}_1\left(\mathbf{C}\phi_i; f(\mathbf{x}_i)\right) + \mathcal{L}_2\left(\mathbf{E}\phi_i; g(\mathbf{y}_i)\right) + \lambda_r\sum_{j=1}^{K}\mathcal{R}_1(\Phi_{j:})\mathcal{R}_2(\mathbf{C}_{:j},\mathbf{E}_{:j})}_{\text{linear two-view objective terms}} \tag{C.7}$$

Here, $\{\mathcal{L}_1,\ \mathcal{L}_2\}$ are the loss functions that measure the divergences between the latent projected views $\{f_{\mathbf{x}_i}, y_{\mathbf{y}_i}\}$ and their corresponding factorized estimates $\{\mathbf{C}\boldsymbol{\phi}_i, \mathbf{E}\boldsymbol{\phi}_i\}$. These losses are assumed to be convex in their first arguments, where different noise assumptions result different loss functions, for instance the i.i.d. Gaussian noise assumption amounts to ℓ_2 losses. The regularizer terms, $\mathcal{R}_1(\Phi_{j:}), \mathcal{R}_2(\mathbf{C}_{:j}, \mathbf{E}_{:j})$, capture special structures on the factors loading matrices and the latent features which are controlled by constant factor λ_r. On the other hands, the loss functions that measure the fitness error between each view and its reconstruction by the auto-encoder are modeled by ℓ_2 losses. Minimizing these loss terms results the latent projections that best reconstruct each view. The parameter $\lambda > 0$ balances the trade-off between the auto-encoder loss and the linear two-view loss.

C.2.1 Deep multi-view with conditionally independent views

One important assumption in multi-view learning is that the views are conditionally independent given the shared latent representation [Yu et al., 2014]. This property is crucial in some applications aiming to recover a natural latent representation. As explained in [White et al., 2012], this property can be encouraged by selecting regularizer terms of the form $\mathcal{R}_2(\mathbf{C}_{:j}, \mathbf{E}_{:j}) = \max\{\|\mathbf{C}_{:j}\|_2, \|\mathbf{E}_{:j}\|_2\}$ in the optimization objective (C.7). Using this regularizer, the basis of reconstruction models of each view are individually constrained and don't compete against each other to obtain their own share in reconstructing the views $\mathbf{x}_i,\ \mathbf{y}_i$, so this regularizer better respects the conditional independence of the views. Here, we select $\mathcal{R}_1(\Phi_{j:}) = \|\Phi_{j:}\|_2$ to encourage row-wise sparsity which, in turn, results in low-rank representation. Subsequently, the two-view objective terms in equation (C.7) can be reformulated as a convex optimization problem in the parameters of linear two-view model, $\{\mathbf{C}, \mathbf{E}, \mathbf{\Phi}\}$, [White et al., 2012, Yu et al., 2014]. Although, the combined objective function of the deep generative model (C.7) is not convex in the parameters of deep networks, we found this convex reformulation of the linear two-view layer to be beneficial for the training of deep two-view model and final latent variable in practice.

C.2.2 Advantages of the proposed model

- As mentioned above, the proposed method provide a model for inferring the hidden representation underlying both views and subsequently predicting the second view based on the available primary view at the test time. This is in contrast to CCA-based methods, such as DCCAE, that don't directly offer a model for generating samples from the latent variable so it is difficult to reconstruct one view based on the other one [Wang et al., 2015a].

- In addition, as opposed to CCA-based methods that require sufficiently large batch size in order to estimate the whitening matrices in the

constraints and the gradients, the average loss function (empirical risk) in (C.7) exhibits the standard summation form that enables random sampling for stochastic gradient calculation therefore the stochastic optimization algorithms can be readily employed here to optimize for deep network parameters.

- In contrast to the DCCAE that is limited to standard CCA formulation on the projected views, our proposed model is more flexible to include different types of losses for the two-view objective formulation to capture different properties of latent variables and hence is able to learn more complex models.

- Also, dissimilar to CCA based methods that are limited to two views, this generative model can be naturally extended to datasets with more than two views available at the training [Guo, 2013], so it can better integrate different information related to the same source to enhance representation learning.

- Additionally, it is expected that the reconstruction losses are more involved in deep generative multi-view training compared to DCCAE since all the objective terms in (C.7) has the form of losses. So, one might expect that other forms of losses can be replaced for the ℓ_2 of reconstruction error in the objective function (C.7) to improve reconstruction ability of the model; the property that doesn't seem practical in the DCCAE as its CCA term tends to dominate in practice while ignoring the reconstruction terms which in turn results in poor reconstructed views. This property will be investigated in the experimental studies in section C.3.

- Similar to the deep variational CCA model [Wang et al., 2016], we can introduce private variables that capture view-specific structures in the datasets and disentangle the underlying shared and private information in each view.

The combination of the aforementioned advantages, make the proposed deep generative two-view model a powerful and flexible candidate in multi-view set-

tings with different downstream goals such as classification, subspace clustering, speech recognition and word pair semantic similarity. In the following section we empirically study the performance of the proposed method.

C.3 Experiments

Experimental design For the experiments, we used the two-view noisy digits datasets of [Wang et al., 2015a] created based on MNIST dataset that consists of grayscale digit images of size 28×28 pixels. To synthesize the views, the pixel values are scaled to range $[0, 1]$. The first view of the dataset is generated by rotating each image at angles randomly sampled from uniform distribution $\mathcal{U}(-\pi/4, \pi/4)$ and the second view is selected from a different image of the same identity as in the first view and a random uniform noise is added, then the final value is truncated to remain in range $[0, 1]$. Following this procedure, both views are just sharing the same identity (label) of the digit but not the style of the handwriting as they are based on arbitrary images in the same class. The training set is divided into training/validation subsets of length $50K/10K$ and the performance is measured on the $10K$ images in the test set. This noisy MNIST two-view dataset was used in [Wang et al., 2015a] to evaluate the performance of the multi-view model.

To make a fair comparison, we used neural network architecture for the auto-encoders with the same capacity as the one used in [Wang et al., 2015a]. Accordingly, for the deep network models, the encoding networks are composed of three fully-connected nonlinear layers of size 1024 units and the last linear layer of size K where K is the dimensionality of the final mapping of the encoding network. The decoding networks consist of three fully-connected layers of 1024 nonlinear units with final layer of size 784 that reconstruct the original images. Sigmoid function is used as the nonlinearity in the deep auto-encoders. Here, we used sigmoid gate function for all the hidden units of the deep networks. In order to prevent over-fitting, we also applied stochastic drop-out to all the layers as regularization techniques.

In the experiments, the downstream task is classification and the misclas-

Method	Classification error (%)
Linear CCA (K=10)	19.6
SpliAE (K=10)	11.9
CorrAE (K=10)	12.9
DistAE (K=20)	16.0
KCCA (K=10)	5.1
DCCA (K=10)	2.9
DCCAE (K=10)	2.2
VCCA	3.0
VCCA-private	2.4
DGMV (K=50)	**1.32**
DGMV (K=70)	**1.30**

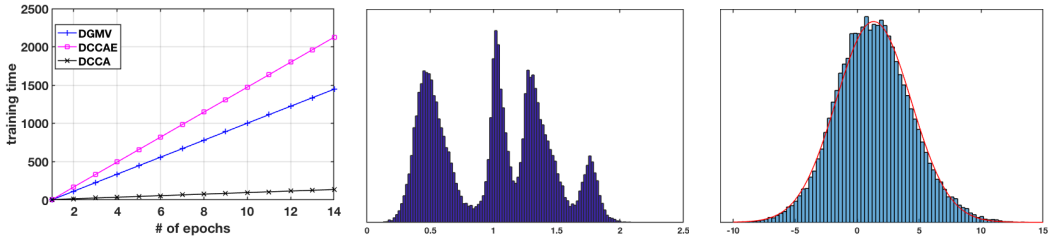
Table C.1: Classification error of different multi-view learning algorithms on a two-view data set generated based on the MNIST digit images. The results of DGMV method are averaged over 3 rials. The performance of the DGMV is compared against the following benchmark methods: ***Linear CCA****: linear single layer CCA,* ***SplitAE****: split autoencoder with Sigmoid gates [Ngiam et al., 2011] ,* ***DCCA****: deep CCA with Sigmoid gates [Andrew et al., 2013],* ***Randomized KCCA****: randomized kernel CCA approximation with Gaussian RBF kernels and random Fourier features [Lopez-Paz et al., 2014],* ***CorrAE****: deep correlated auto-encoder with Sigmoid gates (C.2) [Wang et al., 2015a],* ***DistAE****: deep minimum-distance auto-encoder with Sigmoid gates [Wang et al., 2015a],* ***DCCAE****: deep CCA-Auto encoder with Sigmoid gates (C.1) [Wang et al., 2015a],* ***VCCA****: deep variational CCA with ReLU gates [Wang et al., 2016],* ***VCCA-private****: deep variational CCA with an extra pair of latent variables for modeling the private information within each view. ReLU gates are used as the nonlinearities in all the networks [Wang et al., 2016], The performance results of the benchmark methods are from [Wang et al., 2015a, 2016].*

sification rate is measured as the performance metric. For that goal, the one-versus-one linear SVM classification algorithm is applied on the shared latent representation $\boldsymbol{\phi}$ of the proposed models or the projected mappings of the CCA based methods. It is worth emphasizing that the proposed DGMV model is able to infer the shared underlying representation of both views based on both encoding projections $\{f_{\mathbf{x}_i},\ g_{\mathbf{y}_i}\}$. The shared latent representation is not naturally available in the CCA-based methods which are only able to construct the projection of each individual view. To tune the parameters of the SVM algorithm, cross-validation procedure is employed selecting the best performing model, averaged over 3 trials, on the validation set and the final classification error is evaluated on the test set. For the proposed deep

multi-view models, we used the ℓ_2 loss function for both $\mathcal{L}_1, \mathcal{L}_2$ in the objective function (C.7). To train the deep generative multi-view (DGMV) model, the stochastic gradient descent is used for learning the parameters of the deep networks and accelerated proximal gradient descent [Karami et al., 2017] is employed for optimization of the latent two-view model while we alternatively switch between training of latent multi-view model and the deep AEs after each epoch of training while keeping the other set fixed. Furthermore, we practically found that the convex reformulation of the linear two-view model results in better performance than non-convex optimization algorithm for the training of the latent two-view model and inference of shared latent variable. Similar to [Wang et al., 2015a], deep auto-encoders are pre-trained using the layer-wise training method of restricted Boltzmann machines (RBMs) [Hinton and Salakhutdinov, 2006]. The parameters of each algorithm are tuned through cross validation with grid search.

Classification performance of different methods are presented in Table C.1 in bit error rate where the best dimensionality of latent variable for each method is reported in parenthesis. The results highlight that DGMV outperform the available methods in terms of the classification performance. In CCA based methods, the dimensionality of the projected latent variable, K, is selected from the set $\{5, 10, 20, 30, 50\}$ in [Wang et al., 2015a] and the best results are achieved by $K = 10$ while in our experiments we found that DGMV can benefit from larger projected latent variable size and it achieves better performance with larger K.

In order to evaluate the learning behavior of the methods, we also compare the running time of different learning algorithms in CPU seconds over the rounds (epochs) of optimization in Figure C.2(a). To make a fair comparison, all experiments were rerun on the same machine using Matlab. Comparing the computation times, we can see that the training of the proposed DGMV methods is faster than DCCAE and while the running time of DCCA is shorter per epoch but it needs more epochs of training (50 epochs versus 14 epochs used for DCCAE and deep two-view models) until it converge to a reasonable result.



(a) (b) (c)

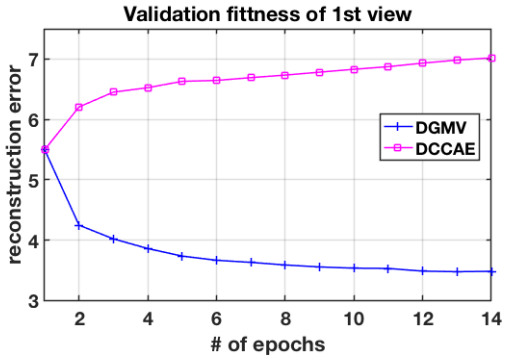
Figure C.2: (a) Running time of different learning algorithms over the rounds (epochs) of SGD optimization, (b) histogram of one dimensions of the primary projected view $f(x_i)$ of DCCAE and (c) histogram of one dimensions of the primary projected view $f(x_i)$ of DGMV.

Moreover, the histograms of projected view, depicted in Figure C.2(b) and C.2(c), confirm that the outputs of the encoders in DCCAE are not Gaussian distributed while CCA is known to work well in the Gaussian setting while on the other hand, the histograms of projected view of deep generative multi-view model in Figure C.2(c) shows that its distribution is approximately Gaussian.

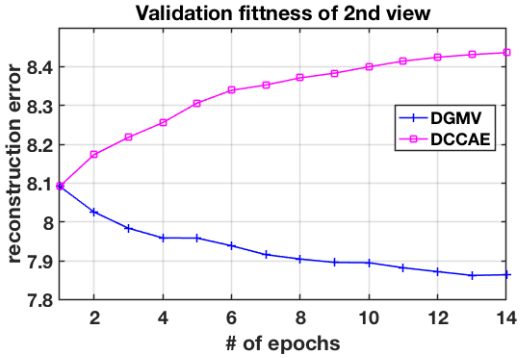
C.3.1 Reconstruction Performance

To examine the sample generation behavior of the proposed method, the reconstruction performance of the proposed methods is also evaluated and compared against that of DCCAE. First, the reconstruction error of each view is evaluated for different methods with latent variable dimensionality of $K = 10$. As the validation fitness over the course of training in Figure C.3 illustrates, DGMV tends to decrease the reconstruction errors of both views as the training algorithm progresses while DCCAE leads to increased reconstruction error to achieve smaller canonical correlation among the projected views. This empirical study shows that DCCAE sacrifices the reconstruction ability and focuses on canonical correlation term in order to achieve good discrimination performance while accurate reconstruction of input signal is highly desirable in practice.

Also to illustrate the reconstruction capability of the proposed method, some training samples of digits in both views and their reconstructed images are depicted in Figures C.4(a) and C.4(b) each reconstruction image is generated by its own autoencoder network. Figure C.4(c) depicts the predicted images of



(a) validation fitness of the 1st view



(b) validation fitness of the 2nd view

Figure C.3: Reconstruction fitness of both views for different learning algorithms over the rounds (epochs) of optimization.

the second view based on the 1st view using the combined network: *1st encoder* $(f())$ $\rightarrow$ *latent linear multi-view on the encoded projections* $\rightarrow$ *2nd decoder* $(q())$. Here, the network is trained with latent variable dimensionality of $K = 70$. These figures shows the reconstruction capability of DGMV method where the generated samples in the input space can denoise the noisy observation, the ability that was missing in DCCA and DCCAE. More specifically, one can observe from Figure C.4(c) that the rotations of images in the first view are eliminated from the generated images in the second view and a prototypical image of same digit is reconstructed by feeding a sample from that digit class to the network. This observation, which is also reported in [Wang et al., 2016] for variational CCA (VCCA) model, can be justified by the fact that the 2nd view only contains the class information of the 1st view but not its style and the rotation so the trained autoencoder of the second view (AE2) will ignore

(a) (b) (c)

Figure C.4: (a) Samples of the training dataset in the first views and their reconstructed images generated by autoencoder network of view 1 (AE1) depicted in columns 1 and 2, respectively. (b) Samples of the training dataset in the first views and their reconstructed images generated by autoencoder network of view 2 (AE2) depicted n columns 1 and 2, respectively. (c) Column 3 is the predicted images of the second view based on the samples from the first (primary) view of the test dataset in column 1. The second column shows the observed noisy samples of the second view.

the style information of the 1st view.

C.4 Conclusion and Discussion

In this work, a new deep generative multi-view model is proposed that extends the linear generative interpretation of classical CCA to a nonlinear deep architecture. The proposed deep multi-view network provides a model for inferring the hidden representation underlying both views that subsequently provides better class separation and also reconstruction. Furthermore, training of the model parameters enjoys the stochastic optimization algorithms that provide fast and efficient learning. This deep network can generate samples in the input space, so it can be employed to reconstruct one view based on available primary view at the test time. In addition to its denoising capability, this method also showed the potential to suppress more complex forms of distortion,

such as random rotation, from the signal. While CCA based methods achieve good discrimination performance at the expense of sacrificing the reconstruction error, the proposed method offers both class separation and sample generation in a more flexible way.

Appendix D

Linear Dynamical System Identification

D.1 Proof of Theorems 4.1 and 4.2

Proof: [**Theorem 4.1**] Let $x = [x_1; x_2]$ $y = [\mathbf{v}_1; \mathbf{v}_2]$ then the proximal operator of $f(x) = \lambda \max(\|x_1\|_2, \|x_2\|_2)$ can be expressed as

$$\begin{aligned} u = \text{prox}_{f(\mathbf{v})} &= \arg\min_x \frac{1}{2}\|x - y\|_2^2 + f(x) \\ &= \arg\min_{x_1,x_2} \frac{1}{2}\|x_1 - \mathbf{v}_1\|_2^2 + \frac{1}{2}\|x_2 - \mathbf{v}_2\|_2^2 \\ &\quad + \lambda \max(\|x_1\|_2, \|x_2\|_2). \end{aligned} \tag{D.1}$$

Now, splitting the search space into two subspace $S_1 = \{x | \|x_1\|_2 \geq \|x_2\|_2\}$ and its complement space $S_2 = \{x | \|x_2\|_2 \geq \|x_1\|_2\}$. If we confine our search to subspace S_1 the optimization problem (D.1) can be reformulated as the following convex constraint problem

$$\begin{aligned} &\min_{x_1,x_2} \frac{1}{2}\|x_1 - \mathbf{v}_1\|_2^2 + \frac{1}{2}\|x_2 - \mathbf{v}_2\|_2^2 + \lambda\|x_1\|_2. \\ &\text{subject to } \|x_1\|_2 \geq \|x_2\|_2 \end{aligned} \tag{D.2}$$

According to KarushKuhnTucker (KKT) optimality conditions for convex

problem [Boyd and Vandenberghe, 2004], a point $x^* = [x_1^*; x_2^*]$ is optimal point of this optimization problem if the following conditions are satisfied:

$$\begin{aligned}
(x_2^* - \mathbf{v}_2) + \nu x_2^* / \|x_2^*\|_2 &= 0 \\
(x_1^* - \mathbf{v}_1) + \lambda x_1^* / \|x_1^*\|_2 - \nu x_1^* / \|x_1^*\|_2 &= 0 \\
\|x_1^*\|_2 - \|x_2^*\|_2 &\geq 0 \\
\nu(\|x_1^*\|_2 - \|x_2^*\|_2) &= 0 \\
\nu &\geq 0
\end{aligned} \tag{D.3}$$

where ν is the Lagrange multiplier for the inequality constraint. Solving (D.3) for $x_1^*,\ x_2^*$ and ν one can readily obtain

$$\begin{bmatrix} x_1^* = \mathbf{v}_1 * \max\{1 - \frac{\nu}{\|\mathbf{v}_1\|}, 0\} \\ x_2^* = \mathbf{v}_2 * \max\{1 - \frac{\lambda - \nu}{\|\mathbf{v}_2\|}, 0\} \end{bmatrix} \tag{D.4}$$

According to the slackness condition () if $\|x_1^*\|_2 - \|x_2^*\|_2 \geq 0$ then $\nu = 0$ or if $\nu > 0$ then $\|x_1^*\|_2 = \|x_2^*\|_2$. Therefore the optimal ν can be obtained as

$$\begin{cases} \nu = 0 & \text{if } \|\mathbf{v}_1\| + \lambda < \|\mathbf{v}_2\| \\ \nu = .5(\|\mathbf{v}_1\| - \|\mathbf{v}_2\| + \lambda) & \text{if } \|\mathbf{v}_1\| \leq \|\mathbf{v}_2\| \leq \|\mathbf{v}_1\| + \lambda \end{cases}$$

Hence, the optimum solution under S_1 is

$$\begin{cases} \begin{bmatrix} x_1^* = \mathbf{v}_1 \\ x_2^* = \mathbf{v}_2 * \max\{1 - \frac{\lambda}{\|\mathbf{v}_2\|}, 0\} \end{bmatrix} & \text{if } \|\mathbf{v}_1\| + \lambda < \|\mathbf{v}_2\| \\ \begin{bmatrix} x_1^* = \mathbf{v}_1 * \max\{1 - \frac{\nu}{\|\mathbf{v}_1\|}, 0\} \\ x_2^* = \mathbf{v}_2 * \max\{1 - \frac{\lambda - \nu}{\|\mathbf{v}_2\|}, 0\} \end{bmatrix} & \text{if } \|\mathbf{v}_1\| \leq \|\mathbf{v}_2\| \leq \|\mathbf{v}_1\| + \lambda \end{cases} \tag{D.5}$$

We can repeat the same approach to obtain the optimal solution for the complement subspace S_2. ■

Lemma D.1 *Let* $G_{\lambda f}(x,v) := \frac{1}{2}\|x-v\|_2^2+\lambda f(x)$ *therefore the Moreau envelope of function* λf *is defined as* $\mathrm{M}_{\lambda f}(\mathbf{v}) := \min_x G_{\lambda f}(x,v)$ *[Parikh and Boyd, 2013].*
a) $\mathrm{M}_{\lambda f}(\mathbf{v}) = G_{\lambda f}(\mathrm{prox}_{\lambda f(\mathbf{v})}, v)$
b) If $f(x) = \|x\|_{2v} = \max(\|x_1\|_2, \|x_2\|_2)$ *we have*

$$\mathrm{M}_{\lambda\|.\|_{2v}}(\mathbf{v}) = \max_{0\le\gamma\le1} \mathrm{M}_{\lambda\gamma\|.\|_2}(\mathbf{v}_1) + \mathrm{M}_{\lambda(1-\gamma)\|.\|_2}(\mathbf{v}_2) \tag{D.6}$$

Proof: a) This is simply follows from the definition of proximal operator.
b) We can simply show that

$$\|x\|_{2v} = \max_{0\le\gamma\le1} (\gamma\|x_1\|_2 + (1-\gamma)\|x_2\|_2)$$

then its Moreau envelop is

$$\begin{aligned}
\mathrm{M}_{R_c(\mathbf{v})}(\mathbf{v}) =& \min_{x_1,x_2} \frac{1}{2}\|x_1-\mathbf{v}_1\|_2^2 + \frac{1}{2}\|x_2-\mathbf{v}_2\|_2^2 \\
&+\lambda \max_{0\le\gamma\le1} (\gamma\|x_1\|_2 + (1-\gamma)\|x_2\|_2) \\
=& \max_{0\le\gamma\le1} \min_{x_1} \frac{1}{2}\|x_1-\mathbf{v}_1\|_2^2 + \lambda\gamma\|x_1\|_2 \\
&\qquad + \min_{x_2} \frac{1}{2}\|x_2-\mathbf{v}_2\|_2^2 + (1-\gamma)\lambda\|x_2\|_2 \\
=& \max_{0\le\gamma\le1} \mathrm{M}_{\lambda\gamma\|.\|_2}(\mathbf{v}_1) + \mathrm{M}_{\lambda(1-\gamma)\|.\|_2}(\mathbf{v}_2)
\end{aligned}$$

■

Proof: [**Theorem 4.2**] The Moreau envelope of $R_c(\mathbf{v})$ is

$$\begin{aligned}
\mathrm{M}_{R_c(\mathbf{v})}(\mathbf{v}) &= \min_{x_1,x_2} \frac{1}{2}\|x_1-\mathbf{v}_1\|_2^2 + \frac{1}{2}\|x_2-\mathbf{v}_2\|_2^2 \\
&\qquad + \lambda \max_{0\le\gamma\le 1} (\gamma\|x_1\|_2 + (1-\gamma)\|x_2\|_2) + \nu_1 R_1(\mathbf{v}_1) + \nu_2 R_2(\mathbf{v}_2) \\
&= \max_{0\le\gamma\le 1} \min_{x_1} \frac{1}{2}\|x_1-\mathbf{v}_1\|_2^2 + \lambda\gamma\|x_1\|_2 + \nu_1 R_1(\mathbf{v}_1) \\
&\qquad + \min_{x_2} \frac{1}{2}\|x_2-\mathbf{v}_2\|_2^2 + (1-\gamma)\lambda\|x_2\|_2 + \nu_2 R_2(\mathbf{v}_2) \\
&= \max_{0\le\gamma\le 1} \mathrm{M}_{\lambda\gamma\|.\|_2+\nu_1 R_1}(\mathbf{v}_1) + \mathrm{M}_{\lambda(1-\gamma)\|.\|_2+\nu_2 R_2}(\mathbf{v}_2)
\end{aligned} \tag{D.7}$$

Let $h_1(\mathbf{v}_1) := \lambda\gamma\|\mathbf{v}_1\|_2+\nu_1 R_1(\mathbf{v}_1)$ and $h_2(\mathbf{v}) := \lambda(1-\gamma)\|\mathbf{v}_2\|_2+\nu_2 R_2(\mathbf{v}_2)$. From [Haeffele et al., 2014, Theorem 3], we know that $\mathrm{prox}_{h_1}(\mathbf{v}_1) = \mathrm{prox}_{\lambda\gamma\|.\|_2}(\mathrm{prox}_{\nu_1 R_1}(\mathbf{v}_1))$ and $\mathrm{prox}_{h_2}(\mathbf{v}_2) = \mathrm{prox}_{\lambda(1-\gamma)\|.\|_2}(\mathrm{prox}_{\nu_2 R_2}(\mathbf{v}_2))$, and so

$$\begin{aligned}
\mathrm{M}_{h_1}(\mathbf{v}_1) &= \mathrm{M}_{\lambda\gamma\|.\|_2}(\mathrm{prox}_{\nu_1 R_1}(\mathbf{v}_1)) \\
\mathrm{M}_{h_2}(\mathbf{v}_2) &= \mathrm{M}_{\lambda(1-\gamma)\|.\|_2}(\mathrm{prox}_{\nu_2 R_2}(\mathbf{v}_2))
\end{aligned} \tag{D.8}$$

Then based on (D.7) and (D.8), we obtain

$$\begin{aligned}
\mathrm{M}_{R_c(\mathbf{v})}(\mathbf{v}) = \max_{0\le\gamma\le 1} &\mathrm{M}_{\lambda\gamma\|.\|_2}(\mathrm{prox}_{\nu_1 R_1}(\mathbf{v}_1)) \\
&+ \mathrm{M}_{\lambda(1-\gamma)\|.\|_2}(\mathrm{prox}_{\nu_2 R_2}(\mathbf{v}_2))
\end{aligned} \tag{D.9}$$

Finally, based on above equation and (D.6), we conclude the following composition rule for

$$\mathrm{M}_{R_c(\mathbf{v})}(\mathbf{v}) = \mathrm{M}_{\lambda\|.\|_{2v}}([\mathrm{prox}_{\nu_1 R_1}(\mathbf{v}_1);\ \mathrm{prox}_{\nu_2 R_2}(\mathbf{v}_2)])$$

and according to Lemma D.1 the proximal operator is

$$\mathrm{prox}_{R_c}(\mathbf{v}) = \mathrm{prox}_{\lambda\|.\|_{2v}}([\mathrm{prox}_{\nu_1 R_1}(\mathbf{v}_1);\ \mathrm{prox}_{\nu_2 R_2}(\mathbf{v}_2)]).$$

■

D.2 Experimental results for discrete value time series

One of the major advantages of formulation equation 4.7 is its natural flexibility to encompass any convex loss function such as the Bregman divergences that associate with exponential family distributions and can express a broad range of data property with non-linear transfers. An application that gains benefits from the aforementioned property is to model the count data process with generalized LDS model and consequently adopting the two view formulation to identify the model parameters. An integer-valued stochastic process, that explains the number of occurrence of one phenomenon, can be properly modeled by Poisson distribution [Macke et al., 2015]. Therefore, the LDS with Poisson distributed observation can be expressed as:

$$\begin{aligned} \boldsymbol{\phi}_{t+1} &= \mathbf{A}\boldsymbol{\phi}_t + \eta_t \\ \mathbf{z}_t &= \mathbf{f}(\mathbf{C}\boldsymbol{\phi}_t) \\ \mathbb{P}(\mathbf{x}_{i,t}|\mathbf{z}_{i,t}) &= \frac{1}{\mathbf{x}_{i,t}!}(\mathbf{z}_{i,t})^{\mathbf{x}_{i,t}} \exp(-\mathbf{x}_{i,t}) \end{aligned} \tag{D.10}$$

where $\mathbf{f}(\boldsymbol{\theta}) = \exp(\boldsymbol{\theta})$. The exponential mapping is not only a natural choice in applications such as neural spike-rate modeling, as explained in [Macke et al., 2015], it also matches with the transfer function associated with the Poisson distribution. Therefore, the negative log-likelihood loss for this model can be characterized by the Bregman divergence, defined as $D_F(\hat{\mathbf{z}}\|\mathbf{z}) := F(\hat{\mathbf{z}}) - F(\mathbf{z}) - \mathbf{f}(\mathbf{z})^\top(\hat{\mathbf{z}} - \mathbf{z})$ where $F(\boldsymbol{\theta}) = \mathbf{1}^\top \exp(\boldsymbol{\theta})$ ($\mathbf{f}(\boldsymbol{\theta}) = \exp(\boldsymbol{\theta})$) is potential (transfer) function corresponding to Poisson distribution.

In Table 3, we compare the performance of the LDS-DV method against the standard N4SID and EM for synthetic time series setting.

For boolean setting, data are sampled from Bernoulli distribution whose mean is changed according to non-linear transfer function of the LDS model

Table 3: Synthetic time series

	Bernoulli d=5 , k=3		Bernoulli d=8 , k=6		Bernoulli d=16 , k=9		Poisson d=5 ,k=3		Poisson d=8 , k=6	
	GOF_b	Time	GOF_b	Time	GOF_b	Time	GOF_p	Time	GOF_p	Time
LDS-MV	**0.66**$_{\pm 0.01}$	3.03	**0.59**$_{\pm 0.01}$	2.40	**0.51**$_{\pm 0.01}$	2.52	**0.59**$_{\pm 0.02}$	0.61	**0.51**$_{\pm 0.02}$	1.70
N4SID	0.77$_{\pm 0.01}$	1.01	0.82$_{\pm 0.01}$	1.68	0.75$_{\pm 0.01}$	4.63	1.40$_{\pm 0.21}$	0.36	1.59$_{\pm 0.33}$	0.49
EM	0.75$_{\pm 0.01}$	2.46	0.67$_{\pm 0.01}$	4.61	0.63$_{\pm 0.01}$	24.17	1.60$_{\pm 0.34}$	1.83	2.53$_{\pm 0.60}$	2.48

Table D.1: Table of prediction performance for binary-values stochastic process. The first column of each dataset is the average goodness-of-fit (GOF) for one step prediction with standard error and the second column is the algorithm runtime in CPU seconds. The best GOF according to pairwise t-test with significance level of 5% *is highlighted.*

where sigmoid transfer function $\mathbf{f}(\boldsymbol{\theta}) = (1 + \exp(-\boldsymbol{\theta}))^{-1}$ is used [Banerjee et al., 2005]. Each test case is averaged over 100 data sequences where data are generated similar to synthetic setting **S1** of section 4.5. For Poisson setting, data are sampled based on model equation D.10 where the final results averaged over 30 data sequences.

Goodness-of-fit for the Bernoulli distribution is the misclassification error: $\text{GOF}_b = \frac{1}{Td}\sum_{t=1}^{T_{test}} \|y_t \neq g(\hat{\mathbf{z}}_t)\|_1$, $g(\theta) = \mathbf{I}_{\theta \geq 0.5}$. And for the Poisson distribution, we define goodness-of-fit as $\text{GOF}_p = \frac{1}{Td}\sum_{t=1}^{T_{test}} \|y_t - h(\hat{\mathbf{z}}_t)\|_1$, $h(\hat{z}) = \text{mode}(P(\hat{z})) = \max_x p(x|\mu = \hat{z})$.

This results are just some primitive results to show the capability of the proposed method in modeling generalized-LDS models.